# Group Recommendation with Latent Voting Mechanism

Lei Guo[†] Hongzhi Yin[§*] Qinyong Wang[§] Bin Cui[‡] Zi Huang[§] Lizhen Cui[£]
[†]*Shandong Normal University, Jinan, China*
[§]*School of Information Technology and Electrical Engineering, The University of Queensland, Australia*
[‡]*Department of Computer Science & Key Laboratory of High Confidence Software Technologies (MOE), Peking University*
[£]*Shandong University, Jinan, China*
[†]leiguo.cs@gmail.com [§]{h.yin1, qinyong.wang}@uq.edu.au
[‡]bin.cui@pku.edu.cn [§]huang@itee.uq.edu.au [£]clz@sdu.edu.cn

***Abstract*—Group Recommendation (GR) is the task of suggesting relevant items/events for a group of users in online systems, whose major challenge is to aggregate the preferences of group members to infer the decision of a group. Prior group recommendation methods applied predefined static strategies for preference aggregation. However, these static strategies are insufficient to model the complicated decision making process of a group, especially for occasional groups which are formed ad-hoc. Compared to conventional individual recommendation task, GR is rather dynamic and each group member may contribute differently to the final group decision. Recent works argue that group members should have non-uniform weights in forming the decision of a group, and try to utilize a standard attention mechanism to aggregate the preferences of group members, but they do not model the interaction behavior among group members, and the decision making process is largely unexplored.**

**In this work, we study GR in a more general scenario, that is Occasional Group Recommendation (OGR), and focus on solving the preference aggregation problem and the data sparsity issue of group-item interactions. Instead of exploring new heuristic or vanilla attention-based mechanism, we propose a new social self-attention based aggregation strategy by directly modeling the interactions among group members, namely Group Self-Attention (GroupSA). In GroupSA, we treat the group decision making process as multiple voting processes, and develop a stacked social self-attention network to simulate how a group consensus is reached. To overcome the data sparsity issue, we resort to the relatively abundant user-item and user-user interaction data, and enhance the representation of users by two types of aggregation methods. In the training process, we further propose a joint training method to learn the user/item embeddings in the group-item recommendation task and the user-item recommendation task simultaneously. Finally, we conduct extensive experiments on two real-world datasets. The experimental results demonstrate the superiority of our proposed GroupSA method compared to several state-of-the-art methods in terms of HR and NDCG.**

## I. Introduction

Owning to the popularity of Online-to-Offline (O2O) platforms, it becomes convenient and easy for a group of people to find a wide range of suitable activities to attend [1]. For example, a group of friends can work together to plan a trip, find a restaurant or attend a party [2], [3]. The interaction data among groups, group members and their attended activities [4], [5] provides us with a valuable opportunity to analyze their Decision-Making Process (DMP), and based on this, we are motivated to develop group recommender systems.

Depending on whether groups have stable members, user groups can be categorized into persistent groups and occasional groups [6]–[8]. Persistent groups [9]–[11] are predefined static groups with stable members and sufficient historical group-item interactions. When making recommendations, the system views each group as a pseudo user, and thus traditional individual recommendation techniques can be straightforwardly employed. On the other hand, occasional groups [3], [12], [13] refer to ad-hoc formed groups (also known as cold-start groups) whose members might join them for the first time. This kind of group is often established for a temporary event, such as attending a social party or an academic conference. Since occasional group is more general and realistic, we focus on developing recommender systems for them in this work. Compared to recommendations for individual users [14]–[16], there are two types of challenges in designing such tasks: (1) In OGR, only limited group-item interactions are available, which makes directly learning the representations of occasional groups infeasible, and the recommendation methods designed for individual users are inapplicable. (2) The DMP within an occasional group is more complicated, since its process is rather dynamic and each group member may contribute differently to the final decision. More importantly, a member may have various impacts across different groups, and the opinions of all group members should not be equally considered. However, prior studies [12], [17]–[20] only focused on leveraging heuristic and predefined strategies (e.g., average, least misery and maximum pleasure) to aggregate the preferences of group members, which cannot adapt to the dynamic decision making process, thus resulting in sub-optimal recommendation performance. Recently, AGREE [9] and SIGR [6] were proposed to use the vanilla attention mechanism [21] to capture the complicated and dynamic group decision making process. However, we argue that they face two serious drawbacks. Firstly, they ignore the group members' interactions (such as argues and votes) during the DMP, which are usually indispensable and critical to reach the consensus. Secondly, they do not explore the expertise level of each group member that may result in different contribution

[*]Corresponding author and contributing equally with the first author

2375-026X/20/$31.00 ©2020 IEEE
DOI 10.1109/ICDE48307.2020.00018

to a specific activity. For example, a food critic may dominate the choosing process of a restaurant, but may not have any contribution to the choosing of a movie.

To address the aforementioned challenges, we propose a novel group recommendation model called **Group S**elf-**A**ttention (GroupSA). Similar to prior work, the goal of GroupSA is to model the group decision making process, and solves its fundamental issue - preference aggregation, that is, how to aggregate the preferences of group members for a final decision. However, instead of exploring vanilla attention-based or new heuristic strategies, we propose to learn the aggregation strategy from the interactions among group members. Intuitively, before making a group decision, the members often first have a full discussion with each other where more than one potential candidate activities might be proposed, and some (or all) of the members would debate on their pros and cons. Only after the discussion would the members vote for decision makers. As such, the interaction among group members is one of the most crucial steps in achieving a consensus within a group. Moreover, as social animals, users usually appreciate and value the suggestions from their friends. Thus, the group decision making process can be simulated as: (1) each user first exchanges opinions with his/her friends to reach a consensus with them, (2) then votes to nominate certain experts on the current topic as the main decision makers of the group. With the above intuitions, we leverage the self-attention mechanism [21], [22], which has achieved great success in many areas, to solve the preference aggregation problem in OGR, and propose a GroupSA framework to model how a group consensus is reached. That is, to model the interactions among group members, we consider the decision making as $l$ sub-voting processes that happen simultaneously ($l$ is the number of group members), and propose a social self-attention network to simulate how a group decision is made. After that, we represent each group by dynamically aggregating the user latent factors with the vanilla attention [23].

To overcome the sparsity issue of group-item interaction data, we resort to the relatively abundant user-item and user-user interactions, and assume that the users' latent features can be learned from the items and users that they have interacted with. More specifically, to improve the representation of users, we model the user-item and user-user interactions as two graphs. Users are simultaneously involved in both graphs, therefore these two graphs are implicitly linked. Then, two types of aggregation methods, namely item aggregation and social aggregation methods are utilized to learn the latent factors of corresponding users. To accelerate the training of the group recommendation task, we further propose a joint training method to learn the user/item embeddings from user-item, user-user and group-item interaction data simultaneously. In this way, the user-item recommendation task and group-item recommendation task can enhance each other, and eventually lead to better group recommendation performance.

To the best of our knowledge, we are the first to leverage the self-attention mechanism to model the interaction behaviors among group members in the OGR problem. The main contributions of this work are summarized as follows:

- We propose a social self-attention network to model the voting scheme of group members, where the social influence and the dynamic weighting adjustment across groups are jointly considered.
- To alleviate the data sparsity of group-item interactions, we resort to the relatively abundant user-item and user-user interactions, and propose two types of aggregation methods (i.e., item aggregation and social aggregation) to enhance the representation of users.
- To improve the training process of the group-item recommendation task, we propose a joint model optimization method to simultaneously train the group-item and user-item recommendation task by sharing the embeddings of users and items.
- We conduct extensive experiments on two real-world datasets to demonstrate the effectiveness of our method.

## II. Methodologies

### A. Task Definition

We use bold capital letters (e.g., $\boldsymbol{X}$) and bold lowercase letters (e.g., $\boldsymbol{x}$) to represent matrices and vectors, respectively. We employ squiggle letters (e.g., $\mathcal{X}$) to denote sets, and none-bold lowercase or capital letters (e.g., $x$ or $X$) to denote scalars. All vectors are in column forms if not clarified.

Suppose we have $m$ users $\mathcal{U} = \{u_1, u_2, ..., u_j, ..., u_m\}$, $n$ items $\mathcal{V} = \{v_1, v_2, ..., v_h, ..., v_n\}$, and $k$ groups $\mathcal{G} = \{g_1, g_2, ..., g_t, ..., g_k\}$. Each user can establish social relationships with other users who have similar interests with him/her. The $t$-th group $g_t \in \mathcal{G}$ consists of a set of users, and we use $\mathcal{G}(t) = \{u_1, u_2, ..., u_j, ..., u_l\}$ to denote the user set of $g_t$, where $u_j \in \mathcal{U}$, and $l$ is the size of group $g_t$. In a typical group recommendation scenario, there are three kinds of observed interactions among $\mathcal{U}, \mathcal{V}$ and $\mathcal{G}$, that is, user-item interactions, group-item interactions and user-user interactions. We use $\mathcal{R}^U = [r^U_{j,h}]_{m\times n}$ to denote the user-item interactions, $\mathcal{R}^G = [r^G_{t,h}]_{k\times n}$ to denote the group-item interactions, and $\mathcal{R}^S = [r^S_{j,j'}]_{m\times m}$ to denote the user-user interactions (i.e., user social network), where $r \in \{1, 0\}$ denotes whether a group/user has a direct interaction with the corresponding item/user. Then, given the target group $g_t$, our task is to recommend a ranked list of items that group $g_t$ might be interested in, which can be formally defined as:

**Input:** User set $\mathcal{U}$, item set $\mathcal{V}$, group set $\mathcal{G}$, user-item interactions $\mathcal{R}^U$, group-item interactions $\mathcal{R}^G$, and user-user interactions $\mathcal{R}^S$.

