2021 IEEE 37th International Conference on Data Engineering (ICDE) | 978-1-7281-9184-3/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICDE51399.2021.00139

Knowledge-Aware Group Representation Learning for Group Recommendation

Zhiyi Deng† Changyu Li† Shujin Liu† Waqar Ali† Jie Shao†‡*
†University of Electronic Science and Technology of China, Chengdu, 611731, China
‡Sichuan Artificial Intelligence Research Institute, Yibin, 644000, China
{zhiyideng,changyulve,shujin}@std.uestc.edu.cn waqar.uestc@yahoo.com shaojie@uestc.edu.cn

***Abstract*—Nowadays, going out and participating in group activities is an indispensable part of human life, and group recommendation systems are needed to provide suggestions. In practice, group recommendation faces serious sparsity issues due to the lack of group-item interaction data, and the key challenge is to aggregate group member preference for group decision making. Conventional group recommendations applied a predefined strategy to aggregate the preferences of group members, which cannot model the group decision making process and do not address the data sparsity problem well. In this paper, we introduce knowledge graph into group recommendation as side information, and propose a novel end-to-end method named knowledge graph-based attentive group recommendation (KGAG) to solve the data sparsity and preference aggregation problems. Specifically, a graph convolution network (GCN) is employed to capture abundant structure information of items and users in knowledge graph to overcome the sparsity problem. Besides, to learn knowledge-aware group representation for inferring the group decision better, we capture the user-item connectivity and user-user connectivity in knowledge graph, and then adopt attention mechanism to learn the influence of each member according to user-user interaction in group and the candidate item, for member preference aggregation. Additionally, the attention mechanism can provide interpretability to group recommendation. Moreover, we extend the margin loss to our KGAG which forces the prediction score of positive item to be a distance larger than that of negative item. Experimental results show the superiority of the proposed KGAG and verify the efficacy of each component of KGAG.**

***Index Terms*—group recommendation, knowledge graph, attention mechanism, pairwise loss**

I. Introduction

With the advent of big data era, data begin to grow explosively. In order to solve the problem of information overload, recommendation systems have been widely applied in online services including e-commerce, content sharing, social networking, forum and so on. Group recommendation system is also needed since people can meet more and more conveniently for group activities together, such as eating, watching movies, or taking a tour. These people may be familiar with each other, such as in a family or have a few mutual friends. It can also be a group of people who meet by chance in an activity such as several travelers joining a same tour group. The task of group recommendation is to recommend one or several suitable items for these groups.

Corresponding author: Jie Shao. This work is supported by the National Natural Science Foundation of China (No. 61832001) and Sichuan Science and Technology Program (No. 2019YFG0535).

Groups can be categorized into two types, including persistent groups [1], [2], and occasional groups [3], [4]. A persistent group is predefined such as a family which has stable members and sufficient historical group-item interactions, while an occasional group is formed ad-hoc and its members may not have social relations such as random people taking the same bus. For persistent group recommendation, we can treat each group as a special user, and use the methods of individual recommendation directly. However, as for occasional group, members meet rarely, so the record of group-item interaction is too sparse to learn the preference for it straightforwardly. For occasional group, some researchers tend to learn the score [4], [5] or the preference [6] of each member in the group and then aggregate the scores or the preferences. In this paper, we focus on alleviating the sparsity problem and aggregating the preferences of users in occasional group.

A typical recommendation technique is collaborative filtering (CF). CF assumes that users may be interested in the items which are liked by people who share similar interaction records with them. It learns representation vectors for each user and item and then models their interactions by some specific operations, such as inner product [7] or neural networks [8]. However, conventional CF-based methods have poor performance when user-item interactions are sparse, which is known as sparsity problem. To solve this problem and improve the performance of recommendation system, researchers usually use auxiliary information such as item attributes [9]–[11], user's demographic information [12] or user network [13].

In recent years, knowledge graph has become a kind of popular side information due to its abundant information of the items in recommendation systems. In this paper, we introduce knowledge graph into the group recommendation task and propose an end-to-end knowledge graph-based attentive group recommendation (KGAG) model to learn knowledge-aware group representation for group recommendation. Knowledge graph describes various entities and their relationships existed in the real world, which forms a huge heterogeneous graph. Nodes in the graph represent entities, and edges in the graph represent the relationships between entities. Triple is a general representation of knowledge graph, e.g., (*AlfredHitchcock*, *DirectorOf*, *Psycho*) implies that *Alfred Hitchcock* is the director of *Psycho*. Knowledge graph can not only improve the accuracy of recommendation but also provide the interpretability

978-1-7281-9184-3/21/$31.00 ©2021 IEEE
DOI 10.1109/ICDE51399.2021.00139

to recommendation. The attributes of items and the relations between items help improve the representation of the items, and the connectivities in the knowledge graph can provide the interpretability for recommendation.

For group recommendation, the interaction among user interests in a same group is very important in group decision making. When making group decisions, members in the group usually express their preferences at first, and then discuss within the group to get the final decision. For a user, the more people in the group who have similar interests, the more likely her opinions will be accepted by other members of the group, i.e., the greater her influence in the group. The similarity of two users' interests can be found from the knowledge graph. For example, if Jack is interested in movie *Psycho* and Rose is interested in movie *Rear Window*, it may find that Jack and Rose are both interested in the movies which are directed by *Alfred Hitchcock*. Moreover, more high-order connectivities between two users imply the more similar interests the two users share. Yin et al. [14] introduced social network into group recommendation as the side information and used it to model the user influence in the group. Social network can reflect the general influence of users in the interpersonal circle, but it may not be suitable to model the influence of a user's opinions on a specific item. For example, a person who is active in the group may not know much about movies. When this group wants to see a movie, she will also consult other members of the group who know better about movies. Therefore, she does not necessarily play a major role in this group's decision making. Moreover, in occasional groups, the members may not be closely connected in social networks, but the common preferences are largely based their hobbies rather than their social roles. In this paper, we employ knowledge graph for group recommendation. As shown in Figure 1, each item can be mapped to an entity in knowledge graph by matching the name, so that recommendation system makes contact with the knowledge graph and gets its information. Additionally, since graph convolution network (GCN) can propagate node information along edges of knowledge graph, a node can get information from its neigborhood and hence local feature of entities can be captured. Therefore, a GCN is constructed in our KGAG model. Not only rich information of items can be extracted but also the similarity of user interests which are beneficial to model user-user interaction can be caught in the knowledge graph. Then, we learn influences of members according to the candidate item and user-user interaction of the group.

Most of previous works on group recommendation systems only focused on predefined aggregation strategies, including score aggregation strategies [4], [15] and preference aggregation strategies [6]. These simple predefined aggregation strategies treat all users in the group equally which cannot reflect the process of group decision making and do not fully consider the influencing factors of group decision making. Actually, the influence of each member on the final result should be different and it varies according to other group members and candidate item. For example, when choosing to go to the cinema, each member has different understandings of different types of movies. The studies of [16] and [14] are both in consideration of that the opinions of group members will influence each other. However, Tran et al. [16] did not take into account that the influence of members in group decisions will vary with the candidate items. Yin et al. [14] modeled the interactions between users with social network but there may be no information about user preference in the social network. With knowledge graph, we are able to learn knowledge-aware representations of items and users. An attention mechanism is also adopted to learn users' influence in each group according to user-user interaction and candidate item, which also can provide an explanation to group recommendation result.

