Attentive Group Recommendation

Da Cao
Hunan University
caoda0721@gmail.com

Xiangnan He
National University of Singapore
xiangnanhe@gmail.com

Lianhai Miao
Hunan University
lianhaimiao@gmail.com

Yahui An
University of Electronic Science and Technology of China
anyahui.120@gmail.com

Chao Yang*
Hunan University
yangchaoedu@hnu.edu.cn

Richang Hong
Hefei University of Technology
hongrc.hfut@gmail.com

ABSTRACT

Due to the prevalence of group activities in people's daily life, recommending content to a group of users becomes an important task in many information systems. A fundamental problem in group recommendation is how to aggregate the preferences of group members to infer the decision of a group. Most existing group recommender systems applied a predefined strategy for preference aggregation, such as average, least misery, maximum satisfaction and so on. We argue that such static strategies are insufficient to model the complicated process of group decision making and result in the suboptimal performance for group recommendation. In particular, group members should have non-uniform weights in forming a group, and more importantly, the weights should be varied when the group interacts with different items.

In this work, we address the fundamental problem of preference aggregation in group recommendation by learning the aggregation strategy from data. We contribute a novel solution, namely AGREE (short for "**A**ttentive **G**roup **RE**comm**E**ndation"), based on the recent developments of neural attention network and neural collaborative filtering (NCF). Specifically, we adopt an attention mechanism to adapt the representation of a group, and learn the interaction between groups and items from data under the NCF framework. Moreover, since many group recommender systems also have abundant interactions of individual users on items, we further integrate the modeling of user-item interactions into our AGREE model. Through this way, we can reinforce the two tasks of recommending items for both groups and users. By experimenting on two real-world datasets, we demonstrate that our AGREE model not only improves the group recommendation performance but also enhances the recommendation for users, especially for these cold-start users that have no historical interactions individually.

CCS CONCEPTS

•Information systems → Recommender systems; Collaborative filtering; •Computing methodologies → Neural networks;

*Corresponding author.

KEYWORDS

Recommender Systems, Group Recommendation, Attention Mechanism, Neural Collaborative Filtering, Cold-Start Problem

1 INTRODUCTION

To address the information overload issue, recommender systems have been widely deployed in online information systems, such as E-commerce platforms, social media sites, news portals and so on. An effective recommendation solution not only can increase the traffic and profit for service providers, but also can help customers find the items of interest more easily [6, 7]. Moving beyond the traditional task of recommending content for individual users, in this work we focus on providing recommendation for a group of users, known as the *group recommendation* task.

Owing to the prevalence of social media, online group activities have become very common in current social Web. For example, a group of traveler can work together to plan a trip on Mafengwo[1], a group of teenagers can organize a social party on Meetup[2], and a group of researchers can discuss a paper on Mendeley[3]. Such rich sources of information provide us a valuable opportunity to study the online behaviors of groups, which in turn benefit us to develop a better recommendation service for groups.

Distinct to the decision making of an individual user, the decision making process of a group is more complicated, since each member in the group may contribute to the final decision. More importantly, the process can be rather dynamic. For example, a member may play different roles and exhibit different influence in choosing items of different types due to her specialty. Nevertheless, existing recommendation solutions largely applied a predefined and fixed strategy to aggregate the preferences of group members, such as average [3, 4], least misery [1], maximum satisfaction [5] and so on. As such, these solutions are insufficient to capture the complicated and dynamic process of group decision making, resulting in the suboptimal performance for group recommendation.

In this work, we approach the fundamental problem in group recommendation — how to aggregate the preference of group members to decide a group's choice on items. Instead of applying a predefined strategy, we propose to learn the aggregation strategy from the historical data of group-item interactions. The key challenges here are how to design an expressive model to learn the influence of a member, and how to adapt the influence when the

[1]https://www.mafengwo.com
[2]https://www.meetup.com
[3]https://www.mendeley.com

group interacts with different items. Inspired by the recent success of representation learning, we approach the challenges from the perspective of group representation learning in the embedding space. To get the embedding vector that represents a group, we aggregate the embedding of its members in a learnable way. Specifically, we design a neural attention network to learn the weight of a member, which is capable of assigning different weights for a user when the group interacts with different items. Through this way, we can dynamically adjust the aggregation strategy for a group to capture the complicated process of group decision making.

Moreover, many group-aware social platforms also have abundant data of user-item interactions, the key data source to infer user preference to address the traditional item recommendation task [8, 34]. Intuitively, a group's decision should be dependent on the preference of its members. As such, another challenge here is how to effectively leverage the user-item interaction data to improve the performance of group recommendation. To this end, we propose to address both tasks of group recommendation and item recommendation simultaneously under the same framework. Specifically, we employ the state-of-the-art neural collaborative filtering (NCF) framework [18] to learn the user-item and group-item interactions in the same embedding space. Through this way, we not only improve the performance of both tasks, but also enable recommendation for cold-start users that have no historical individual behaviors by leveraging their group data.

The main contributions of this work are summarized as follows:

- This is the first group recommender system that leverages neural attention network to learn the aggregation strategy from data in a dynamic way.
- We integrate the modeling of user-item interactions into group recommendation and improve its performance. As a byproduct, the new-user cold-start problem in item recommendation can be alleviated.
- We conduct extensive experiments on a self-constructed dataset and a public dataset to demonstrate the effectiveness of our method. Meanwhile, we have released the datasets and our implementation to facilitate the research community[4].

2 METHODS

Generally speaking, our proposed AGREE model consists of two components: 1) group representation learning which represents a group based on its members aggregation and its general preference; and 2) interaction learning with NCF which recommends items for both users and groups. We first present the notations and formulate the group recommendation problem to be solved (Section 2.1). We then introduce the two key ingredients of our proposed model (Section 2.2 and 2.3). Lastly, we discuss the optimization method (Section 2.4).

2.1 Notations and Problem Formulation

We use bold capital letters (e.g., $\mathbf{X}$) and bold lowercase letters (e.g., $\mathbf{x}$) to represent matrices and vectors, respectively. We employ non-bold letters (e.g., x) to denote scalars, and squiggle letters (e.g., $\mathcal{X}$) to denote sets. If not clarified, all vectors are in column forms.

[4]https://github.com/LianHaiMiao/Attentive-Group-Recommendation

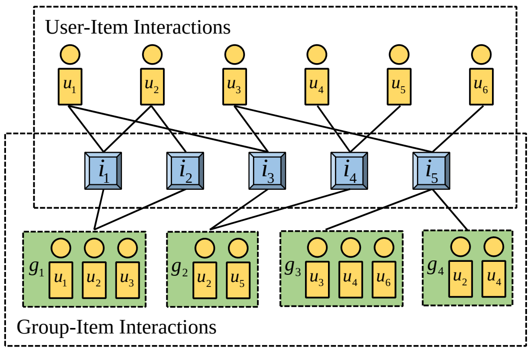


Figure 1: Illustration of the input data of attentive group recommendation task, which contains user-item interactions and group-item interactions.