**Output:** A personalized ranking function that maps an item to a rating score for a target group $f_t : \mathcal{V} \to \mathbb{R}$.

### B. Overview of GroupSA

Fig. 1 illustrates the architecture of GroupSA, which consists of three components: voting scheme modeling, user modeling and the joint optimization technique. (1) The voting scheme modeling simulates the reaching of the group consensus from $l$ individual sub-voting processes via a self-attention network. Specifically, it first divides each group into

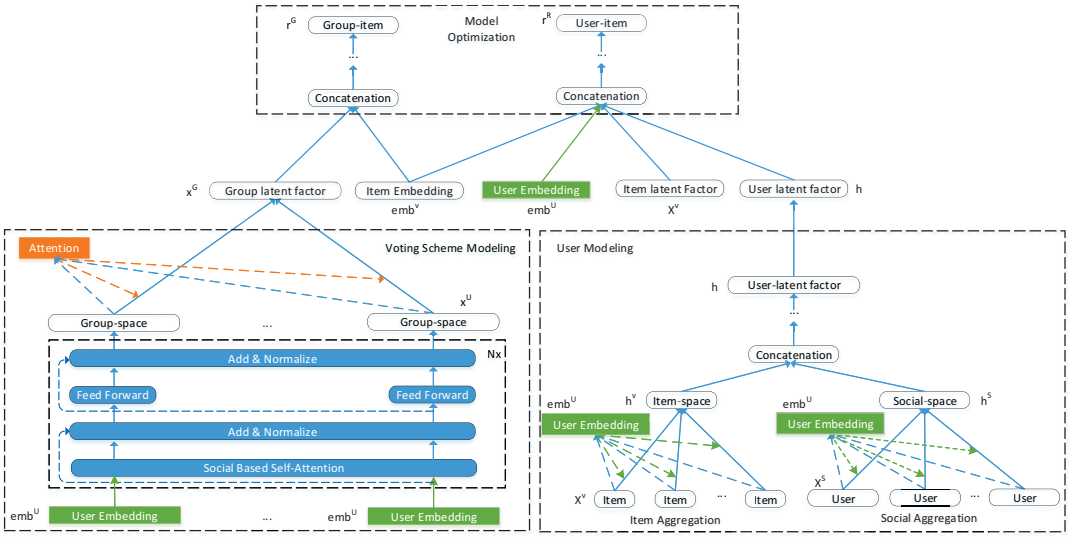


Fig. 1. Illustration of the GroupSA Network.

$l$ *sub-groups*, and then models the member relationship by a self-attention mechanism. In this component, to consider the social influence of users, the self-attention is only applied between members that have direct social connections [24] (details can be seen in Section II-C). (2) The user modeling component enhances the user representation by treating the user-item and user-user interactions as two graphs, and uses two types of aggregation methods to integrate features from item-space and social-space respectively (details can be seen in Section II-D). (3) To improve the training process of the group-item recommendation task, we further propose a joint optimization technique to let the user-item recommendation task and the group-item recommendation task complement and enhance each other (details can be seen in Section II-E).

### C. Voting Scheme Modeling

The goal of the voting scheme modeling is to simulate the decision making process by treating the consensus formation of a group as $l$ sub-voting processes. Each sub-voting process aims to model two types of information for every group member, namely the intrinsic characteristics of the current user and the characteristics of the users in the views of the other group members. The former represents the user's inherent perception to a specific item, and it will remain unchanged during the discussion. The latter denotes how well the current user can represent other group members' interests from their views, which is quite similar to the setting of presidential election. To mimic the process of collecting the views of other group members on the current user, we further define the $i$-th *sub-voting-group*, which excludes the $i$-th group member (the current user) from it. The aim of a *sub-voting-group* is to gather the votes of other group members on user $u_i$. To combine it with the intrinsic characteristics of user $u_i$ and achieve a more comprehensive representation learning, we resort to the self-attention network [21], [25], which is able to jointly learn these two kinds of information by the self-attention mechanism. To achieve this, we model user $u_i$ together with the $i$-th *sub-voting-group*, thus forming a *sub-group* for $u_i$. Note that, there are two types of *sub-groups*, i.e., *sub-voting-group* and *sub-group*. The difference between the $i$-th *sub-group* and the $i$-th *sub-voting-group* is that the $i$-th *sub-voting-group* does not include user $u_i$, while the $i$-th *sub-group* does. Moreover, because users may change their minds after seeing the voting results of others, we exploit a stacked self-attention network with multiple identical layers (the number of layers is denoted as $N_X$) to simulate the multiple rounds of voting processes.

Formally, for a given group $g_t$ with $l$ members, to learn the reviews of other group members, we first divide $g_t$ into $l$ *sub-voting-groups*. Then, to capture these two types of characteristics for every group member, we transform the $i$-th *sub-voting-group* plus user $u_i$ (i.e., the $i$-th *sub-group*) from the $(k-1)$-th layer[1] ($\boldsymbol{X}^{k-1} \in \mathbb{R}^{l \times d_{model}}$) into queries $\boldsymbol{Q} = \boldsymbol{X}^{k-1}\boldsymbol{W}^Q$, keys $\boldsymbol{K} = \boldsymbol{X}^{k-1}\boldsymbol{W}^K$ and values $\boldsymbol{V} = \boldsymbol{X}^{k-1}\boldsymbol{W}^V$ through linear projections, where $\boldsymbol{W}^Q \in \mathbb{R}^{d_{model} \times d_k}$, $\boldsymbol{W}^K \in \mathbb{R}^{d_{model} \times d_k}$, $\boldsymbol{W}^V \in \mathbb{R}^{d_{model} \times d_v}$ are the projection matrices. Then, the attention weight of the interaction between member $u_i$ and an arbitrary user $u_j$ in this *sub-group* can be defined as follows:

$$ATT^*(g_{t,i}, u_i, u_j) = \frac{\boldsymbol{q}_i \boldsymbol{k}_j^T}{\sqrt{d_k}} \tag{1}$$

$$ATT(g_{t,i}, u_i, u_j) = \frac{exp(ATT^*(g_{t,i}, u_i, u_j))}{\sum_{j=1}^{l} exp(ATT^*(g_{t,i}, u_i, u_j))} \tag{2}$$

where $\boldsymbol{q}_i$ is the query of user $u_i$, $\boldsymbol{k}_j$ is the key of user $u_j$ and $\sqrt{d_k}$ is the scaling factor with $d_k$ being the dimension of queries and keys. $ATT(g_{t,i}, u_i, u_j)$ is the dot-product attention [21] between user $u_i$ and $u_j$ in the $i$-th *sub-group* $g_{t,i}$, and

[1]The input of the first layer is the user embedding $\boldsymbol{emb}^U$.

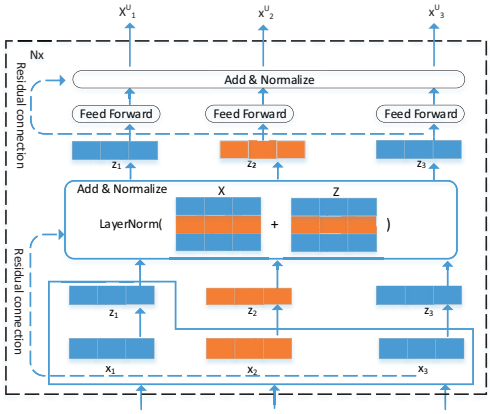


Fig. 2. The Voting Scheme of GroupSA. We group $x_1$-$x_3$ and $z_1$ by a blue rectangle, because $x_1$-$x_3$ belong to the same *sub-group*, and the result is $z_1$. We mark $x_1$, $x_3$ and $z_1$ as blue, because $u_3$ has a social connection with $u_1$. As a result, the self-attention will only be applied between $u_1$ and $u_3$.

the more dependent $u_i$ is on $u_j$, the higher the attention weight will be. The dot-product attention between $q_i$ and $k_i$ indicates how much user $u_i$ insists on her/his own opinions when discussing with other users. Then, the output of each sub-attention network $\boldsymbol{z}_i$ that denotes the representation of a *sub-group* under the view of user $u_i$ is further defined as:

$$\boldsymbol{z}_i = \sum_{j=1}^{l} ATT(g_{t,i}, u_i, u_j)\boldsymbol{v}_j \tag{3}$$

where $\boldsymbol{v}_j$ is the value of user $u_j$.