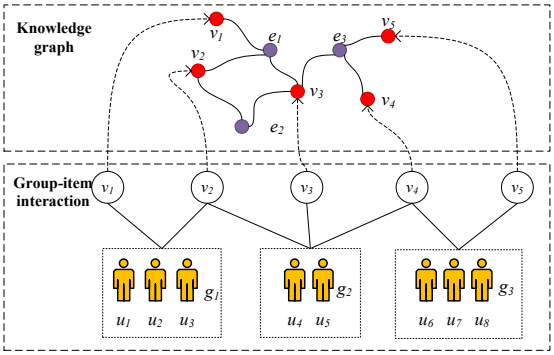

Fig. 1. The dotted box at the bottom shows a group-item interaction example between each group and item, and the above shows a knowledge graph example. Each item is mapped into an entity in the knowledge graph. In the knowledge graph, v_1-v_5 are the entities which have corresponding items in group-item interaction data while e_1-e_3 are not. In the group-item interaction example, three small dotted boxes show three groups g_1, g_2 and g_2 which contain users u_1-u_3, u_4-u_5 and u_6-u_8, respectively. v_1-v_5 are five items.

Our contribution can be summarized as follows:

- Our KGAG model integrates the group recommendation with knowledge graph in which there are abundant structure and semantic information of items helpful for handling the sparsity problem. In addition, a GCN is leveraged to capture the structure and semantic information of items and the similarity between users in terms of interest due to its ability of propagating the information along with the connectivity of knowledge graph.
- We propose an attention mechanism that can learn the influence of each user in a group to better model the group decision-making process. We take both user-user interaction and the influence of candidate item in group into consideration. Besides, the influence values provide reasonable explanations to group recommendation results.
- To learn more discriminative representations of groups and items, we also extend the idea of margin loss which not only requires the score of the positive sample to be higher than that of the negative sample, but also requires the score of the positive sample is higher than that of the negative sample by a given margin to group recommendation. Our experiments show that this loss function improves the performance of KGAG.

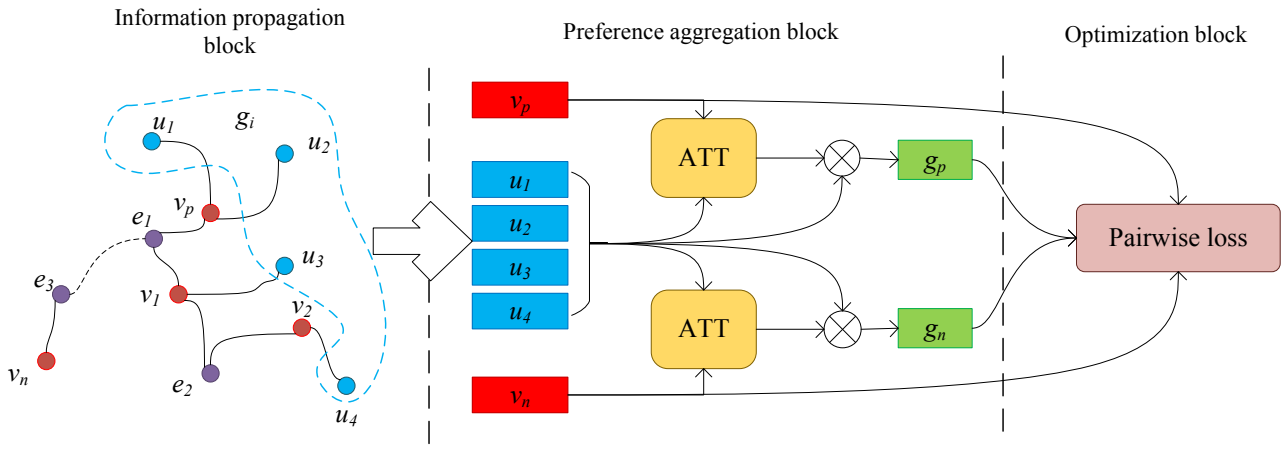

Fig. 2. An overview of the architecture of our KGAG model which consists of three parts. The left part is the collaborative knowledge graph in which blue nodes represent users, red nodes represent items and purple nodes represent the entities which are not user or item. The middle part is the preference aggregation block which aggregates the preference of users. The right part is the optimization block which optimizes the parameters of the model.

II. Related Work

A. Knowledge Graph-based Recommendation

In recent years, generating recommendations with knowledge graph as the side information has attracted researchers' attention, due to its properties that it can not only alleviate the data sparsity problem and cold start problem since it is a supplement to the training data but also provides the interpretability to the recommendation systems. The knowledge graph-based recommendation systems can be mainly divided into three categories, including embedding-based methods, path-based methods, and hybrid methods.

The embedding-based methods use the information from the knowledge graph directly to obtain a better feature representation of items or users in recommendation dataset. As an early knowledge graph-based recommendation algorithm, collaborative knowledge base embedding (CKE) [17] leveraged a knowledge graph embedding algorithm TransR [18] to capture the semantic features of items in knowledge graph. Moreover, other side information such as textual description and the images of items is also used. Deep knowledge-aware network (DKN) [19] was proposed for news recommendation. It utilized the textual news and knowledge graph as the side information. A knowledge graph embedding algorithm TransD [20] was used to obtain the knowledge-level embedding. In addition, some researchers' strategy is multi-task learning. KTUP [21] learns the task of recommendation and knowledge graph completion. It will improve each task because the items in the recommendation task should share the embedding with its corresponding entity in the knowledge graph.

Path-based methods leverage the connectivity patterns including meta-path and meta-graph for recommendation. A path-based method is Hete-MF [22] which leverages the item-item similarity in each meta-path as a component of non-negative matrix factorization method [23]. Meta-path is a path consisted of the entities and the relations on the path while meta-graph is a combination of different meta-paths. The connectivity patterns can provide additional guidance for recommendation. However, both meta-paths and meta-graph need to be predefined manually. It relies heavily on domain knowledge and it is difficult to adopt in practice.

The hybrid methods combine above two methods using the embedding of entity and the connectivity information together whose basic idea is that the representation of entities can be refined by the guidance of the connectivity of the knowledge graph. RippleNet [24] is the first work to introduce the concept of preference propagation. It automatically discovers users' hierarchical potential interests by iteratively propagating users' preferences in the knowledge graph. Graph neural network (GNN) is naturally an excellent tool of hybrid method and the methods inspired by GNN emerge in recent years. KGCN [25] extends GCN approach to knowledge graph-based recommendation which can learn the structure information and semantic information of items in knowledge graph and users' individual preference. KGAT [26] combines the recommendation with the knowledge graph attention network. It employs TransR [18] to obtain the initial embedding of each entity at first, and then recursively propagates the embedding from the entity's neighbors to refine this entity's embedding and applies an attention mechanism to discriminate the importance of the neighbors. KGCN is the state-of-the-art method based on knowledge graph. Therefore, we consider it as a representative of knowledge graph-based individual recommendation and compare our proposed KGAG with it.

B. Group Recommendation

Group recommendation also received a lot of attention in recent years since it needs to be applied in various domains [27]. In general, existing group recommendation can be divided into two categories, including memory-based and model-based methods.

The memory-based methods can be further divided into the score aggregation methods and preference aggregation methods. The score aggregation methods obtain the prediction score of each member in the group by individual recommendation methods at first, and then the prediction score of each member will be aggregated to obtain the prediction score of the group by some predefined score aggregation strategies such as average satisfaction [4], least misery [5] and maximum pleasure [4]. The preference aggregation methods [6] aggregate the profile of each member to form group profile at first, and then employ individual recommendation to make group recommendation.