Figure 1 illustrates the group recommendation task we address in this paper. Suppose we have n users $\mathcal{U} = \{u_1, u_2, ..., u_n\}$, s groups $\mathcal{G} = \{g_1, g_2, ..., g_s\}$, and m items $\mathcal{V} = \{v_1, v_2, ...v_m\}$. The l-th group $g_l \in \mathcal{G}$ is consisted of a set of users, i.e., group members with user indexes $\mathcal{K}_l = \{k_{l,1}, k_{l,2}, ..., k_{l,|g_l|}\}$, where $u_{k_{l,*}} \in \mathcal{U}$, and $|g_l|$ is the size of the group. There are two kinds of observed interaction data among $\mathcal{U}$, $\mathcal{G}$, and $\mathcal{V}$, namely, group-item interactions and user-item interactions. We use $\mathbf{Y} = [y_{lj}]_{s\times m}$ to denote the group-item interactions and $\mathbf{R} = [r_{ij}]_{n\times m}$ to denote the user-item interactions. Then, given a target group g_l (or target user u_i), our task is defined as recommending a list of items that group g_l (or the user u_i) may be interested in, which is formally defined as:

Input: Users $\mathcal{U}$, groups $\mathcal{G}$, items $\mathcal{V}$, group-item interactions $\mathbf{Y}$, and user-item interactions $\mathbf{R}$.

Output: Two personalized ranking functions that map an item to a real value for each group $f_g : \mathcal{V} \to \mathbb{R}$ and each user $f_u : \mathcal{V} \to \mathbb{R}$, respectively.

2.2 Attentive Group Representation Learning

Most existing group recommender systems aggregate the scores of group members via some predefined strategies. We first recapitulate these common aggregation strategies to motivate our use of the attention mechanism to learn group representation, and then present our designed group representation learning component.

2.2.1 Motivation. The average strategy [3, 4] treats group members as contributing equally, and estimates the group preference over an item as the arithmetic mean of the preference scores of its members. The least misery strategy [1] tries to please all members in a group by choosing the lowest preference score among its members over an item. The maximum satisfaction strategy [5] tries to maximize the greatest preference of a group, and it averages the preference scores of members above a certain threshold. The expertise strategy [33] endows an individualized weight for a user based on her expertise on the items.

We argue that these predefined strategies are data independent, lacking the flexibility to dynamically adjust the weights of group members. This flexibility is particularly useful when a group makes decision on items of different types. Inspired by the recent developments of neural attention mechanism [2, 40] which can learn the importance of different model components from data, we consider using attention to learn the aggregation strategy. The idea of attention is to perform weighted sum on a set of representations when compressing them into one representation, where the weights are learned by a neural network. We find that it is a natural fit for addressing the group recommendation task, and more importantly, existing aggregation strategies can be regarded as its special cases. To be specific, the *average* is equivalent to giving a uniform weight to all members, the *least misery* and *maximum satisfaction* correspond to assigning nonzero weights to partial members only, and the *expertise* can be seen as fixing the attention weights based on the members' expertise.

2.2.2 Method. We propose to address the group recommendation problem under the representation learning (RL) framework. Under the RL paradigm, each entity of interest is represented as an embedding vector, which encodes the inherent properties of the entity (e.g., semantics of a word, interests of a user etc.) and is to be learned from data. The well-known matrix factorization method in item recommendation [19, 28] is a typical RL model that associates each user and item with an embedding vector.

Let $\mathbf{u}_i$ and $\mathbf{v}_j$ be the embedding vector for user u_i and item v_j, respectively, which are basic representation blocks in our AGREE model. Our target is to obtain an embedding vector for each group to estimate its preference on an item. To learn dynamic aggregation strategy from data, it is necessary to define the group embedding as dependent of the embeddings of its member users and the target items, which can be abstracted as,

$$\mathbf{g}_l(j) = f_a(\mathbf{v}_j, \{\mathbf{u}_t\}_{t\in\mathcal{K}_l}), \tag{1}$$

where $\mathbf{g}_l(j)$ denotes the embedding of group g_l tailored for predicting its preference on target item v_j, $\mathcal{K}_l$ contains the user indexes of group g_l, and f_a is the aggregation function to be specified. In AGREE, we design the group embedding as consisting of two components — user embedding aggregation and group preference embedding:

$$\mathbf{g}_l(j) = \underbrace{\sum_{t\in\mathcal{K}_l} \alpha(j,t)\mathbf{u}_t}_{\text{user embedding aggregation}} + \underbrace{\mathbf{q}_l}_{\text{group preference embedding}}. \tag{2}$$

Next we elaborate the two components.

User embedding aggregation. We perform a weighted sum on the embeddings of group g_l's member users, where the coefficient $\alpha(j,t)$ is a learnable parameter denoting the influence of member user u_t in deciding the group's choice on item v_j. Intuitively, if a user has more expertise on an item (or items of the similar type), she should have a larger influence on the group's choice on the item [33]. To understand this, let us consider an example that a group discusses which city to travel to; if a user has traveled to China many times, she should be more influential when the group considers whether should travel to a city in China. Since in the RL framework, embedding $\mathbf{u}_t$ encodes the member user's historical preference and embedding $\mathbf{v}_j$ encodes the target item's property, we parameterize $\alpha(j,t)$ as a neural attention network with $\mathbf{u}_t$ and

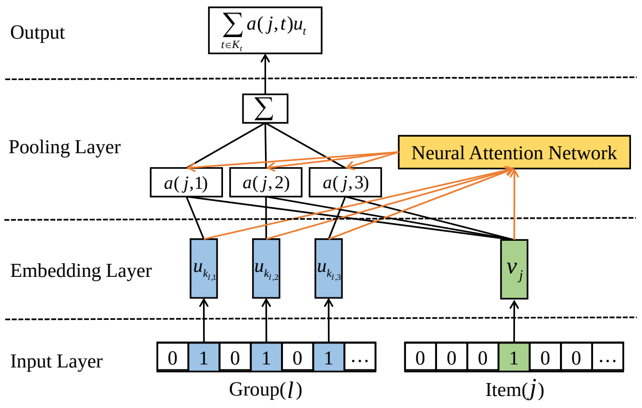


Figure 2: Illustration of the user embedding aggregation component based on neural attention network.