Although the above self-attention mechanism can model the users' global dependencies by directly attending to all users in a given group, it cannot model the social dependencies among users. In reality, as we often turn to friends for recommendations, the interactions with friends are also the key factors that affect our decisions. We argue that the voting process can be further improved by taking into account the user's social interactions. **To simulate the social voting process**, we consider to place a social bias matrix $\boldsymbol{S}$ to mask the dot-product attention $ATT^*$ (denoted in Eq. (1)). Then, the attention model can be modified as:

$$ATT^*(g_{t,i}, u_i, u_j) = \frac{\boldsymbol{q}_i \boldsymbol{k}_j^T}{\sqrt{d_k}} + S_{i,j} \tag{4}$$

The $softmax$ function (as shown in Eq. (2)) is also applied to Eq. (4). The first term of Eq. (4) is the original self-attention model. $\boldsymbol{S} \in \{-\infty, 0\}^{l \times l}$ is the social bias matrix, whose elements control whether the attention scheme will be applied to the user pair $(u_i, u_j)$:

$$S_{i,j} = \begin{cases} -\infty, & f(i,j) = 0 \\ 0, & otherwise \end{cases} \tag{5}$$

where $f(i,j) \in [0,1]$ measures the closeness between user $u_i$ and $u_j$ in the social network. $f(i,j)$ can be computed by any real-valued score function (such as PageRank, closeness and betweeness). In experiments, we simply let $f(i,j) = 1$ for a direct social connection between $u_i$ and $u_j$, and $f(i,j) = 0$ otherwise. In this attention model, the calculation of the self-attention between $u_i$ and $u_j$ is disabled for $f(i,j) = 0$ by setting $S_{i,j} = -\infty$ and enabled by setting $S_{i,j} = 0$ for $f(i,j) = 1$. Note that due to the exponential operation in $softmax$ function (as shown in Eq. (2)), adding the dot-product attention with a bias of $-\infty$ or 0 approximates to multiplying the attention distribution by a weight of 0 or 1.

Fig. 2 illustrates the architecture of our voting scheme, which contains a stacked self-attention network with two sub-layers. The first layer is the social self-attention network, which takes the embedding of each group member as the input[2], and outputs the representations of every *sub-group* under the view of different users (the output of the $i$-th *sub-group* is denoted as $\boldsymbol{z}_i$). The second layer is a fully connected Feed-Forward Network (FFN) [21] with two linear transformations:

$$FFN(\boldsymbol{z}) = \sigma(\boldsymbol{z}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2 \tag{6}$$

where $\boldsymbol{b}$ and $\boldsymbol{W}$ are the bias and weight of a neural network, while $\sigma(\cdot)$ denotes the non-linear activation function (we use rectified linear unit in this paper). We employ a residual connection [21] to each of the two sub-layers, followed by layer normalization. That is, the output of each sub-layer is $LayerNorm(\boldsymbol{x} + Sublayer(\boldsymbol{x}))$, where $Sublayer(\boldsymbol{x})$ is the FFN or social self-attention described above.

**To simulate the dynamic preference aggregation process**, as shown in Fig. 1, the representations of all *sub-groups* of group $g_t$ are further aggregated by a vanilla attention network:

$$\boldsymbol{x}_t^G = \sigma(\boldsymbol{W} \cdot Aggre_{group}(\{\boldsymbol{x}_{t,i}^U, \forall u_i \in \mathcal{G}(t)\}) + \boldsymbol{b}) \tag{7}$$

where $\boldsymbol{x}_{t,i}^U$ is the representation of the $i$-th *sub-group* (also denotes the preference of the $i$-th group member), $\boldsymbol{x}_t^G$ is the representation of the $t$-th group $g_t$. $Aggre_{group}(\cdot)$ is the preference aggregation function of a group, which determines the way of aggregating the representations of group members:

$$Aggre_{group}(\{\boldsymbol{x}_{t,i}^U, \forall u_i \in \mathcal{G}(t)\}) = \sum_{u_i \in \mathcal{G}(t)} \gamma_{t,i} \boldsymbol{x}_{t,i}^U, \tag{8}$$

where $\gamma_{t,i}$ implies the importance of the $i$-the group member. As the expertise level of each group member may vary for different given topics, we parameterize $\gamma_{t,i}$ with a two-layer neural network:

$$\gamma_{t,i}^* = \boldsymbol{w}_2^T \cdot \sigma(\boldsymbol{W}_1 \cdot [\boldsymbol{emb}_h^V \oplus \boldsymbol{x}_{t,i}^U] + \boldsymbol{b}_1) + \boldsymbol{b}_2 \tag{9}$$

$$\gamma_{t,i} = \frac{exp(\gamma_{t,i}^*)}{\sum_{u_i \in \mathcal{G}(t)} exp(\gamma_{t,i}^*)}, \tag{10}$$

where $\oplus$ is the concatenation operator of two vectors, $\boldsymbol{emb}_h^V$ is the embedding of the target item. The final attention weights are normalized by the $softmax$ function (shown in Eq. (10)).

[2]In the first layer, the input of the $j$-th user is denoted as $\boldsymbol{emb}_j^U$, and in other layers it is denoted as $\boldsymbol{x}_j$.

### D. User Modeling

The task of user modeling aims to enhance the user representation by leveraging the relatively abundant user-item and user-user interaction data. Inspired by [26], [27], we first treat these two kinds of data as user-item graph and user-user graph respectively (users are involved in both graphs), and then use two types of aggregation methods to learn the user latent factor from item-space and social-space respectively (as shown in Fig. 1). Finally, we combine these two aggregated representations into the final user latent factor $\boldsymbol{h}_j \in \mathbb{R}^d$. The learned user latent feature factor is different from the user embedding learned from the user-item space, and thus should be treated differently.

We first perform **item aggregation** to learn the user latent factor $\boldsymbol{h}_j^V$ by aggregating the information of items that user $u_j$ has interacted with. Mathematically, the item aggregation is defined as follows:

$$\boldsymbol{h}_j^V = \sigma(\boldsymbol{W} \cdot Aggre_{items}(\{\boldsymbol{x}_h^V, \forall v_h \in \mathcal{C}(j)\}) + \boldsymbol{b}) \quad (11)$$

where $\boldsymbol{x}_h^V$ is the latent factor of item $v_h$ in item-space, $\mathcal{C}(j)$ is the set of items that user $u_j$ has interacted with. In practice, we rank the items according to TF-IDF [28], and select Top-$H$ of them to represent the specific user ($H$ is the number of items that we aggregated). $Aggre_{items}(\cdot)$ is the item aggregation function for aggregating related items:

$$Aggre_{items}(\{\boldsymbol{x}_h^V, \forall v_h \in \mathcal{C}(j)\}) = \sum_{v_h \in \mathcal{C}(j)} \alpha_{j,h} \boldsymbol{x}_h^V, \quad (12)$$

where $\alpha_{j,h}$ is the corresponding weight assigned to item $v_h$. Instead of using a unified weight for all items, we point out that the influence of interacted items may vary dramatically, and different item contexts should contribute differently to a user's latent factor. Inspired by the vanilla attention mechanism [23], for a given user $u_j$, we compute an exclusive weight for each $(u_j, v_h)$ pair, and parameterize $\alpha_{j,h}$ with the following attention network:

$$\alpha_{j,h}^* = \boldsymbol{w}_2^T \cdot \sigma(\boldsymbol{W}_1 \cdot [\boldsymbol{emb}_j^U \oplus \boldsymbol{x}_h^V] + \boldsymbol{b}_1) + \boldsymbol{b}_2 \quad (13)$$

$$\alpha_{j,h} = \frac{exp(\alpha_{j,h}^*)}{\sum_{v_h \in \mathcal{C}(j)} exp(\alpha_{j,h}^*)} \quad (14)$$

where $\boldsymbol{emb}_j^U$ is the embedding of the target user $u_j$ from the user-item space, which guides the item aggregation process. Eq. (14) performs the $softmax$ normalization for all computed weights.