Different from memory-based methods, model-based methods try to model the group decision making process. PIT [3] considers the personal impact of group members and it makes active people have large impact on the group's decision. However, it assumes that a user has the same impact on different groups which does not reflect the reality in real-world scenario. COM [28] assumes that the personal impacts are topic-dependent and both the group's topic preferences and individuals' preferences influence the final group decision. However, it suffers from the same drawback with PIT. Recently, the aggregation strategies based on attention mechanism emerge. AGREE [1] combines attention network with the neural collaborative filtering method (NCF) [8]. MoSAN [16] models the group decision making process by sub-attention network and it considers about the user-user interaction, but it does not consider that group members' decisions will be influenced by the candidate item. SIGR [14] takes an attention mechanism and a bipartite graph embedding model BGEM as building block and learns the personal social influence in group from both group-item interaction data and social network data. GroupSA [29] treats the group decision making process as a multiple voting process and develop a stacked social self-attention network to model this process. Both SIGR and GroupSA rely on social network. Compared with structure information in item-side such as knowledge graph, social network is more enormous and sparse. For example, the number of audiences is much larger than the number of movies. Moreover, the item-side information is not related to privacy, so it will be easier to get than user-side information. Hence, the item-side information such as knowledge graph is better than the user-side information such as social network. In addition, Deng et al. [30] propose to use knowledge graph in group recommendation, but they use a simple hand-craft way to get representations of groups and items which cannot capture the information of knowledge graph adequately.

III. Our Proposed Framework

A. Problem Formulation

In our knowledge graph-based group recommendation, we take both user-item interaction history and group-item interaction history into account. Suppose that there are m users $\mathcal{U} = \{u_1, u_2, u_3, ..., u_m\}$ and n items $\mathcal{V} = \{v_1, v_2, v_3, ..., v_n\}$. The groups are the sets whose elements are users such as $g_i = \{u_{i,1}, u_{i,2}, u_{i,3}, ..., u_{i,l}\}$, and we suppose that there are k groups $\mathcal{S}_g = \{g_1, g_2, g_3, ..., g_k\}$. The user-item interaction matrix $Y^U \in \mathcal{R}^{m \times n}$ is defined according to users' implicit feedback, and $y^U_{u,v} = 1$ indicates that user u engages with item v. Similarly, the group-item interaction matrix $Y^G \in \mathcal{R}^{k \times n}$ and $y^G_{g,v} = 1$ indicates that item v is selected by group g.

In our paper, knowledge graph is introduced as side information which is represented by entity-relation-entity triples in the form of (h, r, t) (which also called a fact). Here, $h \in \mathcal{E}$, $r \in \mathcal{R}$ and $t \in \mathcal{E}$ are head entity, relation, tail entity of the triple respectively. $\mathcal{E}$ is the entity set and $\mathcal{R}$ is the relation set. The knowledge graph is represented as $\mathcal{G} = \{(h, r, t) | h \in \mathcal{E}, t \in \mathcal{E}, r \in \mathcal{R}\}$. In the knowledge graph-based recommendation, items should be mapped into entities of the knowledge graph. So, there is a mapping function f: $\mathcal{V} \rightarrow \mathcal{E}$ which is a single shot. $f(v) = e$ indicates that item v can be mapped to entity e in the knowledge graph.

As what the researchers do in [26], we create a collaborative knowledge graph in which the user and the user-item interaction are added. The user behaviors are represented as triplets $(u, Interact, e)$. When $y^U_{u,v} = 1$, triplet $(u, Interact, f(v))$ will be added. In the collaborative knowledge graph, $\mathcal{E}' = \mathcal{E} \bigcup \mathcal{U}$ and $\mathcal{R}' = \mathcal{R} \bigcup \{Interact\}$.

Input: A set of users $\mathcal{U}$, a set of items $\mathcal{V}$, a set of group $\mathcal{S}_g$, user-item interactions Y^U, group-item interactions Y^G and knowledge graph $\mathcal{G}$.

Output: A prediction function $\hat{y}_{g,v} = \mathcal{F}(g, v | \Theta, Y^U, Y^G, \mathcal{G})$, where $\hat{y}_{g,v}$ denotes the probability that group g will select item v, and Θ denotes the model parameters.

B. Overview

As shown in Figure 2, our KGAG model consists of three parts, including information propagation block, preference aggregation block and optimization block. u_1-u_4 are all the members of group g_i while v_p and v_n are the positive item ($y^G_{g_i,v_p}$=1) and the negative item ($y^G_{g_i,v_n}$=0) of group g_i, respectively. Solid lines indicate edges in collaborative knowledge graph while dotted line indicates a long path. After constructing the collaborative knowledge graph, the knowledge-aware representation of items and users will be learned by GCN in information aggregation block, in the form of node representation update and information propagation along the edges in the graph. After that, not only more structure information and semantic information can be captured but also high-order connectivity between different users which implies potential similarity of their interests can be captured by implementing GCN in collaborative knowledge graph. Then, a candidate item related attention mechanism is leveraged to learn the importance of each member in the group decision making and aggregate the members' preferences to form the group preference in the preference aggregation block. Finally, we extend the idea of margin loss to our model, which enforces the prediction score of positive item should be higher than that of negative item by a margin in the optimization block. The details of each part are described below.

C. Information Propagation Block

As stated above, we can construct the collaborative knowledge graph using the knowledge graph and user-item interaction data. In this section, we present how the information propagates and how the information aggregates to learn a high-order representation of user/item.

No matter for user node, item node or other entity, we all treat it as a node e and initialize a zero-order trainable representation $\mathbf{e}^0$ for each e. In this stage, the information of direct neighbors and indirect neighbors reachable by a path from e is aggregated to form the higher-order representation of the entities recursively.

1) A Propagation Layer: In a single propagation layer, the representation of e's direct neighbors propagates to the node e and aggregates together (which is called neighbor aggregation). Then, the aggregation representation of e's direct neighbors and the representation of the node e will be integrated to form the higher-order representation of e (which is called representation update). In each single layer, a node can only see or be seen by its direct neighbors. In other words, it can only propagate the information along first-order connectivity. Details are as follows.

Neighbor aggregation. At first, the information of e's direct neighbors should be propagated to e and we use $\mathcal{N}_e$ to denote the direct neighbors of e. The strict definition of $\mathcal{N}_e$ is that $\mathcal{N}_e = \{e_t|(e, r, e_t) \in \mathcal{G}\}$. For each neighbor, the importance should not be equal. We compute the aggregation representation of e's neighbors by computing a weighted sum of them:

$$\mathbf{e}_{N_e} = \sum_{(e,r,e_t)\in\mathcal{G}} \tilde{\pi}(e, r, e_t)\mathbf{e}_t. \tag{1}$$

$\tilde{\pi}(e, r, e_t)$ is the weight of e_t when computing e's neighbor aggregation representation and r is the relation between entities e and e_t. The weight of neighbor e_t should be related to the relation r. Considering that this is a component of the recommendation system, we take the representation of e's interaction object i_e in user-item or group-item interaction data into consideration. Because if e is an entity which is corresponding to an item whose interaction object is a user or a group, that user or group may be only interested in a part of the attributes of this item. Simply, we adopt the inner product of the representation of relation r and the representation of e's interaction object to represent $\tilde{\pi}(e, r, e_t)$:

$$\pi(e, r, e_t) = \mathbf{i}_e \cdot \mathbf{r}. \tag{2}$$

Concretely, if e is a member of a group, i_e is the item which e's group interacts with, and if e is an item, $\mathbf{i}_e$ is the equal weight average of the zero-order representation of the members on the group e interacts with. $\mathbf{r}$ is the trainable representation of relation r. Then, the weight should be normalized by adopting the softmax function:

$$\tilde{\pi}(e, r, e_t) = \frac{exp(\pi(e, r, e_t))}{\sum_{(e,r_i,e_i)\in\mathcal{G}} exp(\pi(e, r_i, e_i))}. \tag{3}$$

Representation update. The another component of the single propagation layer is to update a higher-order representation for node e by combining its neighbor aggregation representation and its original representation. The general form of this process is:

$$\mathbf{e}' = f_{aggregate}(\mathbf{e}, \mathbf{e}_{N_e}), \tag{4}$$

where $\mathbf{e}'$ is the higher order representation of e and $f_{aggregate}(\cdot)$ is the aggregation function. The aggregation function can be implemented with two kinds of classical graph neural networks GCN [31] or GraphSage [32], and the performance of them will be compared in our experiments.