$\mathbf{v}_j$ as the input:

$$o(j,t) = \mathbf{h}^T \text{ReLU}(\mathbf{P}_v \mathbf{v}_j + \mathbf{P}_u \mathbf{u}_t + \mathbf{b}),$$
$$\alpha(j,t) = \text{softmax}(o(j,t)) = \frac{\exp o(j,t)}{\sum_{t'\in\mathcal{K}_l} \exp o(j,t')}, \tag{3}$$

where $\mathbf{P}_v$ and $\mathbf{P}_u$ are weight matrices of the attention network that convert item embedding and user embedding to hidden layer, respectively, and $\mathbf{b}$ is the bias vector of the hidden layer. We use ReLU as the activation function of the hidden layer, and then project it to a score $o(j,t)$ with a weight vector $\mathbf{h}$. Lastly, we normalize the scores with a softmax function, which is a common practice in neural attention network [2, 40]; it makes the attention network a probabilistic interpretation, which can also deal with groups of different sizes in our case.

Figure 2 illustrates our design of the user embedding aggregation component. With such a soft attention mechanism, we allow each member user to contribute in a group's decision, where the contribution of a user is dependent on her historical preference and the target item's property, which are learned from past data of group-item interactions and user-item interactions (to be discussed in Section 2.3).

Group preference embedding. Besides aggregating the embeddings of group members, we further associate a group g_l with a dedicated embedding $\mathbf{q}_l$. The intention is to take the general preference of a group into account. Our consideration is that in some cases when users form a group, they may pursue a target that is different from the preference of each user. For example, in a family of three, the child prefers cartoon movie and the parents favor romantic movie; but when they go to a cinema together, the final chosen movie could be an educational movie. As such, it is beneficial to associate a group with an embedding to denote its general preference, in addition to the one aggregated from its members. To combine the components of group preference embedding with user embedding aggregation, we perform a simple addition operation, same as the previous work SVD++ [28] and ACF [9] that combine different signals in the embedding space. Our empirical results in Section 3.4 show that this component can significantly improve the group recommendation performance.

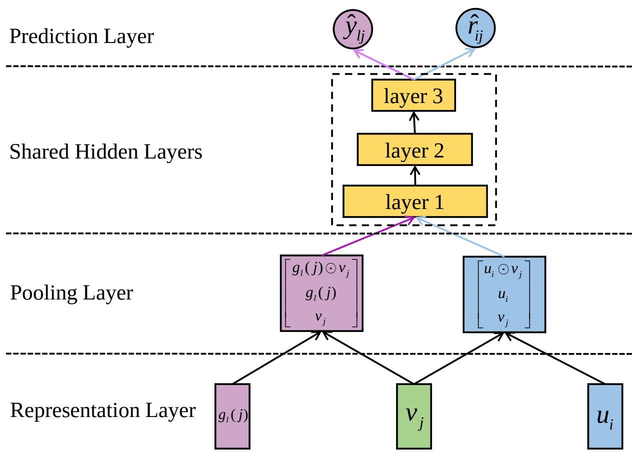


Figure 3: Illustration of interaction learning based on NCF.

2.3 Interaction Learning with NCF

NCF is a multi-layer neural network framework for item recommendation [18]. Its idea is to feed user embedding and item embedding into a dedicated neural network (which needs to be customized) to learn the interaction function from data. As neural networks have the strong ability to fit the data, the NCF framework is more generalizable than the traditional matrix factorization (MF) framework [28], which simply applies a data-independent inner product function as the interaction function. As such, we opt for the NCF framework to perform an end-to-end learning on both embeddings (that represent users, items, and groups) and interaction functions (that predict user-item and group-item interactions).

Figure 3 illustrates our customized NCF solution. Since we aim to achieve both recommendation tasks for groups and users simultaneously, we design the solution to learn the user-item and group-item interaction functions together. Specifically, given a user-item pair (u_i, v_j) or a group-item pair (g_l, v_j), the representation layer first returns the embedding vector for each given entity (details see Section 2.2). Then the embeddings are fed into a pooling layer and hidden layers (shared by the two tasks) to obtain the prediction score. Next we elaborate the two components.

Pooling layer. Assuming the input is a group-item pair (g_l, v_j), the pooling layer first performs element-wise product on their embeddings, i.e., $\mathbf{g}_l(j)$ and $\mathbf{v}_j$, and then concatenates it with the original embeddings:

$$\mathbf{e}_0 = \varphi_{pooling}(\mathbf{g}_l(j), \mathbf{v}_j) = \begin{bmatrix} \mathbf{g}_l(j) \odot \mathbf{v}_j \\ \mathbf{g}_l(j) \\ \mathbf{v}_j \end{bmatrix} \tag{4}$$

The rationale is twofold. 1) The element-wise product subsumes MF, which uses multiplication on each embedding dimension to model the interation between two embedding vectors; moreover, element-wise product has been demonstrated to be highly effective in feature interaction modeling in low-level of neural architecture [17]. 2) Nevertheless, the element-wise product may lose some information in the original embeddings which may be useful for later interaction learning. To avoid such information loss, we further concatenate it with the original embeddings.

Note that such a pooling operation is partially inspired from the state-of-the-art neural recommender model NeuMF [18], which shows that combines MF with MLP in the hidden layer leads to better performance. As MLP concatenates the original user embedding and item embedding, it inspires us to keep the original embeddings to facilitate the learning of later hidden layers. We apply the same pooling operation for the input of a (u_i, v_j) pair.

Shared Hidden layers. Above the pooling layer is a stack of fully connected layers, which enable the model to capture the nonlinear and higher-order correlations among users, groups, and items.

$$\begin{cases} \mathbf{e}_1 = \text{ReLU}(\mathbf{W}_1\mathbf{e}_0 + \mathbf{b}_1) \\ \mathbf{e}_2 = \text{ReLU}(\mathbf{W}_2\mathbf{e}_1 + \mathbf{b}_2) \\ \ldots\ldots \\ \mathbf{e}_h = \text{ReLU}(\mathbf{W}_h\mathbf{e}_{h-1} + \mathbf{b}_h) \end{cases}, \tag{5}$$

where $\mathbf{W}_h$, $\mathbf{b}_h$, and $\mathbf{e}_h$ denote the weight matrix, bias vector, and output neurons of the h-th hidden layer, respectively. We use ReLU function as the non-linear activation function, which has empirically shown to work well, especially for tower-structure feed-forward neural networks [12, 18]. Also, we use the tower structure for hidden layers and leave the further tuning on the structure as future work. Finally, the output of the last hidden layer $\mathbf{e}_h$ is transformed to a prediction score via:

$$\begin{cases} \hat{r}_{ij} = \mathbf{w}^T\mathbf{e}_h,\ if\ \mathbf{e}_0 = \varphi_{pooling}(\mathbf{u}_i, \mathbf{v}_j) \\ \hat{y}_{lj} = \mathbf{w}^T\mathbf{e}_h,\ if\ \mathbf{e}_0 = \varphi_{pooling}(\mathbf{g}_l(j), \mathbf{v}_j) \end{cases}, \tag{6}$$

where $\mathbf{w}$ denotes the weights of the prediction layer; $\hat{r}_{ij}$ and $\hat{y}_{lj}$ represent the prediction for a user-item pair (u_i, v_j) and a group-item pair (g_l, v_j), respectively.