We then introduce **social aggregation** to model users' social influence based on the social correlation theories [29], [30], that is, a user's preference is affected by his/her directly connected friends. Moreover, as a user's social influence may vary in different social relationships, the learning of the user latent factors in social-space should account for the heterogeneous nature of social connections. Thus, for each user, we further utilize a vanilla attention network to aggregate information from his/her friends that can most characterize this user's social information. Specifically, for user $u_j$, we aggregate the latent factors of users in social-space to learn the user's social interest $\boldsymbol{h}_j^S$ as follows:

$$\boldsymbol{h}_j^S = \sigma(\boldsymbol{W} \cdot Aggre_{social}(\{\boldsymbol{x}_{j'}^S, \forall u_{j'} \in \mathcal{N}(j)\}) + \boldsymbol{b}) \quad (15)$$

where $\boldsymbol{x}_{j'}^S$ is the latent factor of user $u_{j'}$ in social-space. $\mathcal{N}(j)$ is user $u_j$'s neighbors that have direct social connection with him/her. Similar to item aggregation, the TF-IDF based ranking score is applied, and only Top-$H$ users are considered. $Aggre_{social}(\cdot)$ is the aggregation function that denotes how we aggregate the user's neighbors:

$$Aggre_{social} = \sum_{u_{j'} \in \mathcal{N}(j)} \beta_{j,j'} \boldsymbol{x}_{j'}^S, \quad (16)$$

where $\beta_{j,j'}$ is the aggregation weight that determines the social strengths of different social relations. To distinguish the social impact of different users, we also parameterize $\beta_{j,j'}$ with a two-layer attention network:

$$\beta_{j,j'}^* = \boldsymbol{w}_2^T \cdot \sigma(\boldsymbol{W}_1 \cdot [\boldsymbol{emb}_j^U \oplus \boldsymbol{x}_{j'}^S] + \boldsymbol{b}_1) + \boldsymbol{b}_2 \quad (17)$$

$$\beta_{j,j'} = \frac{exp(\beta_{j,j'}^*)}{\sum_{u_{j'} \in \mathcal{N}(j)} exp(\beta_{j,j'}^*)} \quad (18)$$

Same as Eq. (13), $\boldsymbol{emb}_j^U$ is the attention signal from user-item space, which determines how we aggregate related users.

**The final user latent factor.** Since the social graph and user-item graph provide different semantic information of users, we further exploit a standard Multi-Layer Perception (MLP) to combine these two factors to learn the final user latent factor, where the item-space user latent factor $\boldsymbol{h}_j^V$ and the social-space user latent factor $\boldsymbol{h}_j^S$ are concatenated as the input. Formally, the user latent factor $\boldsymbol{h}_j$ of user $u_j$ is computed via:

$$\begin{aligned} &\boldsymbol{c}_1 = [\boldsymbol{h}_j^V \oplus \boldsymbol{h}_j^S] \\ &\boldsymbol{c}_2 = \sigma(\boldsymbol{W}_2 \cdot \boldsymbol{c}_1 + \boldsymbol{b}_2) \\ &... \\ &\boldsymbol{h}_j = \sigma(\boldsymbol{W}_k \cdot \boldsymbol{c}_{k-1} + \boldsymbol{b}_k) \end{aligned} \quad (19)$$

where $k$ is the index of a hidden layer, $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weight and bias of a feed-forward neural network.

### E. Model Optimization

As shown in Fig. 1, given the embeddings of the target group and item (i.e., $\boldsymbol{x}_t^G$ and $\boldsymbol{emb}_h^V$), we feed their concatenation into a MLP for ranking score prediction:

$$\begin{aligned} &\boldsymbol{c}_1^G = [\boldsymbol{x}_t^G \oplus \boldsymbol{emb}_h^V] \\ &\boldsymbol{c}_2^G = \sigma(\boldsymbol{W}_2 \cdot \boldsymbol{c}_1^G + \boldsymbol{b}_2) \\ &... \\ &\hat{r}_{t,h}^G = \boldsymbol{w}^T \cdot \boldsymbol{c}_{k-1}^G \end{aligned} \quad (20)$$

where $k$ is the index of a hidden layer, $\boldsymbol{emb}_h^V$ is the item embedding in group-item space and $\hat{r}_{t,h}^G$ is the predicted ranking score from group $g_t$ to item $v_h$.

Due to the implicit nature of the group-item interaction data[3], we treat the group recommendation task as a pair-wise

[3]Only positive interactions can be observed, and the unobserved data is a mixture of negative and potential positive data.

ranking problem, and optimize the following pair-wise loss function [31] :

$$L_G = \arg\min_{\Theta} \sum_{(g_t, v_h, v_{h'}) \in \mathcal{D}_G} -\ln\sigma(\hat{r}^G_{t,h} - \hat{r}^G_{t,h'}) + \lambda||\Theta||^2 \quad (21)$$

where $\Theta$ represents the set of the model parameters. $v_h$ is the observed positive item in $\mathcal{R}^G$, and $v_{h'}$ is the negative item that is unobserved in $\mathcal{R}^G$. We use $N$ to denote the number of randomly sampled negative items per positive example ($v_h$ in this case). In each triple $(g_t, v_h, v_{h'}) \in \mathcal{D}_G$, the group $g_t$ is assumed to prefer $v_h$ over $v_{h'}$, so $\hat{r}^G_{t,h}$ is expected to be greater than $\hat{r}^G_{t,h'}$. The above objective function tries to optimize the parameters by maximizing the margin between the observed and unobserved samples. However, for occasional groups, the observable group-item interactions are extremely sparse, so the learned group and item representations are hardly accurate or reliable.

To mitigate this data sparsity issue, we resort to the user-item interaction data by jointly learning the group-item recommendation task and user-item recommendation task simultaneously. For the user-item recommendation, we use a similar neural network to learn the parameters. That is, given the representations of the target user and item (i.e., $\boldsymbol{emb}^U_j$ and $\boldsymbol{emb}^V_h$), we feed them into a MLP to calculate a ranking score:

$$\begin{aligned} \boldsymbol{c}^U_1 &= [\boldsymbol{emb}^U_j \oplus \boldsymbol{emb}^V_h] \\ \boldsymbol{c}^U_2 &= \sigma(\boldsymbol{W}_2 \cdot \boldsymbol{c}^U_1 + \boldsymbol{b}_2) \\ &... \\ \hat{r}^{R_1}_{j,h} &= \boldsymbol{w}^T \cdot \boldsymbol{c}^U_{k-1} \end{aligned} \quad (22)$$

where $\hat{r}^{R_1}_{j,h}$ is the predicted rating score from user $u_j$ to item $v_h$. $\boldsymbol{emb}^U_j$ and $\boldsymbol{emb}^V_h$ are the shared user and item embeddings, which can bridge the group-item space and user-item space.

To leverage the learned latent factors (i.e., $\boldsymbol{h}_j$ and $\boldsymbol{x}^V_h$) to enhance the recommendation process, we feed the concatenation of $\boldsymbol{h}_j$ and $\boldsymbol{x}^V_h$ into the same MLP network (as shown in Eq. (22)), and compute the final ranking score:

$$\hat{r}^R_{j,h} = (1 - w^u)\hat{r}^{R_1}_{j,h} + w^u \hat{r}^{R_2}_{j,h} \quad (23)$$

where $\hat{r}^{R_2}_{j,h}$ is the user rating score learned from the item-space and social-space. $w^u$ is a hyper-parameter that balances the weights of two components. Due to the implicit feedback of user-item interaction data, we also treat the user-item recommendation as a pair-wise ranking problem, and optimize the following loss function:

$$L_R = \arg\min_{\Theta} \sum_{(u_j, v_h, v_{h'}) \in \mathcal{D}_R} -\ln\sigma(\hat{r}^R_{j,h} - \hat{r}^R_{j,h'}) + \lambda||\Theta||^2 \quad (24)$$

where $(u_j, v_h, v_{h'}) \in \mathcal{D}_R$ is the set of user-item preference samples. Similar to Eq. (21), we use $N$ to denote the number of randomly sampled negative items.

**Training Method.** To jointly optimize $L_G$ and $L_R$ on the heterogeneous interactions, we utilize a two-stage training method. We first optimize the loss function $L_R$ by utilizing the user-item and user-user interaction data to learn the embeddings of users and items, and then use the learned embeddings to initialize the user and item embeddings in the group recommendation task. Then, the parameters for the group recommendation task will be fine-tuned by utilizing the group-item interactions. In our training process, the Stochastic Gradient Descent (SGD) algorithm is adopted. At each gradient step, we randomly sample a positive user-item example $(u_j, v_h)$ (or group-item example $(g_t, v_h)$) and $N$ negative examples $(u_j, v_{h'})$ (or $(g_t, v_{h'})$) to update the model parameters.

### F. Fast Group Recommendation Using GroupSA

Besides the standard group recommendation case, GroupSA can also be applied to the scenarios where only the individual preferences of group members are available. This is important for generating recommendations for large groups, because it can be time-consuming to infer the voting results with a multi-layered neural network, and we need a trade-off between efficiency and effectiveness. Furthermore, as the user embedding in our model is learned by self-attention and user aggregation, it not only considers each user's own intrinsic characteristics, but also carries the interests of other group members, which can help yield comparable results. To illustrate how to make fast group recommendations based on the individual embeddings, we take the typical average strategy as an example. Specifically, for a given occasional group $g_t$, we first obtain the embedding ($\boldsymbol{emb}^U_j$) of each group member and the embedding ($\boldsymbol{emb}^V_h$) of each candidate item by a lookup operation, and calculate the ranking score between every group member and the target item by Eq. (23). Then, we average the scores of all the group members as the group's preference. Finally, the Top-$K$ items with highest ranking scores will be recommended.