- **GCN Aggregator** [31] computes the summation of two representation vectors above and then feed it into a nonlinear transformation:

$$f_{GCN} = \sigma(\mathbf{W} \cdot (\mathbf{e} + \mathbf{e}_{N_e}) + \mathbf{b}), \tag{5}$$

  where $\mathbf{W} \in \mathcal{R}^{d\times d}$ is a trainable transformation weight and $\mathbf{b} \in \mathcal{R}^d$ is a trainable bias. σ is a nonlinear function which can be ReLu, sigmoid and so on.
- **GraphSage Aggregator** [32] concatenates two representation vectors above before feeding them into the nonlinear transformation:

$$f_{GraphSage} = \sigma(\mathbf{W} \cdot concat(\mathbf{e}, \mathbf{e}_{N_e}) + \mathbf{b}), \tag{6}$$

  where $concat(\cdot, \cdot)$ is the concatenation operation of two vectors. $\mathbf{W} \in \mathcal{R}^{2d\times d}$ and $\mathbf{b}$ are trainable transformation weight and bias, respectively.

In general, for an entity e, the information of its neighbors propagates to it and aggregates together to update a higher-order representation of e in a single propagation layer. As a result, the entity e not only contains the information of itself but also the information of its neighbors flowing to e along the first-order connectivity from its direct neighbors.

2) High-order Information Propagation: The single propagation layer propagates the information along the edge for one step. Therefore, the reception field is determined by the depth of propagation layer and capacity of single propagation layer network is limited. For example, in the left part of Figure 2, if the model consists of a single propagation layer, only the information of v_p can propagate to u_1 and u_2. At the same time, u_3 only contains the information of v_1. Thus, it is ineffective to consider the interaction between two users in a same group. In order to overcome this drawback, we stack up a series of propagation layers to make the information propagate along the high-order connectivity to transmit it further. When we stack two propagation layers, u_1 and u_3 can also contain the information of e_1, and then the interaction between u_1 and u_3 can be considered.

The input of the first layer of entity e is its zero-order representation $\mathbf{e}^0$. The output of a layer is the input of the next layer. Without losing its generality, we compute entity e's representation $\mathbf{e}^h$ in the h-th layer. As what we do in the single propagation layer, we aggregate the information from e's neighbor first:

$$\mathbf{e}_{N_e}^{(h-1)} = \sum_{(e,r,e_t)\in\mathcal{G}} \tilde{\pi}(e, r, e_t)\mathbf{e}_t^{(h-1)}, \tag{7}$$

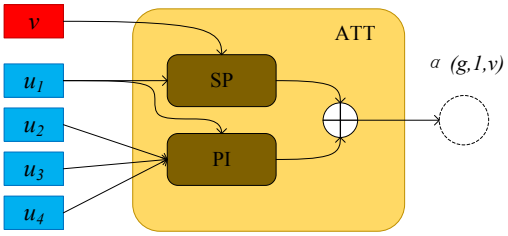

Fig. 3. An illustration of our attention mechanism. Taking u_1 in group g as an example, as in Figure 2, u_1-u_4 are the members of g and v is an item. SP (self persistence) and PI (peer influence) are two components of the attention mechanism.

where $\mathbf{e}_t^{h-1}$ is the representation of e_t in $(h-1)$-th layer, and then we update the representation of e in this layer:

$$\mathbf{e}^h = f_{aggregate}(\mathbf{e}^{(h-1)}, \mathbf{e}_{N_e}^{(h-1)}), \tag{8}$$

where $\mathbf{e}^{(h-1)}$ is the representation of e in $(h-1)$-th layer. Starting from the first layer with embedding of each entity as input, we repeat the process above recursively and the information will propagate node by node in the graph. In other words, if we construct a network with H layers, the representation of entity e in the last layer can capture the information of the entities which can be reached within H steps by e. However, not only the time complexity and the memory cost will increase exponentially with the increase of H, but also a lot of extra noise will inevitably be introduced when H is too large. We will evaluate the effect of H in our experiment.

D. Preference Aggregation Block

As shown in Figure 2, after obtaining the high-order representations of users and items as well as capturing the high-order user-user connectivity in information propagation block, these representations are fed into preference aggregation block to get the representation of the whole group. The attention mechanism to learn users' influences in their group is the most significant part in the preference aggregation block. As shown in Figure 3, our attention mechanism mainly consists of two parts, namely *self persistence* (SP) and *peer influence* (PI).

Empirically, the voice of a person is not only related to her peers, but also the topic they are discussing. In the scenario of recommendation system, the topic is the candidate item of the group. Without loss of generality, we show how to obtain the importance of u_i in group g with candidate item v. Based on life experience, we made a hypothesis in the model that the more interested a user is in the candidate item, the more consistent she will be in group decision making. In our attention model, the SP part is to measure the extent to which she sticks to her own opinion. We compute the preference score of candidate item in the self persistence part as:

$$\alpha_{SP}(g, i, v) = f_{pref}(\mathbf{u}_i, \mathbf{v}), \tag{9}$$

where $f_{pref}(\cdot, \cdot)$ is the preference score function. In our implementation, inner product is used as the preference score function. In individual recommendation, it is usually used as the prediction score function and the higher the inner product value means the user is more likely to like the item.

The PI part in our attention model is to model the influence of her peers on her importance. At first, we need to define the peer set of a user in her group. The peer set of u_i in group g is a set which includes all the members in g except user u_i, and can be denoted as $S_{g,i}^P = \{u | u \in g, u \neq u_i\}$. Then, we compute the importance of u_i from her peer set:

$$\alpha_{PI}(g, i) = \mathbf{v}_c^T \Phi(\mathbf{W}_{c1}\mathbf{u}_i + \mathbf{W}_{c2} CONCAT_{u \in S_{g,i}^P}(\mathbf{u}) + \mathbf{b}), \tag{10}$$

where $\mathbf{v}_c \in \mathcal{R}^d$, $\mathbf{W}_{c1} \in \mathcal{R}^{d \times d}$, $\mathbf{W}_{c2} \in \mathcal{R}^{d \times d \cdot |S_{g,i}^P|}$ and $\mathbf{b}$ are all the trainable parameters of peer influence. $\mathbf{W}_{c1}$ and $\mathbf{W}_{c2}$ are weight matrices. $\mathbf{v}_c$ is a weight vector while $\mathbf{b}$ is a bias. $CONCAT(\cdot)$ is a function that concatenates all the representations of the users in $S_{g,i}^P$. $\Phi(\cdot)$ is a nonlinear function and we use ReLU in our model. At this point, we have modeled how strong the user u_i's importance from her peers influence in her group. Then, we combine the preference score of u_i in candidate item and her importance from her peers in the group by simply adding them:

$$\alpha(g, i, v) = \alpha_{SP}(g, i, v) + \alpha_{PI}(g, i), \tag{11}$$

where $\alpha(g, i, v)$ is the overall importance of user u_i in group g with candidate item v. As what we do to obtain the importance of user u_i in g, the importance of other users in g can be obtained. Before aggregating each member's preference to form the preference of the group, the importance score should be normalized and we normalize it by softmax function:

$$\tilde{\alpha}(g, i, v) = \frac{exp(\alpha(g, i, v))}{\sum_{u_j \in g} exp(\alpha(g, j, v))}. \tag{12}$$

Therefore, the preference of the users in g with candidate item v can be aggregated with their importance values:

$$\mathbf{g} = \sum_{u_i \in g} \tilde{\alpha}(g, i, v)\mathbf{u}_i, \tag{13}$$

where $\mathbf{g}$ is the final representation of group g which is aggregated by knowledge-aware representations of users in it. By adopting the attention mechanism, the above processing is self-adaptive.