It is worth mentioning that we have purposefully designed the prediction of the two tasks share the same hidden layers. This is because that the group embedding is aggregated from user embeddings, which makes them in the same semantic space by nature. Moreover, this can augment the training of group-item interaction function with user-item interaction data and vice versa, which facilitates the two tasks reinforcing each other.

2.4 Model Optimization

2.4.1 Objective Function. Since we address recommendation task from the ranking perspective, we opt for pairwise learning method for optimizing model parameters. The assumption of pairwise learning is that an observed interaction should be predicted with a higher score than its unobserved counterparts. Specifically, we employ the regression-based pairwise loss, which is a common choice in item recommendation [26]:

$$\mathcal{L}_{user} = \sum_{(i,j,s)\in O} (r_{ijs} - \hat{r}_{ijs})^2 = \sum_{(i,j,s)\in O} (\hat{r}_{ij} - \hat{r}_{is} - 1)^2, \tag{7}$$

where O denotes the training set, in which each instance is a triplet (i, j, s) meaning that user u_i has interacted with item v_j, but has not interacted with item v_s before (i.e., v_s is a negative instance sampled from the unobserved interactions of u_i); $\hat{r}_{ijs} = \hat{r}_{ij} - \hat{r}_{is}$, means the margin of the prediction of observed interaction (u_i, v_j) and unobserved interaction (u_i, v_s). Since we focus on implicit feedback, where each observed interaction has a value of 1 and unobserved interaction has a value of 0, we have $r_{ijs} = r_{ij} - r_{is} = 1$.

We are aware that another prevalent pairwise learning method in recommendation is the Bayesian Personalized Ranking (BPR) [34]. It is worth pointing out that an advantage of the above regression-based pairwise loss over BPR is that it eliminates the need of tuning the L_2 regularization for the weights in the hidden layers (i.e., $\{\mathbf{W}_h\}$ and $\mathbf{w}$). In BPR, the loss for an instance (i, j, s) is formulated as $-\log \sigma(\hat{r}_{ij} - \hat{r}_{is})$, where σ is the sigmoid function. To decrease the BPR loss on a multi-layer model, a trivial solution is to scale up the weights in each update. As such, it is crucial to enforce the L_2 regularization on the weights to avoid this trivial solution. In contrast, our chosen loss optimizes the margin term $\hat{r}_{ij} - \hat{r}_{is}$ towards 1, making such a trivial solution fail to decrease the loss. Thus the weights can be learned without any constraint on it.

Similarly, we can obtain the pairwise loss function for optimizing the group recommendation task:

$$\mathcal{L}_{group} = \sum_{(l,j,s)\in O'} (y_{ljs} - \hat{y}_{ljs})^2 = \sum_{(l,j,s)\in O'} (\hat{y}_{lj} - \hat{y}_{ls} - 1)^2, \quad (8)$$

where O' denotes the training set for the group recommendation task, in which each instance (l, j, s) means that group g_l has interacted with item v_j, but has not interacted with v_s before.

2.4.2 Learning Details. We present some learning details that are important to replicate our method.

Mini-batch training. We perform mini-batch training, where each mini-batch contains both user-item and group-item interactions. Specifically, we first shuffle all observed interactions, and then sample a mini-batch of observed interactions. For each observed interaction, we sample a fixed number of negative instances to form the training instances.

Pre-training. It is known that neural networks are rather sensitive to initialization [15, 18]. To better train AGREE, we pre-train it with a simplified version that removes the attention network, i.e., assigning a uniform weight on user embeddings to obtain the group embedding. With the pre-trained model as an initialization, we further train the AGREE model. Note that we employ Adam [27] in the pre-training phase, which has a fast convergence owing to its adaptive learning rate strategy. After pre-training, we use the vanilla SGD, a common choice in fine-tuning a pre-trained model.

Dropout. As the neurons of fully connected layers can easily co-adapt [38], we employ dropout to improve AGREE's generalization performance. Specifically, in the pooling layer, we randomly drop ρ percent of the $\mathbf{e}_0$ vector. Moreover, we also apply dropout on the hidden layer of the neural attention network and the hidden layers of NCF interaction learning component. Note that dropout is only used during training (i.e., computing gradients with back-propagation), and must be disabled during the prediction phase.

3 EXPERIMENTS

In this section, we conduct extensive experiments on one self-collected dataset and one public dataset to answer the following four research questions:

RQ1 How is the effectiveness of our designed attention network? Can it provide better group recommendation performance?

RQ2 How does our proposed AGREE approach perform as compared with state-of-the-art group recommender systems?

RQ3 How do the two components of group representation — user embedding aggregation and group preference embedding — contribute to the performance of AGREE?

RQ4 How does AGREE perform in handling the new-user cold-start problem?

3.1 Experimental Settings

3.1.1 Datasets. We experimented with two real-world datasets, one is crawled from a tourism website Mafengwo[5] and the other one is a publicly accessible dataset released from the competition of context-aware movie recommendation[6].

1. Mafengwo. Mafengwo is a tourism website where users can record their traveled venues, create or join a group travel. We retained the groups which have at least 2 members and have traveled at least 3 venues, and collected their traveled venues. The traveled venues of each group member were also collected. Based on the above criteria, we obtained 5, 275 users, 995 groups, 1, 513 items, 39, 761 user-item interactions, and 3, 595 group-item interactions. On average, each group has 7.19 users.

2. CAMRa2011. CAMRa2011 is a real-world dataset containing the movie rating records of individual users and households. Since the majority of users have no group information in the dataset, we filtered them out and retained users who have joined a group. The user-item interactions and group-item interactions are explicit feedback with the rating scale of [0, 100]. We transformed the rating records to positive instances with the target value of 1 and left the other missing data as negative instances with the target value of 0. The final dataset contained 602 users, 290 groups, 7, 710 items, 116, 344 user-item interactions, and 145, 068 group-item interactions. The average group size is 2.08.