## III. Experimental Setup

### A. Research Questions

Our proposals are fully validated by answering the following five research questions.

- **RQ1** How does our proposed GroupSA approach perform compared with state-of-the-art group recommender systems in terms of accuracy?
- **RQ2** What is the effectiveness of our designed social self-attention network?
  Can it improve the group recommendation?
- **RQ3** How does the item aggregation component of user modeling contribute to the performance of group recommendation?
  How does the social aggregation component of user modeling contribute to the performance of group recommendation?
- **RQ4** Can we improve the group recommendation by integrating the user-item interactions?
- **RQ5** How do the hyper-parameters affect the performance of GroupSA?

TABLE I
STATISTICS OF YELP AND DOUBAN-EVENT.

| Statistics | Yelp | Douban-Event |
|---|---|---|
| # Users | 34,504 | 29,181 |
| # Items/Events | 22,611 | 46,097 |
| # Groups | 24,103 | 17,826 |
| Avg. group size | 4.45 | 4.84 |
| Avg. # interactions per user | 13.98 | 25.22 |
| Avg. # friends per user | 20.77 | 40.86 |
| Avg. # interactions per group | 1.12 | 1.47 |

### B. Datasets

We utilize two large-scale real-world datasets[4] Yelp and Douban-Event as the data source of our experiments. Yelp[5] is an online review website that allows users to share their reviews and check-ins about local businesses. Each user can also create social connections with others to express their social interests. Douban-Event[6] is one of the largest online event-based social networks in China that helps people publish and participate in social events.

As the raw data of Yelp and Douban-Event do not contain explicit group information, [6] extracted the implicit group activities by assuming that if a set of users who are connected on the social network attend the same event (or visit the same restaurant) at the same time, they are members of a group and the corresponding activities are group activities. The resulting datasets not only contain the user-item interactions and user's social network, but also the group-item interactions and the members of each group. The detailed statistics of these two datasets are shown in Table I.

### C. Evaluation Protocols

We randomly select 80% of the group-item and user-item interactions for training, and the remaining are used for testing. In the training dataset, we randomly choose 10% records as the validation set to determine the optimal parameters. Since it is time consuming to rank all items for each user and group, we follow the scheme used in [32] and [9] as our evaluation strategy. That is, we randomly select 100 items that have never been interacted by the tested user or group as the candidate set, and test the ability of all comparison models to rank them. To measure the performance of the Top-$K$ recommendation, we exploit two widely used evaluation metrics Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) as the evaluation metrics [33], where HR@$K$ measures whether the candidate item is ranked within the Top-$K$ list, while NDCG@$K$ accounts for the position of the hit by assigning higher score to hit at Top-$K$ positions.

### D. Baseline Methods

To evaluate the effectiveness of our proposed model GroupSA, we compare it with the following methods.

- **NCF [32].** This is a state-of-the-art item recommendation method developed for individual users. In this method, we treat a group as a virtual user, and ignore the member information of the group. This is to validate whether the traditional Collaborative Filtering (CF) method can be directly applied to the OGR task.
- **Pop [34].** This is a non-personalized recommendation method that recommends items to users (or groups) according to the item popularity, which is computed by the interacted number of items in the training set.
- **AGREE [9].** This is a state-of-the-art OGR method that models the group-item interactions under the Neural Collaborative Filtering (NCF) framework, and adopts a vanilla attention mechanism to learn the weight of each user in a group.
- **SIGR [6].** This is another state-of-the-art OGR method which integrates a vanilla attention mechanism with the bipartite graph embedding technique for group recommendation, and develops a deep social influence learning framework to exploit both global and local social network structure features.

Although COM [13] and PIT [3] are also two state-of-the-art group recommendation methods, we do not compare with them as AGREE [9] and SIGR [6] have shown superior performance over them on all their datasets.

To evaluate our self-attention based aggregation strategy, we further compare it with another category of methods that apply predefined static score aggregation methods to make group recommendations. Specifically, in these methods, we first run GroupSA to predict each member's personal preferences, and then apply the following static aggregation strategies to generate group recommendation results.

- **Group+avg [12].** This method assumes that each member has equal contribution to the final group decision, and simply averages the group member's preference scores as the group preference score.
- **Group+lm [17].** This method assumes that the least satisfied member determines the final group decision, and uses the minimum score of individuals as the group preference score.
- **Group+ms [12].** This method tires to maximize the satisfaction of group members and adopts the maximum score as the group preference score by assuming that a group member prefers to follow other members' opinions.

### E. Implementation Details

We implement GroupSA using the Pytorch deep learning framework accelerated by NVidia GTX 1080 Ti GPU. For the initialization strategy, we follow the work in [9] to apply the Glorot initialization method [35] on embedding layer, and the Gaussian distribution with a mean of 0 and a standard deviation of 0.1 on hidden layers. We use the Adam optimizer for all gradient-based methods, and set the mini-batch size as 256. In GroupSA, the number of negative examples per positive sample and the number of items (or users) utilized in the item aggregation (or social aggregation) are searched

[4]https://sites.google.com/site/dbhongzhi/
[5]www.yelp.com
[6]www.douban.com/location/world/

TABLE II
TOP-K RECOMMENDATION PERFORMANCE ON YELP ($\Delta$ DENOTES THE IMPROVEMENT OF GROUPSA OVER OTHER METHODS IN HR@$K$).

| Overall Performance Comparison (Yelp) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | K=5 | | | | | | K=10 | | | | | |
| | User | | | Group | | | User | | | Group | | |
| | HR | NDCG | $\Delta$% | HR | NDCG | $\Delta$% | HR | NDCG | $\Delta$% | HR | NDCG | $\Delta$% |
| NCF | 0.4914 | 0.3457 | 14.79 | 0.2869 | 0.2028 | 190.65 | 0.6536 | 0.3983 | 6.93 | 0.3911 | 0.2364 | 136.69 |
| Pop | 0.4904 | 0.3460 | 15.02 | 0.3097 | 0.2164 | 169.26 | 0.6502 | 0.3978 | 7.49 | 0.4130 | 0.2498 | 124.14 |
| AGREE | 0.4934 | 0.3479 | 14.32 | 0.3411 | 0.2331 | 144.47 | 0.6540 | 0.4000 | 6.86 | 0.4965 | 0.2829 | 86.44 |
| SIGR | 0.5191 | 0.3802 | 0.89 | 0.4095 | 0.3009 | 103.63 | 0.6608 | 0.4213 | 1.17 | 0.5343 | 0.3411 | 73.25 |
| Group+avg | – | – | – | 0.6423 | 0.4674 | 29.83 | – | – | – | 0.8003 | 0.5188 | 15.66 |
| Group+lm | – | – | – | 0.6141 | 0.4483 | 35.79 | – | – | – | 0.7598 | 0.4955 | 21.83 |
| Group+ms | – | – | – | 0.5806 | 0.4183 | 43.62 | – | – | – | 0.7495 | 0.4730 | 23.50 |
| GroupSA | **0.5641** | **0.4131** | – | **0.8339** | **0.6886** | – | **0.6989** | **0.4568** | – | **0.9257** | **0.7186** | – |

TABLE III
TOP-K RECOMMENDATION PERFORMANCE ON DOUBAN-EVENT ($\Delta$ DENOTES THE IMPROVEMENT OF GROUPSA OVER OTHER METHODS IN HR@$K$).