E. Optimization Block

In this section, we regard the group recommendation task as a ranking problem and adopt the pairwise loss function to make our model able to distinguish the item a group will select (positive item) and the item they will not select (negative item) by giving the positive item a higher score. To further make the model learn more discriminative representations of groups and items, we extend the idea of margin loss to recommendation.

When we train the model with pairwise loss function, its input is a triplet consisted of a group g, a positive item v_p which is selected by group g and a negative item v_n which is not selected by group g. At first, we compute the prediction scores of positive item and negative item with the

representations of items learned in information aggregation block and the representation of group learned in preference aggregation block, respectively. We regard the inner product of the representation of the group g and the item as the prediction score:

$$\hat{y}_{g,v_p} = \mathbf{g}_p \cdot \mathbf{v}_p, \tag{14}$$

$$\hat{y}_{g,v_n} = \mathbf{g}_n \cdot \mathbf{v}_n. \tag{15}$$

A widely used pairwise loss for top-k recommendation is Bayesian personalized ranking (BPR) loss [33]. To improve the effect of our model, we extend the idea of margin loss by not only requiring that the prediction value of positive item is higher than that of negative item, but also further requiring that the score of positive item is higher than that of negative item by a certain margin. Compared with the BPR loss, it requires more in the difference between the prediction score of positive item and that of negative item so that it can enhance the model to learn more discriminative representation. $\hat{y}_{g,v_p}$ and $\hat{y}_{g,v_n}$ need to satisfy the following inequality:

$$\sigma(\hat{y}_{g,v_p}) - \sigma(\hat{y}_{g,v_n}) \geq \mathcal{M}, \tag{16}$$

where $\mathcal{M}$ is the margin between the prediction score of positive item and the prediction score of negative item and it is a hyper-parameter. $\sigma(\cdot)$ is the sigmoid function to normalize the prediction score. Then, we transform this inequality into the form of loss function:

$$\mathcal{L}_{group} = \sum_{(g,v_p,v_n)\in\mathcal{O}_G} max(\sigma(\hat{y}_{g,v_n}) - \sigma(\hat{y}_{g,v_p}) + \mathcal{M}, 0). \tag{17}$$

To further alleviate the sparsity problem, we also introduce user-item interaction information in model training. Considering that our task is group recommendation, we only use log loss to handle the user-item interaction information:

$$\begin{aligned}\mathcal{L}_{user} = \sum_{u_i\in\mathcal{U},v_j\in\mathcal{V}} & - y^U_{u_i,v_j} log\sigma(\hat{y}^U_{u_i,v_j}) \\ & - (1 - y^U_{u_i,v_j}) log(1 - \sigma(\hat{y}^U_{u_i,v_j})),\end{aligned} \tag{18}$$

where $\hat{y}^U_{u_i,v_j}$ is the prediction score for user u_i to item v_j. It is also the inner product of the representation vector of user and item:

$$\hat{y}^U_{u_i,v_j} = \mathbf{u}_i \cdot \mathbf{v}_j. \tag{19}$$

Finally, we combine two losses by a weighting coefficient:

$$\mathcal{L} = \beta\mathcal{L}_{group} + (1 - \beta)\mathcal{L}_{user} + \lambda\|\Theta\|^2, \tag{20}$$

where the hyper-parameter β is the weight of the group ranking loss and Θ denotes the parameters of our model. We perform mini-batch training, where each mini-batch contains both user-item and group-item interactions to minimize the loss in Eq. 20 with adaptive moment estimation (Adam).

Time Complexity Analysis. We assume that the number of entities' neighbors, the size of training set and the number of epochs to reach convergence are K, n_t and N. The computational graph of information propagation block is a tree. For each training instance, the computational complexity of the h-th layers in information propagation block is $O(K^{H-h}(d^2 + dK)) = O(K^{H-h}d^2)$, where K is often small and less than d. The time complexity of information propagation block is $O(\sum_{h=1}^{H} d^2K^{H-h})$. For preference aggregation block, the computational complexity is $O(d^2)$ which has omitted constant coefficient and lower item. The computational complexity of optimization block is $O(d)$ which is caused by inner product. Finally, the overall training time complexity of KGAG is $O(Nn_t\sum_{h=1}^{H} d^2K^{H-h} + Nn_td^2)$.

IV. Experiments

A. Research Questions

We conduct experiments to answer the following research questions and validate our technical contributions:

- **RQ1:** How does our proposed KGAG model perform compared with other group recommendation methods?
- **RQ2:** How do different components (i.e., knowledge graph, attention mechanism, and our novel pairwise loss function) affect KGAG?
- **RQ3:** How do the hyper-parameters affect the performance of KGAG?
- **RQ4:** Can KGAG provide reasonable explanations about group recommendation results?

B. Datasets

We utilize MovieLens-20M and Yelp in our experiments. MovieLens-20M is a widely used benchmark dataset in movie recommendation and consists of approximately 20 million explicit ratings (ranging from 1 to 5) on the MovieLens website, and Yelp is a real-world dataset crawled from a website which allows users and their friends to share their check-ins about local businesses such as restaurants.

Besides the user-item interactions, we need to construct knowledge graph for the dataset. For MovieLens-20M, we follow the way in [25] to map items into Microsoft Satori via matching items' names with the tail of triples $(head, type.object.name, tail)$ and remove the items with multiple matched or no matched entities. After that, there are 138159 users, 16954 items and 13501622 user-item interaction records in MovieLens-20M and 102569 entities, 32 relations and 499474 triples in the knowledge graph. There is not ready-made knowledge graph about the business in Yelp dataset. We construct a knowledge graph from the information of business such as attributes, location and categories. After that, there are 1751137 entities, 17 relations and 4022683 triples in the knowledge graph of Yelp.

In addition, we need to construct group recommendation dataset from MovieLens-20M. We follow the approach in [4] and extract two datasets including MovieLens-20M-Rand and MovieLens-20M-Simi from MovieLens-20M. The groups in MovieLens-20M-Rand are structured adopting the users in MovieLens-20M randomly and have not any restriction on the user-to-user similarity between its members so it led to a lower overall inner group similarity. In real-life situations, the random group corresponds to a set of persons without any

TABLE I
DATASET STATISTICS.

	MovieLens-20M-Rand	MovieLens-20M-Simi	Yelp
Total groups	49472	29670	19322
Total items	3413	3413	1130
Total users	5802	5802	3511
Group size	8	5	3
Interactions	249596	332021	19442
Inter./group	5.05	11.19	1.00

social relations, such as random people taking the same bus or a group of people gathered in an accidental event and they want to go to the cinema together. The groups in MovieLens-20M-Simi are the groups whose members are relatively similar and share some similar tastes such as some friends in a same interest community or some similar interest organizations. As in [4], we compute the similarity between users using Pearson correlation (PCC). As what [4] has done, we adopt 0.27 as the threshold of the similarity and the similarity between two users in a same group in MovieLens-20M-Simi should be 0.27 at least. For a given group, if every member in the group gives a rating to movie which is higher than 4 or equal to 4, we consider that the group will select this movie and the movie is a positive item. Besides, since Yelp does not contain explicit group information, we extract implicit group activities as follows: if a set of users who are friends visit the same restaurant or attend the same event at the same time, they are the members of a group and the corresponding activities are group activities. The statistics of these datasets are shown in Table I. For our experiments, we randomly split each dataset into training set, validation set and test set with the ratio of 60%, 20% and 20%.