As both datasets only contain positive instances (i.e., observed interactions), we randomly sampled various missing data as negative instances to pair with each positive instance. Previous efforts have shown that increasing the negative sampling ratio from 1 to larger values is beneficial to the top-K recommendation [18]. For AGREE on both datasets, the optimal sampling ratio is around 4 to 6, so we fixed the negative sampling ratio as 4. Specifically, for each log of Mafengwo, we randomly sampled 4 venues that the user (group) has never visited; for each log of CAMRa2011, we randomly sampled 4 movies that the user (group) has never watched. Each negative instance is assigned to a target value of 0.

3.1.2 Evaluation Protocols. We adopted the *leave-one-out* evaluation protocol, which has been widely utilized to evaluate the performance of the top-K recommendation [9, 18, 34]. Specifically, for each user (group), we randomly removed one of her (its) interactions for testing. This results in disjoint training set $\mathcal{S}_{train}$ and testing set $\mathcal{S}_{test}$. Since it is too time-consuming to rank all items for each user and group, we followed the common scheme [14, 28] that randomly selected 100 items that were not interacted by the user or the group and ranked the testing item among the 100 items. For the new-user cold-start issue, we conducted experiments on the Mafengwo dataset. Among all the 5, 277 users, 1, 296 (24.6%)

[5] http://www.mafengwo.cn

[6] http://2011.camrachallenge.com/2011

of them have interactions on items and joined at least 2 groups. We randomly selected 10% of these users as the testing users and removed their user-item interactions. To evaluate the performance of the top-K recommendation, we employed the widely used metric — Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG). Large values indicate better performance. In leave-one-out evaluation, HR measures whether the testing item is ranked in the top-K list (1 for yes and 0 for no), while NDCG accounts for the position of the hit by assigning higher score to hit at top positions.

3.1.3 Baselines. To justify the effectiveness of our method, we compared it with the following methods.

- **NCF.** This method treats a group as a virtual user and ignores the member information of the group [35]. Then users and virtual users are embedded into our NCF solution.
- **Popularity [13].** This method recommends items to users and groups based on the popularity of items. The popularity of an item is measured by its number of interactions in the training set. It is a non-personalized method to benchmark the performance of other personalized methods.
- **COM [41].** This is the state-of-the-art group recommender system. It is a probabilistic method that models the generative process of group activities. We used the implementation released by the authors and modified the evaluation codes only to adapt our testing scenario. We tuned the number of topics for COM and left other hyper-parameters as the default setting.
- **GREE.** This is a variant of our AGREE method by removing the attention component and setting a uniform weight on the member embeddings. This is to demonstrate the effect of learning varying weights for group members.

To justify the usefulness of learning the aggregation strategy from data, we further compare with another line of methods that apply a predefined score aggregation strategy. For these methods, we first run the NCF method to predict the individual preference scores, and then apply the aggregation strategy to get the group preference score.

- **NCF+avg [3, 4].** NCF+avg is short for "NCF combined with average". It is the simplest aggregation strategy that averages the preference scores of individuals as the group preference score. The hypothesis behind this method is that each member contributes equally to the final group decision.
- **NCF+lm [1].** NCF+lm applies the least misery strategy. It tries to please all members in a group, which uses the minimum score of individuals as the group preference score. The underlying assumption is that the least satisfied member determines the final group decision, which is similar to the well-known *cask principle*.
- **NCF+ms [5].** NCF+ms employs the maximum satisfaction strategy. In contrast to NCF+lm, it tries to maximize the satisfaction of group members. It averages the individual scores above a specified threshold as the group preference score. In this work, we assumed a member prefers to follow other members' options, and treated the maximum score as the preference of the group.
- **NCF+exp [33].** NCF+exp adopts the expertise scheme. It applies a weighted average on individual scores, where the weight reflects the expertise of the user. In our experiments, the expertise of a user is defined as the number of items she has interacted with in the training set.

3.1.4 Implementation and Hyper-Parameter Setting. We implemented our method based on Pytorch[7], which can be accessible at http://grouprec.wixsite.com/agree. For hyper-parameter tuning, we randomly sampled one interaction for each user and group as the validation set. As have mentioned before, the negative sampling ratio was set to 4. For the initialization of the embedding layer, we applied the Glorot initialization strategy [16], which was found to have a good performance. For hidden layers, we randomly initialized their parameters with a Gaussian distribution of a mean of 0 and a standard deviation of 0.1. We used the Adam optimizer for all gradient-based methods, where the mini-batch size and learning rate were searched in $[128, 256, 512, 1024]$ and $[0.001, 0.005, 0.01, 0.05, 0.1]$, respectively. In neural attention network and NCF, we empirically set the size of the first hidden layer same as the embedding size with the dimension of 32, and employed three layers of a tower structure and ReLU activation function. We repeated each setting for 5 times and reported the average results. We further conducted the paired two-sample t-test on NDCG based on the 5 times experiment results.

3.2 Effect of Attention (RQ1)

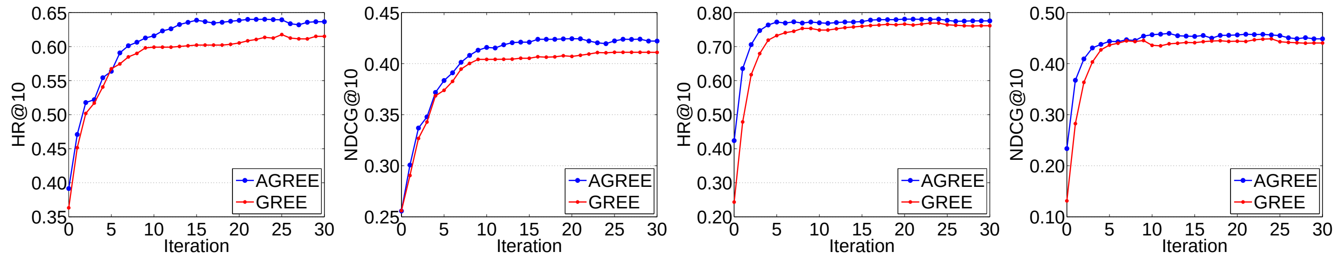
The primary motivation of this work is to learn variable attention weights for group members, rather than the commonly used uniform weighting strategy. Therefore, in order to investigate the effectiveness of the attention network, we first compare the performance of AGREE with the GREE baseline.

Figure 4 shows the performance of AGREE and GREE in each training iteration under the optimal parameter settings. We have the following observations: 1) Compared with GREE, AGREE achieves a relative improvement on both datasets with respect to both metrics. The improvements are statistically significant and mainly stem from the strong representation power of the attention network. 2) Both AGREE and GREE converge rather fast, reaching their stable performance around the 20th iteration. Compared with GREE, AGREE additionally uses an attention network to re-weight the embedding vectors of group members. This improves generalization without affecting the convergence speed, which provides evidence on the effectiveness and rationality of AGREE.