| Overall Performance Comparison (Douban-Event) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | K=5 | | | | | | K=10 | | | | | |
| | User | | | Group | | | User | | | Group | | |
| | HR | NDCG | $\Delta$% | HR | NDCG | $\Delta$% | HR | NDCG | $\Delta$% | HR | NDCG | $\Delta$% |
| NCF | 0.3949 | 0.2747 | 58.54 | 0.2246 | 0.1504 | 205.83 | 0.5553 | 0.3264 | 34.82 | 0.3261 | 0.1832 | 151.70 |
| Popularity | 0.3851 | 0.2666 | 62.58 | 0.2198 | 0.1452 | 212.51 | 0.5426 | 0.3173 | 37.98 | 0.3153 | 0.1791 | 160.32 |
| AGREE | 0.3945 | 0.2719 | 58.70 | 0.2810 | 0.1832 | 144.44 | 0.5582 | 0.3247 | 34.12 | 0.4280 | 0.2301 | 91.77 |
| SIGR | 0.5388 | 0.4066 | 1.17 | 0.3412 | 0.2547 | 101.31 | 0.6422 | 0.4282 | 2.25 | 0.4739 | 0.3052 | 73.20 |
| Group+avg | – | – | – | 0.6058 | 0.4415 | 13.38 | – | – | – | 0.7476 | 0.4877 | 9.79 |
| Group+lm | – | – | – | 0.5485 | 0.4023 | 25.23 | – | – | – | 0.6861 | 0.4470 | 19.63 |
| Group+ms | – | – | – | 0.5633 | 0.4068 | 21.94 | – | – | – | 0.7209 | 0.4580 | 13.85 |
| GroupSA | **0.6261** | **0.4762** | – | **0.6869** | **0.5239** | – | **0.7487** | **0.5160** | – | **0.8208** | **0.5675** | – |

within [1, 2, 3, 4, 5] and [2, 3, 4, 5, 6], respectively. The dimensions of the embeddings of user, group and item are all set as 32. For the self-attention component, the layer number of the social self-attention is searched within [1, 2, 3, 4, 5]. The dimension ($d_{model}$) of the hidden layers in the FFN and the dimensions ($d_v, d_k$) of the queries, keys and values in the point-wise attention network are both set as 32. For parameter $w^u$ that determines how much we depend on the latent factors learned from the item-space and social-space is searched within [0.1, 0.3, 0.5, 0.7, 0.9, 1.0]. To prevent from over-fitting, we use the dropout regularization with dropout ratio 0.1 for both datasets. For all the hyper-parameters, we tune them on the validation set, and repeat each setting 5 times and report the average results. More details on tuning the hyper-parameters are shown in Section V-C. We conduct the one sample paired t-tests to verify that all improvements are statistically significant with $p < 0.01$.

## IV. EXPERIMENTAL RESULTS (RQ1)

Tables II and III show the experimental results on Yelp and Douban-Event in terms of HR@$K$ and NDCG@$K$. Note that since score aggregation methods (i.e., Group+avg, Group+lm, Group+ms) are specially developed for group recommendation, they are not applicable for individual user recommendation. From the results, we have the following observations: 1) Our GroupSA method achieves the best performance on the two datasets for both recommendation tasks, in particular, the improvements over the compared state-of-the-art methods are statistically significant with $p < 0.01$. This demonstrates the positive effect of the social self-attention network in aggregating the preferences of group members, and the joint training method in overcoming the sparsity issue of the group-item interaction data. 2) Predefined aggregation strategies (i.e., Group+avg, Group+lm and Group+ms) can work better than other state-of-the-art group recommendation methods (AGREE and SIGR), demonstrating that we can learn more accurate user and item representations from GroupSA. The result that GroupSA performs better than these predefined static strategies demonstrates the importance of learning varying weights for group members, and simply aggregating the preferences of group members to make group recommendation can only get sub-optimal results. 3) On both datasets, score aggregation-based strategies (i.e., Group+avg, Group+lm and Group+ms) have similar recommendation performance, and none of them performs better than GroupSA. Although Group+avg achieves the best performance among these three methods, the experimental results are unstable. For example, Group+lm achieves better performance than Group+mp on Yelp dataset, but performs worse than Group+mp on Douban-Event dataset. This result, again, demonstrates the ineffectiveness of static score aggregation-based methods compared to the methods that can learn the aggregation weights dynamically.

Although our method is only tested on Yelp and Douban-Event, it can also be applied to other real-world settings. For example, when users attend an academic conference abroad, they may want to plan a trip with other attendees after the conference ends. As the group is formed occasionally, we do not have many historical group activities. The recommenda-

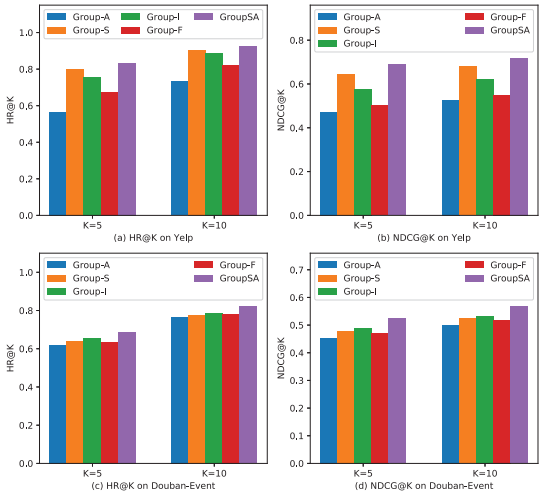


Fig. 3. Importance of Social Self-Attention and User Modeling Components.

tion problem falls into the occasional group recommendation scenario, where GroupSA can be applied. Similarly, we can also help users who occasionally meet at a social event to find a restaurant. In the above cases, both groups are formed occasionally and users have social connections with each other.

## V. Model Analysis

### A. Importance of Components (RQ2 & RQ3)

To understand the importance of social self-attention and user modeling (i.e, item aggregation and social aggregation), we compare GroupSA with its four variants:

- **Group-A** This is a simplified version of GroupSA that removes the voting scheme and user modelling component from it, and only uses a vanilla attention network for preference aggregation. This is to verify the effectiveness of these two components.
- **Group-S** This variant removes the social self-attention network from it, and only uses a vanilla attention network to conduct preference aggregation. This is to demonstrate the importance of our social self-attention mechanism in forming the group's final decision.
- **Group-I** This is another variant of GroupSA that removes the item aggregation component of user modeling, and only utilizes the social aggregation component to learn the representation of users. This is to validate the usefulness of our item aggregation component.
- **Group-F** This variant removes the social aggregation component of user modeling from GroupSA, and only utilizes the item aggregation component to learn user's representation. This is to validate the importance of our social aggregation component.

*1) Importance of Social Self-Attention (RQ2):* To validate the effectiveness of the social self-attention network in the group recommendation task, we compare GroupSA with Group-A and Group-S. The experimental results are shown in Figure 3. We have the following observations: 1) GroupSA significantly outperforms Group-A and Group-S ($p < 0.01$) on both Yelp and Douban-Event, showing that both social self-attention and user modeling components are beneficial to group decision making, and only utilizing one of them alone cannot get better results than combining them together. 2) GroupSA achieves better performance than Group-S on both datasets, revealing the importance of the social self-attention network in improving the performance of group recommendation. Apart from the macro-level analysis, we further conduct a qualitative analysis of the model by case studies. More specifically, we randomly select a testing group, which consists of three users (#101, #102 and #103), from the Yelp dataset. The group has visited two restaurants/items (#50 and #82) with the target value of 1. Each group member has her/his own visiting history. Besides the positive items, we also randomly picked two negative restaurants (#20 and #124) with the target value of 0. To demonstrate the importance of the self-attention network, we compare GroupSA with Group-S which removes the self-attention network. The attention weights of group members and prediction scores for positive items and negative items are shown in Table IV, from which we can observe that the attention weights of group members vary significantly for different models and different target items. For example, for the positive item #50, Group-S considers that user #101 should have a higher attention weight than others, while GroupSA assigns a higher weight to user #103. But from the prediction results, we can derive that GroupSA is more accurate than Group-S, because for positive items, the prediction scores of GroupSA are closer to the target value of 1; for negative items, GroupSA are closer to the target value of 0 than Group-S. This result demonstrates that GroupSA is capable of assigning higher weights to influential users and thus leads to a better recommendation result.

TABLE IV
Case Studies of a Sampled Group from Yelp on the Effect of Social Self-Attention.

| | Model | User#101 | User#102 | User#103 | $\hat{r}^G$ |
|---|---|---|---|---|---|
| Item#50 | Group-S | **0.4672** | 0.2700 | 0.2628 | 0.8962 |
| | GroupSA | 0.2541 | 0.2519 | **0.4940** | **0.9380** |
| Item#82 | Gropu-S | **0.4724** | 0.2951 | 0.2325 | 0.6970 |
| | GropuSA | 0.3885 | **0.4863** | 0.1251 | **0.8619** |
| Item#20 | Group-S | 0.1121 | **0.5145** | 0.3734 | 0.1071 |
| | GroupSA | **0.6840** | 0.1672 | 0.1488 | **0.0764** |
| Item#124 | Group-S | **0.3793** | 0.2889 | 0.3318 | 0.0954 |
| | GroupSA | 0.3521 | 0.1979 | **0.4500** | **0.0468** |

*2) Importance of User-Modeling (RQ3):* To validate the effectiveness of the item aggregation and social aggregation component of user modeling for the group recommendation task, we conduct another ablation study to compare GroupSA with Group-I and Group-F. From the results presented in Figure 3, we have the following observations: 1) GroupSA outperforms Group-I on both datasets, indicating that aggregating the items that user has interacted with can lead to a better user representation than only considering the user-item interactions. Learning the user latent features from the item-space is helpful for modelling the user's preferences. 2) GroupSA achieves a significant improvement over Group-F

TABLE V
IMPORTANCE OF USER-ITEM INTERACTION DATA ($\Delta$ DENOTES THE IMPROVEMENT OF GROUPSA OVER OTHER METHODS IN HR@$K$).

| Group Recommendation Performance | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Yelp | | | | | | Douban-Event | | | | | |
| | K=5 | | | K=10 | | | K=5 | | | K=10 | | |
| | HR | NDCG | $\Delta\%$ | HR | NDCG | $\Delta\%$ | HR | NDCG | $\Delta\%$ | HR | NDCG | $\Delta\%$ |
| NCF | 0.2869 | 0.2028 | 190.65 | 0.3911 | 0.2364 | 136.69 | 0.2246 | 0.1504 | 205.83 | 0.3261 | 0.1832 | 151.70 |
| Group-G | 0.3339 | 0.2333 | 149.74 | 0.4491 | 0.2706 | 106.12 | 0.2777 | 0.1870 | 147.35 | 0.3972 | 0.2255 | 106.64 |
| GroupSA | **0.8339** | **0.6886** | – | **0.9257** | **0.7186** | – | **0.6869** | **0.5239** | – | **0.8208** | **0.5675** | – |

on both datasets, demonstrating the importance of the social aggregation component on group recommendation. That is, we can learn more accurate user representations by sharing the user latent feature factor between item-space and social-space, since users are likely to be affected by their social connections.