C. Evaluation Metrics

We cast the group recommendation task as a ranking problem. To evaluate the performance of a group recommendation method, we first obtain the prediction scores to each item in test set for all the groups in test set. Then, we rank all the items in a list of each group according to the prediction scores decreasingly. Finally, we employ the following standard metrics to measure the recommendation accuracy.

One evaluation metric we adopt is Hit ratio [34]. For each group in the test set, we pick k items at the top of the list after ranking according to prediction score decreasingly. If there is an item v_i satisfying $y^G_{g,v_i} = 1$ for group g, we have a *hit* for g in top-k. Otherwise, we have a *miss*. The *hit@k* metric is define as the ratio of the groups which have a *hit* in their top-k items in the test set:

$$hit@k = \frac{Number\ of\ Hit@k}{|\mathcal{S}^G_{test}|}, \tag{21}$$

where $\mathcal{S}^G_{test}$ is the set of groups in test set and *NumberofHit@k* is the number of groups with at least a *hit* in their top-k items. A better group recommendation method should have a larger *hit@k* value.

TABLE II
PERFORMANCE COMPARISON OF OUR MODEL KGAG AND OTHER METHODS ON DIFFERENT DATASETS.

	MovieLens-20M-Rand		MovieLens-20M-Simi		Yelp	
	rec@5	hit@5	rec@5	hit@5	rec@5	hit@5
CF+LM	0.1440	0.4901	0.1808	0.6556	0.6954	0.6954
CF+MP	0.1331	0.4437	0.1769	0.6887	0.6821	0.6821
CF+AVG	0.1343	0.4570	0.1775	0.6556	0.6887	0.6887
KGCN+LM	0.1584	0.4834	0.1699	0.6159	0.7219	0.7219
KGCN+MP	0.1501	0.4636	0.1658	0.6026	0.7351	0.7351
KGCN+AVG	0.1532	0.4834	0.1687	0.5828	0.7152	0.7152
MoSAN	0.1482	0.4967	0.1667	0.6093	0.5960	0.5960
KGAG	**0.1627**	**0.5497**	**0.1913**	**0.7417**	**0.7748**	**0.7748**

Another widely adopted evaluation metric, recall ratio, is leveraged to evaluate the performance of group recommendation in our experiment as well. We also pick k items at the top of ranking list. If item v_i satisfies $y^G_{g,v_i} = 1$ for group g and it is in the top-k ranking list, it has been recalled. *rec@k* of g is the ratio of recalled items in g's positive items. *rec@k* of the test set is the average value of groups in this test set.

In our experiment, we will report the *hit@5* and *rec@5* values of KGAG and other state-of-the-art methods and then make a comparison.

D. Baseline Methods

For group recommendation, we compare our proposed KGAG model with the following methods including state-of-the-art baselines and the combination of individual recommendation and conventional aggregation strategies for group recommendation.

- **CF for individual recommendation** [35]: We combine a popular and effective CF method for individual recommendation, matrix factorization (MF), with three predefined static score aggregation methods including average satisfaction [4] (**CF+AVG**), least misery [5] (**CF+LM**) and maximum pleasure [4] (**CF+MP**) to make group recommendation. They use maximum average, minimum and maximum of member prediction scores for group representation, respectively.
- **KGCN** [25]: KGCN is a state-of-the-art knowledge graph-based individual recommendation which introduces a graph convolutional network to discover both high-order structure information and semantic information of items in knowledge graph. We extend it by combining with the three aggregation methods mentioned above (**KGCN+AVG**, **KGCN+LM**, and **KGCN+MP**).
- **MoSAN** [16]: MoSAN is a state-of-the-art group recommendation method, with a sub-attention network module to simulate the decision making process of a user under the presence of other group members. In the original MoSAN, user-context vector is used to differentiate the ownership of each attention sub-network while we use knowledge graph in our method. For a fair comparison,

we replace the user-context vector by the knowledge-aware user representation in the knowledge graph.

To make it fair, we feed not only the group-item interaction information but also the user-item interaction information into each baseline method to be compared in the training stage. All the loss functions of those methods are combined by group recommendation loss and individual recommendation loss, as shown in Eq. 20. Note that, although there are other related group recommendation methods such as PIT [3] and COM [28], they have been outperformed by MoSAN so we compare KGAG with MoSAN but not PIT and COM. Other existing group recommendation studies such as SIGR [14] and GroupSA [29] rely on social network and cannot be implemented in our dataset.

E. Overall Performance Comparison (RQ1)

We conduct experiments on the MovieLens-20M-Rand, MovieLens-20M-Simi and Yelp datasets to compare the performance of different models on the group recommendation task. Table II shows the overall performance of our proposed model as well as the baseline methods. From the experiment results, we can observe that:

- Our proposed KGAG consistently provides the best performance compared with other baseline approaches on all three datasets in both two evaluation metrics. As we can see from Table II, KGAG improves the group recommendation significantly. KGAG outperforms all three KGCN-based approaches including KGCN+LM, KGCN+MP and KGCN+AVG. This verifies the significance of our attentive group preference aggregation and our new pairwise loss function. Moreover, compared with MoSAN, KGAG justifies that it has sufficient ability to distill information from knowledge graph to learn better representations of groups and items to make group recommendation.
- The approaches with the least misery aggregation strategy perform better than the approaches with maximum pleasure and average satisfaction on MovieLens-20M-Rand and MovieLens-20M-Simi. The main reason is that we suppose when every member in the group gives a rating to movie which is higher than 4 or equal to 4, the group will select this movie. The least misery strategy fits this assumption better.
- All models perform better on MovieLens-20M-Simi than on MovieLens-20M-Rand. The main reason is that the groups in MovieLens-20M-Simi have more group-item interaction records than the groups in MovieLens-20M-Rand. For CF-based approaches, their improvement in performance from MovieLens-20M-Rand to MovieLens-20M-Simi illustrates that they suffer from sparsity problem although they have leveraged user-item interaction information to alleviate the sparsity problem. As for KGAG, the users on MovieLens-20M-Simi have more partial items hence more high-order connectivities between users can be captured in MovieLens-20M-Simi than on MovieLens-20M-Rand whose users in a group may be far from each other in the knowledge graph.

TABLE III
PERFORMANCE COMPARISON OF OUR PROPOSED MODEL KGAG AND OTHER SIMPLIFIED VERSIONS WITH THE EVALUATION METRICS *rec@5* AND *hit@5* ON MOVIELENS-20M-RAND DATASET.

	rec@5	*hit@5*
KGAG	**0.1627**	**0.5497**
KGAG-KG	0.1530	0.4636
KGAG-SP	0.1567	0.5166
KGAG-PI	0.1582	0.5298
KGAG (BPR)	0.1525	0.5099

- The performance of KGAG validates that it can really work in real-world dataset. In addition, models perform better on Yelp than on MovieLens-20M-Simi and MovieLens-20M-Rand. The groups on MovieLens-20M-Simi and MovieLens-20M-Rand are sampled randomly, so members in a group may be scattered in knowledge graph. However, the members in Yelp will be more centralized which lead to a better performance.

F. Ablation Experiments (RQ2)

The overall performance comparison shows that KGAG obtains the best results, demonstrating the effectiveness of the integrated end-to-end solution. In this section, we will illustrate the importance of each component of our proposed model KGAG by ablation experiments. We will compare KGAG with four weak versions which are listed as follows:

- **KGAG-KG** This is a weak version of KGAG. It does not start from the knowledge graph and feeds the embedding of users and items into the preference aggregation block directly. In simple terms, it removes the information propagation block of our model KGAG. It is to verify the effectiveness of the use of the knowledge graph in group recommendation.
- **KGAG-SP** This is a weak version of KGAG in that it removes the SP (self persistence) component of the attention mechanism in the preference aggregation block. This is to verify whether the influence of the candidate item on the importance of users in their group is important in the preference aggregation of group recommendation.
- **KGAG-PI** This is another weak version of KGAG in the attention mechanism that it removes the PI (peer influence) component of the attention mechanism. This is to verify whether the influence of other users in the same group on the importance of users in their group is important in the preference aggregation of group recommendation.
- **KGAG (BPR)** This is a different version of KGAG which replaces the new pairwise loss we proposed by a widely used pairwise loss Bayesian personalized ranking (BPR). It is to verify the effectiveness of our new pairwise loss which forces the prediction value of positive item to be larger than that of negative item by a certain margin.