Micro-Level Analysis. Apart from the superior recommendation performance, another key advantage of AGREE is its ability in interpreting the attention weights of group members. To demonstrate this, we performed some micro-level case studies. To be specifically, we implemented AGREE in a two-stage scheme. After obtaining the GREE model, we fixed the parameters and trained the attention network only to make the effect of attention more distinct. Prediction scores of the group toward positive and negative items are investigated.

We randomly selected a testing group which consists of three users (#805, #806, and #807), and the group has traveled three venues (#30 Argentina, #32 Chile, and #106 Bolivia) with the target value

[7]http://www.pytorch.org



(a) Mafengwo — HR@10 (b) Mafengwo — NDCG@10 (c) CAMRa2011 — HR@10 (d) CAMRa2011 — NDCG@10

Figure 4: Performance of AGREE and GREE in each training iteration on Mafengwo and CAMRa2011 datasets (Section 3.2).

Table 1: Case studies of a sampled group on the effect of attention (Section 3.2). The member weights and prediction scores of the group for positive venues (Venue #30, #32, #106) and negative venues (Venue #65, #121, #123) are shown (Section 3.2).

	Model	User #805	User #806	User #807	$\hat{y}$
Venue #30	GREE	0.333	0.333	0.333	0.260
	AGREE	0.286	0.302	**0.412**	**0.572**
Venue #32	GREE	0.333	0.333	0.333	0.096
	AGREE	0.222	**0.583**	0.195	**0.370**
Venue #106	GREE	0.333	0.333	0.333	0.192
	AGREE	**0.364**	0.287	0.347	**0.318**
Venue #65	GREE	0.333	0.333	0.333	0.132
	AGREE	**0.408**	0.311	0.281	**0.091**
Venue #121	GREE	0.333	0.333	0.333	0.132
	AGREE	0.335	**0.374**	0.291	**0.053**
Venue #123	GREE	0.333	0.333	0.333	0.109
	AGREE	0.288	**0.411**	0.301	**0.063**

of 1. Each group member also has her owning traveling history[8]. Besides traveled venues, we also randomly picked three negative venues (#65 Iran, #121 Qiandao Lake, and #123 Baoji) with the target value of 0.

Table 1 shows the attention weights and prediction score for the group of GREE and AGREE. We have the following observations: 1) For different target venues, the attention weights of group members vary significantly in AGREE. For example, when predicting the group's preference on negative venues #121 and #123, the attention weights of user #806 are relatively high. This is probably because that the user has traveled a lot of Chinese venues (#547 Jiangxi, #62 Yunnan, and #553 Hunan), and thus she has more power in deciding whether the group should travel to other Chinese venues (note that #121 Qiandao Lake and #123 Baoji are Chinese venues). 2) For positive venues, the prediction scores of AGREE are much larger than that of GREE and are closer to the target value of 1. While for negative venues, the prediction scores of AGREE are closer to the target value of 0 than that of GREE. As GREE assigns the same weight for all members in the group, the model's representation ability is limited. By augmenting GREE with a learnable attention network, AGREE is capable of assigning higher weights for influential users and thus leads to better recommendation performance.

[8]User #805 has traveled #58 Brazil, #31 Trukey, and #136 Japan; user #806 has traveled #547 Jiangxi, #62 Yunnan, and #553 Hunan; user #807 has traveled #139 Los Angeles and #86 New York.

3.3 Overall Performance Comparison (RQ2)

Now we compare the performance of ARGEE with the baselines of interest. Note that since COM and score aggregation methods are specially designed for group recommendation, they can not provide recommendation for individual users.

Table 2 and Table 3 show the results on Mafengwo and CAMRa2011, respectively. We have the following observations: 1) Our AGREE method achieves the best performance on the two datasets for both recommendation tasks, significantly outperforming state-of-the-art methods (all the p-values between our model and each baseline are much smaller than 0.05, which indicates that the improvements are statistically significant). This validates the effectiveness of our AGREE solution, more specifically, the positive effect of the attention network in aggregating the preference of group members and simultaneously addressing the two tasks. 2) The performance of neural network-based solutions (i.e., NCF, GREE, NCF+avg, NCF+lm, NCF+ms, NCF+exp, and AGREE) are superior to that of non-personalized approach Popularity and probabilistic graphical model COM. This demonstrates the superiority of neural networks, especially their great ability in modeling the high-order interactions among users, groups, and items. 3) There is no obvious winner among the score aggregation-based solutions. For example, NCF+avg outperforms NCF+lm when $K = 5$ on Mafengwo, but underperforms when $K = 10$. This again confirms that a predefined, static score aggregation strategy is insufficient to predict the group decision well. In contrast, AGREE dynamically associates weights for group members by learning from data, which shows remarkable flexibility and superiority.

3.4 Importance of Components (RQ3)

The overall performance comparison shows that AGREE obtains the best results, demonstrating the effectiveness of the integrated end-to-end solution. To further understand the importance of user embedding aggregation and group preference embedding in learning group representation, we performed some ablation studies. For convenience, we use the name AGREE-U to denote the method "AGREE with user embedding aggregation only", and AGREE-G to denote "AGREE with group preference embedding only" (which is equivalent with the NCF method).

Table 4 and Table 5 show the results of AGREE and the two simplified variants. We have the following observations: 1) AGREE consistently and significantly outperforms AGREE-U and AGREE-G on both datasets with respect to both metrics, which can be evidenced by the small p-values. This indicates that both

Table 2: Top-K performance of both recommendation tasks for users and groups on Mafengwo (Section 3.3).

Overall Performance Comparison (Mafengwo)												
	K=5						K=10					
	User			Group			User			Group		
	HR	NDCG	p-value	HR	NDCG	p-value	HR	NDCG	p-value	HR	NDCG	p-value
NCF	0.6363	0.5432	4.46e-06	0.4291	0.3405	7.18e-09	0.7417	0.5733	3.68e-05	0.6181	0.4020	3.95e-08
Popularity	0.4047	0.2876	2.02e-12	0.3115	0.2169	1.55e-11	0.4971	0.3172	2.09e-12	0.4251	0.2537	1.13e-11
COM	—	—	—	0.4420	0.3297	6.54e-09	—	—	—	0.5434	0.3727	1.36e-09
GREE	0.6306	0.5395	7.87e-07	0.4513	0.3577	1.20e-07	0.7206	0.5687	1.74e-06	0.6151	0.4111	3.25e-07
NCF+avg	—	—	—	0.4774	0.3669	2.86e-06	—	—	—	0.6222	0.4140	8.84e-07
NCF+lm	—	—	—	0.4744	0.3631	5.67e-07	—	—	—	0.6302	0.4152	1.45e-06
NCF+ms	—	—	—	0.4700	0.3616	3.46e-07	—	—	—	0.6281	0.4114	3.57e-07
NCF+exp	—	—	—	0.4724	0.3647	1.03e-06	—	—	—	0.6251	0.4015	3.61e-08
AGREE	**0.6383**	**0.5502**	—	**0.4814**	**0.3747**	—	**0.7491**	**0.5775**	—	**0.6400**	**0.4244**	—

Table 3: Top-K performance of both recommendation tasks for users and groups on CAMRa2011 (Section 3.3).