### *B. Importance of User-Item Interaction Data (RQ4)*

To evaluate the effectiveness of the user-item interaction data and the joint training technique, we compare GroupSA with NCF and **Group-G**:

- **Group-G** This is a variant version of GroupSA, which removes the user-item recommendation component and only utilizes the group-item interaction data to learn the group representations. This is to demonstrate the importance of the user-item interaction data.

The experimental results are shown in Table V, from which we can observe that: 1) GroupSA consistently and significantly outperforms Group-G in both datasets. This result demonstrates the importance of training GroupSA with user-item interaction data to address the sparsity issue of group-item interaction data. The experimental results also indicate that our joint training method can effectively perform model optimization on heterogeneous interaction data (i.e., the mixture of both user-item and group-item interaction data). 2) The gap between Group-G and NCF again demonstrates the effectiveness of our social self-attention and user modeling components. Meanwhile, we also notice that, without the help of user-item interaction data, we can only reach sub-optimal group recommendation performance due to the sparsity issue of the group-item interaction data.

### *C. Impact of Hyper-Parameters (RQ5)*

Tables VI-IX show the experimental results of the hyper-parameters. Due to the space limitation, we only report the experimental results on Yelp dataset, and similar results are also observed on Douban-Event Dataset. The hyper-parameter $N_X$ controls how many identical self-attention layers are adopted in GroupSA. As one self-attention layer denotes one voting process of the group members, a stacked self-attention network models the multiple voting processes. From Table VI, we can observe that the best group recommendation performance is achieved when $N_X = 3$. In our experiments, to reduce the training time, we set $N_X = 1$ for Yelp, and $N_X = 2$ for Douban-Event. The hyper-parameter $w^u$ plays a role of determining how much we should dependent on the embeddings learned from the item-space and social-space. As shown in Table VII, as $w^u$ increases, the recommendation performance increases at first, then reaches a peak value (e.g., 0.9), after that, the recommendation performance begins to decrease. This result demonstrates the importance of the user latent factors learnt from item-space and social-space, which can provide better results than only utilizing the user-item interactions.

TABLE VI
IMPACT OF PARAMETER $N_X$.

| $N_X$ | HR@5 | HR@10 | NDCG@5 | NDCG@10 |
|---|---|---|---|---|
| 1 | 0.8339 | 0.9257 | 0.6886 | 0.7186 |
| 2 | 0.8099 | 0.9034 | 0.6489 | 0.6793 |
| 3 | 0.8415 | 0.9253 | 0.6824 | 0.7099 |
| 4 | 0.7971 | 0.8950 | 0.6253 | 0.6572 |
| 5 | 0.8242 | 0.9117 | 0.6746 | 0.7033 |

TABLE VII
IMPACT OF PARAMETER $w^u$.

| $w^u$ | HR@5 | HR@10 | NDCG@5 | NDCG@10 |
|---|---|---|---|---|
| 0.1 | 0.5494 | 0.7430 | 0.3833 | 0.4460 |
| 0.3 | 0.7304 | 0.8718 | 0.5377 | 0.5837 |
| 0.5 | 0.7282 | 0.8753 | 0.5417 | 0.5895 |
| 0.7 | 0.7878 | 0.8995 | 0.6135 | 0.6501 |
| 0.9 | 0.8339 | 0.9257 | 0.6886 | 0.7186 |
| 1.0 | 0.6763 | 0.8517 | 0.4933 | 0.5502 |

TABLE VIII
IMPACT OF PARAMETER $N$.

| $N$ | HR@5 | HR@10 | NDCG@5 | NDCG@10 |
|---|---|---|---|---|
| 1 | 0.8339 | 0.9257 | 0.6886 | 0.7186 |
| 2 | 0.8900 | 0.9424 | 0.7641 | 0.7814 |
| 3 | 0.9166 | 0.9690 | 0.7953 | 0.8125 |
| 4 | 0.9103 | 0.9701 | 0.7642 | 0.7840 |
| 5 | 0.8631 | 0.9288 | 0.7225 | 0.7442 |

TABLE IX
PERFORMANCE ON DIFFERENT GROUP SIZES. $l$ DENOTES THE SIZE OF A GROUP.

| $l$ | HR@5 | HR@10 | NDCG@5 | NDCG@10 |
|---|---|---|---|---|
| $l < 3$ | 0.8295 | 0.9149 | 0.6704 | 0.6984 |
| $3 \leq l \leq 7$ | 0.8816 | 0.9592 | 0.7292 | 0.7547 |
| $7 < l$ | 0.9746 | 0.9980 | 0.8648 | 0.8727 |

We also explore the convergence of our GroupSA when increasing the number of negative examples drawn for per positive sample. As presented in Table VIII, the best performance is achieved when $N = 3$, that is, we do not need to sample too many negative examples when very few of them can already lead to satisfactory results. Therefore, we set $N = 1$ to ensure the training efficiency of GroupSA. To study the performance of GroupSA on different group sizes, we further run experiments for three levels of group sizes (small ($< 3$), medium (3-7) and large ($> 7$)). The groups are classified into bins based on group size, and the parameter

settings of GroupSA are all the same for all the experiments. The results are shown in Table IX, from which we can observe that our method has clear improvements when making recommendations for larger groups. This result demonstrates the expected capability of GroupSA in addressing a relatively complex decision making process to reach a group consensus.

## VI. Related Work

Categorized by the group types, there are two lines of research on group recommendation, that is, persistent group recommendation [8], [10] and occasional group recommendation [6], [7]: (1) Persistent group recommendation mainly focuses on groups with stable members and rich historical interactions. This type of group can be treated as a pseudo user and the traditional personalized recommendation methods such as CF and matrix factorization can be utilized to make group recommendations. (2) Occasional group recommendation mainly aims at modeling groups that are formed by ad-hoc users, and only limited group-item interactions are available. As occasional group recommendation is more general and representative, we focus on making recommendations to this type of group. Existing studies on occasional group recommendations mainly focus on leveraging aggregation approaches, which can be further divided into late aggregation and early aggregation methods.

### A. Late Aggregation Methods

Late aggregation-based methods (also known as score aggregation) [17], [18], [36], [37] aggregate the recommendation results of group members as the group recommendations. Specifically, this kind of method first generates recommendation results or lists for each group member, and then aggregates these individual results to produce the group recommendations via the predefined strategies [7], [12], [17], [18] (such as average satisfaction, least misery and maximum pleasure). For example, average satisfaction assumes that each group member has equal contribution to the group decision making process, and assigns equal importance to the result of each group member. Least misery assumes that the least satisfied member determines the final group decision, and uses the minimum score of group members as the group's preference. However, these aggregation strategies are heuristic and manually predefined, which utilize trivial methods to aggregate each member's individual results. [38] systematically evaluates all the existing predefined strategies, and finds that these aggregation strategies have similar performances, and no single method achieves the best performance on all datasets.

### B. Early Aggregation Methods

Early aggregation-based methods (also known as preference aggregation) [3], [8], [9], [13] aggregate individual preferences of the group members as the group preferences or group recommendations. Specifically, they first aggregate the group members' profiles [5], [39] into a group profile or representation, and then make group recommendations based on this group representation. In these methods, one kind of typical works are based on probabilistic generative models [3], [13], [40], which model groups by capturing not only the group members' personal preferences, but also their impacts in the group. The basic intuition behind these methods is that users may have different contributions to the group and they should be treated differently. For example, PIT [3] captures the item selection process of a group with a probabilistic topic model, and identifies the group preference profile by introducing the notion of personal impact to differentiate the contributions of group members. COM [13] proposes a latent Dirichlet allocation-based generative model to make group recommendations, and estimates the preference of a group by aggregating the preferences of the group members with different weights. However, both models assume that a user has the same probability to follow the group's decision, which is infeasible in the real-world scenario. For example, a movie expert may determine the selection of a movie when he/she goes to the cinema with his/her friends, but he/she may not be one to decide which restaurant to dine in afterward.