The performance comparison of our proposed model KGAG and four weak versions of KGAG is showed in Table III. We have the following observations:

- **Effect of the introduction of knowledge graph:** To illustrate the effect of the introduction of knowledge graph in the group recommendation task, we make a comparison between KGAG and KGAG-KG. As shown in Table III, KGAG significantly outperforms KGAG-KG on MovieLens-20M-Rand which shows that the information propagation block really works and the knowledge graph is beneficial to learn a good representation for the group since not only the knowledge graph can provide more abundant semantic and structural information to the item but also the high-order user-user and user-item connectivity relation can be captured by the propagation of the information along the edges. It is also worth noting that the performance of KGAG-KG is even worse than KGAG-SP and KGAG-PI which cut the attention mechanism of preference aggregation block, which inspires us that in addition to finding out an effective preference aggregation strategy, how to learn an appropriate representation of user and item should also be taken seriously.
- **Effect of the attention mechanism in preference aggregation:** To validate the effect of two components including SP and PI of the attention mechanism to learn the influences of group members in group decision making in preference aggregation block, we conduct another ablation study to compare KGAG with KGAG-SP and KGAG-PI. From the results presented in Table III, KGAG achieves a significant improvement over KGAG-SP and KGAG-PI. The results of the ablation experiment KGAG-SP demonstrates that our assumption that the more a user like the candidate item of her group, the more she will stick to her preferences and has higher influence in the group decision making is true. The result of KGAG-PI indicates that the user-user interaction in the group also has impact on the weight of group members in group decision making as some prior group recommendation works have demonstrated. In this ablation experiment, the performance of KGAG-SP is worse than that of KGAG-PI, which illustrates that SP (self persistence) is a more important component than PI (peer influence) of KGAG on MovieLens-20M-Rand at least. The reason may be that the random group is formed ad-hoc whose members have relatively little in common in their interests and the intra-group interaction is less important than the individual preferences of the group members.
- **Effect of our novel pairwise loss function:** To illustrate the effect of our novel pairwise loss function in model optimization stage, we conduct another ablation study to compare KGAG with KGAG (BPR). As shown by the experiment result in Table III, KGAG significantly outperforms KGAG (BPR) on MovieLens-20M-Rand which illustrates that our novel pairwise loss function has a stronger ability to force the model to learn discriminative representations of groups and items than the conventional pairwise loss function BPR. Our pairwise loss function can significantly improve the performance of the model in ranking problem.

G. Impact of Hyper-parameters (RQ3)

In this section, we evaluate the effect of different hyper-parameters on the performance of our model on MovieLens-20M-Simi. We adjust each hyper-parameter and analyze the influence of the parameters on the performance of the model by combining the experimental results of different values of each parameter. The results are shown in Figure 4, Figure 5 and Table IV.

To study the impact of the dimension d of various representation vectors in our model, we investigate the performance of our model by varying the values of d from 16 to 64. As shown in Figure 5, with the increase of dimension d, the performance firstly increases and then decreases. When d is too small, the representation vector has not enough capacity to encode the information in interaction matrix and knowledge graph. However, due to the sparsity of group-item interaction data, too large d will make the model be trapped in overfitting.

The number of the propagation layers H controls the range that an entity in knowledge graph can aggregate information from and the length of the connectivity can be captured. We test the performance of our model by varying the value of H from 1 to 3. The requirement of memory increases exponentially with the increasing H. When H is large than 3, we do not have enough memory. The performance firstly increases and then decreases with the increasing H. When H is too small, the information in knowledge graph can not be aggregated adequately to learn knowledge-aware representation of user and item. Moreover, some useful high-order connectivity cannot be captured. However, when H is too large, excessive noise will be introduced.

$\mathcal{M}$ controls the margin between the prediction score of positive item and that of negative item in our pairwise loss function. To test the impact of $\mathcal{M}$, we conduct the experiment by varying the value of $\mathcal{M}$ from 0.2 to 0.6. The performance also increases firstly and then begins to decrease with the increasing $\mathcal{M}$. When $\mathcal{M}$ is too small, the loss is easy to approach 0, but the model may have not enough capability to distinguish the positive item and negative item of a group. When $\mathcal{M}$ is too large, the model may be difficult to converge.

β controls the weight of the group ranking loss in the loss function of our model. It would influence the performance of our model. To study its impact, we investigate the performance of our model by varying the value of β from 0.5 to 0.9. From the results in Figure 5, we observe that the recommendation accuracy of our model increases firstly and then begins to decrease with the increase of β. The use of user-item interaction data can alleviate the sparsity problem of group recommendation and if β is too small, the user-item interaction data has not been fully utilized to alleviate sparsity of group-item interaction data. However, when β is too large, the model will not pay enough attention to the group recommendation task.

As shown in Table IV, the KGAG with GCN aggregator performs better than the KGAG with GraphSage aggregator. One possible reason is that GCN aggregator tries to model the

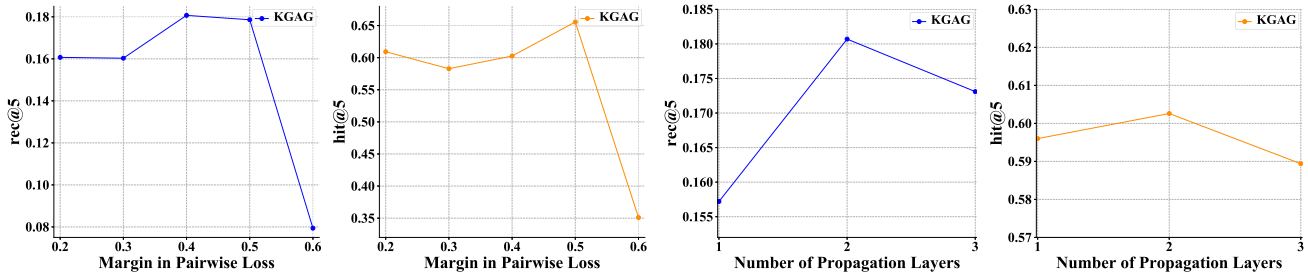

Fig. 4. Influence of the margin in pairwise loss function $\mathcal{M}$ and the number of the propagation layer H.

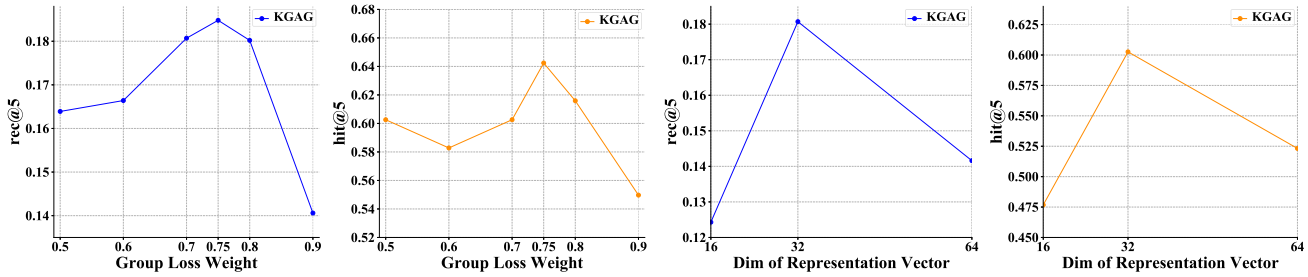

Fig. 5. Influence of group loss weight β and the dimension of representation vector d.