Overall Performance Comparison (CAMRa2011)												
	K=5						K=10					
	User			Group			User			Group		
	HR	NDCG	p-value	HR	NDCG	p-value	HR	NDCG	p-value	HR	NDCG	p-value
NCF	0.6119	0.4018	1.03e-06	0.5803	0.3896	9.02e-06	0.7894	0.4535	1.89e-07	0.7593	0.4448	3.92e-07
Popularity	0.4624	0.3104	9.15e-11	0.4324	0.2825	5.92e-11	0.6026	0.3560	5.99e-11	0.5793	0.3302	3.67e-11
COM	—	—	—	0.5793	0.3762	7.20e-08	—	—	—	0.7682	0.4368	1.46e-17
GREE	0.6163	0.4079	5.02e-05	0.5883	0.3871	2.11e-06	0.7841	0.4571	5.67e-07	0.7690	0.4479	5.43e-08
NCF+avg	—	—	—	0.5683	0.3819	2.97e-07	—	—	—	0.7641	0.4452	4.47e-07
NCF+lm	—	—	—	0.5593	0.3788	1.29e-07	—	—	—	0.7648	0.4455	4.94e-07
NCF+ms	—	—	—	0.5434	0.3710	2.75e-08	—	—	—	0.7607	0.4348	3.74e-08
NCF+exp	—	—	—	0.5648	0.3787	1.26e-07	—	—	—	0.7621	0.4426	2.05e-07
AGREE	**0.6223**	**0.4118**	—	**0.5883**	**0.3955**	—	**0.7967**	**0.4687**	—	**0.7807**	**0.4575**	—

Table 4: Top-K performance of AGREE and its two simplified variants on the Mafengwo dataset (Section 3.4).

Component Performance Comparison (Mafengwo)												
	K=5						K=10					
	User			Group			User			Group		
	HR	NDCG	p-value	HR	NDCG	p-value	HR	NDCG	p-value	HR	NDCG	p-value
AGREE-U	0.6220	0.5364	2.80e-07	0.4141	0.3322	2.99e-09	0.7309	0.5716	9.02e-06	0.5709	0.3832	3.39e-09
AGREE-G	0.6363	0.5432	4.46e-06	0.4291	0.3405	7.18e-09	0.7417	0.5733	3.68e-05	0.6181	0.4020	3.95e-08
AGREE	**0.6383**	**0.5502**	—	**0.4814**	**0.3747**	—	**0.7491**	**0.5775**	—	**0.6400**	**0.4244**	—

Table 5: Top-K performance of AGREE and its two simplified variants on the CAMRa2011 dataset (Section 3.4).

Component Performance Comparison (CAMRa2011)												
	K=5						K=10					
	User			Group			User			Group		
	HR	NDCG	p-value	HR	NDCG	p-value	HR	NDCG	p-value	HR	NDCG	p-value
AGREE-U	0.6043	0.3945	1.12e-07	0.5793	0.3832	4.47e-07	0.7601	0.4465	4.09e-08	0.7441	0.4376	6.36e-08
AGREE-G	0.6119	0.4018	1.03e-06	0.5803	0.3896	9.02e-06	0.7894	0.4535	1.89e-07	0.7593	0.4448	3.92e-07
AGREE	**0.6223**	**0.4118**	—	**0.5883**	**0.3955**	—	**0.7967**	**0.4687**	—	**0.7807**	**0.4575**	—

components of user embedding aggregation and group preference embedding are beneficial to model group decisions, and combining them leads to better performance. 2) AGREE-G shows better performance than AGREE-U on both datasets. This reveals that the group preference embedding has a larger impact in learning group representation in our method.

3.5 Handling New-User Cold-Start Issue (RQ4)

We consider addressing the new-user cold-start problem, where users have participated in groups but never consumed items individually. Since user embeddings are shared in AGREE for learning group-item and user-item interactions, the embeddings of users can be well learnt from the group-item interactions even the

Table 6: Top-K performance of the new-user cold-start scenario on Mafengwo (Section 3.5).

New-User Cold-Start (Mafengwo)						
	K=5			K=10		
	HR	NDCG	p-value	HR	NDCG	p-value
Popularity	0.3115	0.2169	1.22e-12	0.4251	0.2537	8.18e-13
NCF+group	0.6576	0.4687	2.20e-09	0.7716	0.5512	9.62e-09
AGREE	**0.6989**	**0.5146**	—	**0.8013**	**0.5830**	—

users have no user-item interactions. In fact, the average number of groups joined by a user on the Mafengwo dataset is 1.36, while that of the CAMRa2011 dataset is 1.00. Therefore, the CAMRa2011 dataset is not suitable for this scenario since we know few about the user if she only participates in one group, and we conducted the experiment on the Mafengwo dataset by selecting the users who participate as least two groups. The experiment setting is illustrated in Section 3.1.2. We compared our AGREE model with two methods 1) Popularity, and 2) NCF+group, which enriches the user-item interaction data by assuming that a user has consumed the historical items of her participated groups; then NCF is applied on the enriched data.

The results are shown in Table 6. The conclusions are twofold: 1) AGREE demonstrates significant improvements over Popularity and NCF+group. Specifically, the improvements over NCF+group are 6.28% in HR and 9.79% in NDCG when $K = 5$, 3.85% in HR and 5.77% in NDCG when $K = 10$. This indicates the significance of AGREE in addressing the cold-start issue by leveraging users' group activities. 2) AGREE and NCF+group achieve preferable results as compared with Popularity, which verifies that personalized methods are superior to non-personalized method even for the cold-start setting.