Recently, [9] develops an attention-based group recommender system to address the preference aggregation problem, which adopts a vanilla attention mechanism to learn the representation of a group. Compared to our GroupSA method, it has the following drawbacks. First, it does not consider the data sparsity issue of the group-item interaction data. Another drawback is that it does not consider the user's social influence, which is also an important factor that may determine the user's final decision. Finally, the multiple voting processes are not learned, which is one of the most important steps in forming a group's final decision. [8] also considers a vanilla attention to model the process of generating decision makers, but it is mainly proposed for groups with sufficient interactions, and the data sparsity issue and social influence are also not considered. Another recent work is SIGR [6], which adopts the attention mechanism to learn each user's social influence and adapt their social influences to different groups. However, it only models the user's social interests while ignoring the interactions among group members.

In summary, our work falls into the category of early aggregation-based approaches, and it mitigates the data sparsity issue of the group-item interaction data by leveraging the relative abundant user-item and user-user interaction data, and addresses the preference aggregation problem by modeling the interactions among group members.

## VII. Conclusions

In this work, we investigated the occasional group recommendation problem, and proposed a novel group recommender GroupSA by leveraging the power of the self-attention mechanism and an effective aggregation strategy based on it. Specifically, to simulate how a group consensus is reached, we treated the group decision making process as multiple voting processes, and proposed a stacked social self-attention network to learn the voting scheme of group members. To alleviate the sparsity issue of group-item interaction data, we resort to the relative abundant user-item and user-user interaction data,

where two types of aggregation methods and a joint training technique were proposed. We conducted a wide range of experiments on two real-world datasets, and the experimental results demonstrate the effectiveness of our GroupSA method in the OGR task.

Our GroupSA method is generalizable to most decision-making processes in the homogeneous social groups, where the social connections are established based on similar interests, following the principle of homophily. Another different decision-making process may exist in heterogeneous social groups, in which the principle of homophily does not hold. In this case, the decision-making process is subject to more randomness. It is hard for humans to achieve consensus on this occasion, and such kinds of social groups are less likely to participate leisure activities spontaneously. So, we did not consider these types of social groups in our work.

## ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China (Nos. 61602282, 61832001, 61702016, 61572039), ARC Discovery Projects (Grant No. DP190101985 and No. DP170103954), China Postdoctoral Science Foundation (No. 2016M602181), PKU-Tencent joint research Lab and Beijing Academy of Artificial Intelligence (BAAI).

## REFERENCES

[1] H. Yin, L. Zou, Q. V. H. Nguyen, Z. Huang, and X. Zhou, "Joint event-partner recommendation in event-based social networks," in *ICDE*. IEEE, 2018, pp. 929–940.

[2] K. McCarthy, M. Salamó, L. Coyle, L. McGinty, B. Smyth, and P. Nixon, "Cats: A synchronous approach to collaborative group recommendation," in *FLAIRS*, 2006, pp. 86–91.

[3] X. Liu, Y. Tian, M. Ye, and W.-C. Lee, "Exploring personal impact for group recommendation," in *CIKM*. ACM, 2012, pp. 674–683.

[4] H. Yin, B. Cui, L. Chen, Z. Hu, and C. Zhang, "Modeling location-based user rating profiles for personalized recommendation," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 9, no. 3, p. 19, 2015.

[5] H. Yin, Z. Hu, X. Zhou, H. Wang, K. Zheng, Q. V. H. Nguyen, and S. Sadiq, "Discovering interpretable geo-social communities for user behavior prediction," in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 2016, pp. 942–953.

[6] H. Yin, Q. Wang, K. Zheng, Z. Li, J. Yang, and X. Zhou, "Social influence-based group representation learning for group recommendation," in *ICDE*, 2019.

[7] E. Quintarelli, E. Rabosio, and L. Tanca, "Recommending new items to ephemeral groups using contextual user influence," in *RecSys*. ACM, 2016, pp. 285–292.

[8] T. D. Q. Vinh, T.-A. N. Pham, G. Cong, and X.-L. Li, "Attention-based group recommendation," *arXiv preprint arXiv:1804.04327*, 2018.

[9] D. Cao, X. He, L. Miao, Y. An, C. Yang, and R. Hong, "Attentive group recommendation," in *SIGIR*. ACM, 2018, pp. 645–654.

[10] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and W. Cao, "Deep modeling of group preferences for group-based recommendation," in *AAAI*, 2014.

[11] A. Said, S. Berkovsky, and E. W. De Luca, "Group recommendation in context," in *CAMRa*. ACM, 2011, pp. 2–4.

[12] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation and collaborative filtering," in *RecSys*. ACM, 2010, pp. 119–126.

[13] Q. Yuan, G. Cong, and C.-Y. Lin, "Com: a generative model for group recommendation," in *SIGKDD*. ACM, 2014, pp. 163–172.

[14] L. Guo, H. Yin, Q. Wang, T. Chen, A. Zhou, and N. Quoc Viet Hung, "Streaming session-based recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 1569–1577.

[15] T. Chen, H. Yin, H. Chen, R. Yan, Q. V. H. Nguyen, and X. Li, "Air: Attentional intention-aware recommender systems," in *ICDE*. IEEE, 2019, pp. 304–315.

[16] J. Ma, J. Wen, M. Zhong, W. Chen, and X. Li, "Mmm: Multi-source multi-net micro-video recommendation with clustered hidden item representation learning," *Data Science and Engineering*, pp. 1–14, 2019.

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

[18] A. Salehi-Abari and C. Boutilier, "Preference-oriented social networks: Group recommendation and inference," in *RecSys*. ACM, 2015, pp. 35–42.

[19] S. Berkovsky and J. Freyne, "Group-based recipe recommendations: analysis of data aggregation strategies," in *RecSys*. ACM, 2010, pp. 111–118.

[20] L. Boratto and S. Carta, "State-of-the-art in group recommendation and new approaches for automatic identification of groups," in *Information retrieval and mining in distributed environments*. Springer, 2010, pp. 1–20.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[22] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," *arXiv preprint arXiv:1803.02155*, 2018.

[23] J. Liu and Y. Zhang, "Attention modeling for targeted sentiment," in *EACL*, vol. 2, 2017, pp. 572–577.

[24] B. Yang, Z. Tu, D. F. Wong, F. Meng, L. S. Chao, and T. Zhang, "Modeling localness for self-attention networks," *arXiv preprint arXiv:1810.10182*, 2018.

[25] B. Yang, L. Wang, D. Wong, L. S. Chao, and Z. Tu, "Convolutional self-attention networks," *arXiv preprint arXiv:1904.03107*, 2019.

[26] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," *arXiv preprint arXiv:1902.07243*, 2019.

[27] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *SIGKDD*. ACM, 2008, pp. 426–434.

[28] H. C. Wu, R. W. P. Luk, K. F. Wong, and K. L. Kwok, "Interpreting tf-idf term weights as making relevance decisions," *TOIS*, vol. 26, no. 3, p. 13, 2008.

[29] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual review of sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[30] J. Guo, Y. Zhu, A. Li, Q. Wang, and W. Han, "A social influence approach for group user modeling in group recommendation systems," *IEEE Intelligent Systems*, vol. 31, no. 5, pp. 40–48, 2016.

[31] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *AUAI*. AUAI Press, 2009, pp. 452–461.

[32] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *WWW*, 2017, pp. 173–182.

[33] X. He, T. Chen, M.-Y. Kan, and X. Chen, "Trirank: Review-aware explainable recommendation by modeling aspects," in *CIKM*. ACM, 2015, pp. 1661–1670.

[34] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *RecSys*. ACM, 2010, pp. 39–46.

[35] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *AI and statistics*, 2010, pp. 249–256.

[36] A. Crossen, J. Budzik, and K. J. Hammond, "Flytrap: intelligent group music recommendation," in *IUI*. ACM, 2002, pp. 184–185.

[37] L. Xiao, Z. Min, Z. Yongfeng, G. Zhaoquan, L. Yiqun, and M. Shaoping, "Fairness-aware group recommendation with pareto-efficiency," in *RecSys*. ACM, 2017, pp. 107–115.

[38] T. De Pessemier, S. Dooms, and L. Martens, "Comparison of group recommendation algorithms," *Multimedia tools and applications*, vol. 72, no. 3, pp. 2497–2541, 2014.

[39] X. Ruifeng, D. Jiachen, Z. Zhishan, H. Yulan, G. Qinghong, and L. Lin, "Inferring user profiles in social media by joint modeling of text and networks," *SCIENCE CHINA Information Sciences*.

[40] J. Gorla, N. Lathia, S. Robertson, and J. Wang, "Probabilistic group recommendation via information matching," in *WWW*. ACM, 2013, pp. 495–504.