TABLE IV
INFLUENCE OF AGGREGATION FUNCTION.

	MovieLens-20M-Rand		MovieLens-20M-Simi	
	rec@5	*hit@5*	*rec@5*	*hit@5*
GCN	0.1627	0.5497	0.1913	0.7417
GraphSage	0.1589	0.4901	0.1638	0.5960

interaction between $\mathbf{e}$ and $\mathbf{e}_{N_e}$ while GraphSage aggregator does not.

H. Case Study (RQ4)

Figure 6 shows a group on MovieLens-20M-Simi and v_{1085} is recommended to this group with a prediction score 0.8518. As we can see, u_{14514} has the largest influence in this group while u_{18345} has the second large influence while others have little influence. KGAG can give an explanation that this recommendation is made according to u_{14514} and u_{18345}'s preference who have main influence in the group when the candidate item is v_{1085}. Furthermore, more specific explanation can be given according to the values of SP and PI. u_{14514} is consistent in her preference and supported by peers while u_{18345} is supported by peers but not very consistent, so u_{14514} has a larger influence. The example shows a real phenomenon that a few people influence group decision making and others just follow.

V. CONCLUSION

In this paper, we propose an effective end-to-end knowledge graph-based attentive method named KGAG for group recommendation. To overcome the serious limitations of sparse group-item interaction data, we leverage the knowledge graph

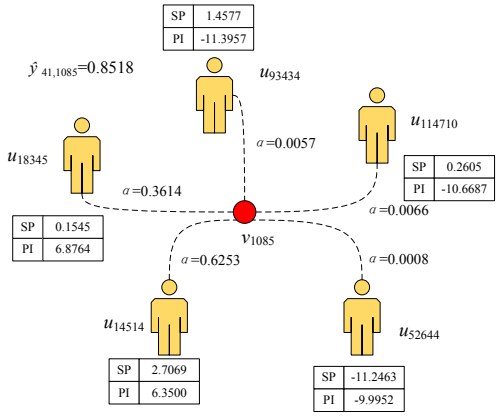

Fig. 6. Real example from MovieLens-20M-Simi. Group g_{41} contains u_{18345}, u_{14514}, u_{52644}, u_{114710} and u_{93434}

as the side information for the group recommendation task. We employ a graph convolution network to capture the abundant structure information of items as well as user-item and user-user connectivity in the graph. In this way, informative knowledge-aware representations of users and items can be learned by propagating the information along the edges in the knowledge graph recursively. Moreover, we consider both candidate item and user-user interaction to learn user influences in group by an attention mechanism self-adaptively, which also provides the interpretability. Finally, a margin loss is introduced to learn more discriminative representation for items and improve the ranking. The comparison and ablation studies show the superiority of KGAG.

REFERENCES

[1] D. Cao, X. He, L. Miao, Y. An, C. Yang, and R. Hong, "Attentive group recommendation," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, 2018, pp. 645–654.

[2] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and W. Cao, "Deep modeling of group preferences for group-based recommendation," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, 2014, pp. 1861–1867.

[3] X. Liu, Y. Tian, M. Ye, and W. Lee, "Exploring personal impact for group recommendation," in *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, 2012, pp. 674–683.

[4] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation and collaborative filtering," in *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, 2010, pp. 119–126.

[5] S. Amer-Yahia, S. B. Roy, A. Chawla, G. Das, and C. Yu, "Group recommendation: Semantics and efficiency," *Proc. VLDB Endow.*, vol. 2, no. 1, pp. 754–765, 2009.

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

[7] H. Wang, J. Wang, M. Zhao, J. Cao, and M. Guo, "Joint topic-semantic-aware social recommendation for online voting," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, 2017, pp. 347–356.

[8] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, 2017, pp. 173–182.

[9] S. Sen, J. Vig, and J. Riedl, "Tagommenders: connecting users to items through tags," in *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*, 2009, pp. 671–680.

[10] Y. Zhen, W. Li, and D. Yeung, "Tagicofi: tag informed collaborative filtering," in *Proceedings of the 2009 ACM Conference on Recommender Systems, RecSys 2009, New York, NY, USA, October 23-25, 2009*, 2009, pp. 69–76.

[11] C. Ziegler, G. Lausen, and L. Schmidt-Thieme, "Taxonomy-driven computation of product recommendations," in *Proceedings of the 2004 ACM CIKM International Conference on Information and Knowledge Management, Washington, DC, USA, November 8-13, 2004*, 2004, pp. 406–415.

[12] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme, "Learning attribute-to-feature mappings for cold-start recommendations," in *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, 2010, pp. 176–185.

[13] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proceedings of the 2007 ACM Conference on Recommender Systems, RecSys 2007, Minneapolis, MN, USA, October 19-20, 2007*, 2007, pp. 17–24.

[14] H. Yin, Q. Wang, K. Zheng, Z. Li, J. Yang, and X. Zhou, "Social influence-based group representation learning for group recommendation," in *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*, 2019, pp. 566–577.

[15] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl, "Polylens: A recommender system for groups of user," in *Proceedings of the Seventh European Conference on Computer Supported Cooperative Work, 16-20 September 2001, Bonn, Germany*, 2001, pp. 199–218.

[16] L. V. Tran, T. N. Pham, Y. Tay, 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, SIGIR 2019, Paris, France, July 21-25, 2019*, 2019, pp. 255–264.

[17] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 353–362.

[18] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, 2015, pp. 2181–2187.

[19] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: deep knowledge-aware network for news recommendation," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, 2018, pp. 1835–1844.

[20] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, 2015, pp. 687–696.

[21] Y. Cao, X. Wang, X. He, Z. Hu, and T. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, 2019, pp. 151–161.

[22] X. Yu, X. Ren, Q. Gu, Y. Sun, and J. Han, "Collaborative filtering with entity similarity regularization in heterogeneous information networks," in *Proceedings of IJCAI-13 HINA Workshop*, 2013.

[23] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," in *Proceedings of the Sixth SIAM International Conference on Data Mining, April 20-22, 2006, Bethesda, MD, USA*, 2006, pp. 549–553.

[24] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, "Ripplenet: Propagating user preferences on the knowledge graph for recommender systems," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, 2018, pp. 417–426.

[25] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, 2019, pp. 3307–3313.

[26] X. Wang, X. He, Y. Cao, M. Liu, and T. Chua, "KGAT: knowledge graph attention network for recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, 2019, pp. 950–958.

[27] L. Guo, J. Shao, K. Tan, and Y. Yang, "Wheretogo: Personalized travel recommendation for individuals and groups," in *IEEE 15th International Conference on Mobile Data Management, MDM 2014, Brisbane, Australia, July 14-18, 2014 - Volume 1*, 2014, pp. 49–58.

[28] Q. Yuan, G. Cong, and C. Lin, "COM: a generative model for group recommendation," in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, 2014, pp. 163–172.

[29] L. Guo, H. Yin, Q. Wang, B. Cui, Z. Huang, and L. Cui, "Group recommendation with latent voting mechanism," in *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020*, 2020, pp. 121–132.

[30] W. Deng, P. Zhu, and J. Ma, "Enhancing group recommendation by knowledge graph," in *22nd Pacific Asia Conference on Information Systems, PACIS 2018, Yokohama, Japan, June 26-30, 2018*, 2018, p. 214.

[31] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[32] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017, pp. 1024–1034.

[33] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: bayesian personalized ranking from implicit feedback," in *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, 2009, pp. 452–461.

[34] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, 2010, pp. 39–46.

[35] Y. Koren, R. M. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer*, vol. 42, no. 8, pp. 30–37, 2009.