4 RELATED WORK

4.1 Group Recommendation

Group recommendation has received great attentions in recent years and has been widely applied in various domains. The techniques for handling the group recommendation can be divided into two categories — *memory-based* and *model-based* approaches. Memory-based approaches can be further subdivided into *preferences aggregation* and *score aggregation* [41]. Preferences aggregation strategy first aggregates the profiles of group members into a new profile, and then employs recommendation techniques designed for individuals to make group recommendation. Score aggregation strategy first predicts the individuals' scores over candidate items, and then aggregates the predicted scores of members in a group via predefined strategies (e.g., average, least misery, maximum satisfaction and so on) to represent the group's preferences. However, both approaches are predefined and inflexible, which utilize trivial methods to aggregate members' preferences. Different from memory-based approaches, model-based methods exploit the interactions among members by modeling the generative process of a group. The PIT model [30] effectively identifies the group preference profile for a given group by considering the personal preferences and personal impacts of group members. A probabilistic model named COM [41] is proposed to model the generative process of group activities and make group recommendations. The most similar method to our algorithm is a deep learning approach, DLGR [20], which learns high-level comprehensive features to represent group preference so as to avoid the vulnerabilities in a shallow representation. Our work falls into the model-based category. Except for learning the high-level interactions among users, groups, and items under the deep learning framework, our work employs the attention mechanism as the underlying principle for the aggregation of users' embedding representations. Meanwhile, user-item and group-item recommendations are mutually reinforced under our framework.

4.2 Deep Learning for Recommendation

The proliferation of deep learning has swept the research community, among which recommender systems are no exception. The majority of work that integrates recommender systems with deep learning methods primarily utilized deep neural networks for modeling auxiliary information. The features learnt by deep neural networks are then incorporated into collaborative filtering algorithms. However, these work somehow falls into a two-stage mode, in which recommender systems and deep learning methods are implemented separately. Different from previous work, there are some attempts that try to seamlessly combine recommender systems with deep learning methods by modeling user-item interactions [18] and higher-order feature interactions [17] with non-linear neural networks. The success of NCF [18] has been further applied to the cross-domain social recommendation [39] and is utilized as the foundation of our work as well.

Attention mechanism with the realization of neural networks has been shown practical in machine learning tasks. It simulates human recognition by focusing on some selective parts of the whole image or the whole sentence while ignoring some other informative (less informative) parts. In fact, the attention mechanism has been investigated in the field of recommender systems. Regarding the multimedia retrieval [24, 31, 32] and its applications [25, 29, 36], attentive collaborative filtering [9] introduced the item- and component-level attention model for multimedia recommendation. In attentional factorization machines [40], the weights of feature interactions are learnt via neural attention network. For microblog hashtag recommendation, a co-attention network was proposed [43] which generates textual attention and visual attention sequentially. Moreover, an attention- and recurrent neural network-based deep architecture was proposed [42] for modeling queries and advertisements in online advertising. Inspired by these pioneering work, the key idea of our AGREE model is to regard a group as an image or a sentence and learn to assign attention weights for members (components) in the group (image or sentence): higher weights indicate that the corresponding members (components) are significant to the end task (image or sentence).

5 CONCLUSION AND FUTURE WORK

In this work, we address the group recommendation problem from the perspective of neural representation learning. Under the framework, there are two key factors to estimate a group's preference on an item well: 1) how to obtain a semantic representation for a group, and 2) how to model the interaction between a group and an item. We propose a novel solution named AGREE, which addresses the first factor of group representation learning with an attention network and the second factor of interaction learning with NCF. Specifically, by leveraging the attention network, AGREE can automatically learn the importance of a group member from data; by leveraging NCF, it is capable of learning the complicated interactions among groups, users, and items. Furthermore, we integrate the modeling of user-item interaction data into AGREE, allowing the two tasks of recommending items for groups and users to be mutually reinforced. To validate the effectiveness of AGREE, we perform extensive experiments on two real-world datasets. The results show that AGREE achieves state-of-the-art performance for group recommendation; further micro-level analyses demonstrate how the attention network improves the performance, and how components of AGREE affect the results, and how AGREE alleviates the new-user cold-start issue.

In future, we plan to extend our work in the following two directions. First, we will explore the utility of social network [11, 23, 37], since it contains valuable signal on how users form a group and how users trust each other. Second, we are interested in realizing group recommender systems in an online fashion. The interests of users evolve over time, and so do the preferences of groups. As it is computationally prohibitive to retrain a recommender model in real-time, it would be extremely helpful to do online learning [10, 21, 22]. Along this line, we are particularly interested in leveraging reinforcement learning methods to provide online recommendation.

6 ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No. 61702176) and the Hunan Provincial Natural Science Foundation of China (No.2017JJ3038).

REFERENCES

[1] Sihem Amer-Yahia, Senjuti Basu Roy, Ashish Chawlat, Gautam Das, and Cong Yu. 2009. Group recommendation: Semantics and efficiency. *VLDB* 2, 1 (2009), 754–765.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.

[3] Linas Baltrunas, Tadas Makcinskas, and Francesco Ricci. 2010. Group recommendations with rank aggregation and collaborative filtering. In *RecSys*. 119–126.

[4] Shlomo Berkovsky and Jill Freyne. 2010. Group-based recipe recommendations: analysis of data aggregation strategies. In *RecSys*. 111–118.

[5] Ludovico Boratto and Salvatore Carta. 2010. State-of-the-art in group recommendation and new approaches for automatic identification of groups. In *Information retrieval and mining in distributed environments*. 1–20.

[6] Da Cao, Xiangnan He, Liqiang Nie, Xiaochi Wei, Xia Hu, Shunxiang Wu, and Tat-Seng Chua. 2017. Cross-platform app recommendation by jointly modeling ratings and texts. *TOIS* 35, 4 (2017), 37.

[7] Da Cao, Liqiang Nie, Xiangnan He, Xiaochi Wei, Jialie Shen, Shunxiang Wu, and Tat-Seng Chua. 2017. Version-sensitive mobile App recommendation. *INS* 381 (2017), 161–175.

[8] Da Cao, Liqiang Nie, Xiangnan He, Xiaochi Wei, Shunzhi Zhu, and Tat-Seng Chua. 2017. Embedding Factorization Models for Jointly Recommending Items and User Generated Lists. In *SIGIR*. 585–594.

[9] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Wei Liu, Wei Liu, and Tat-Seng Chua. 2017. Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention. In *SIGIR*. 335–344.

[10] Zhiyong Cheng, Shen Jialie, and Steven CH Hoi. 2016. On effective personalized music retrieval by exploring online user behaviors. In *SIGIR*. 125–134.

[11] Zhiyong Cheng and Jialie Shen. 2014. Just-for-me: An adaptive personalization system for location-aware social music recommendation. In *ICML*. 185.

[12] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *RecSys*. 191–198.

[13] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *RecSys*. 39–46.

[14] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. In *WWW*. 278–288.

[15] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *JMLR* 11, 3 (2010), 625–660.

[16] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. *JMLR* 9 (2010), 249–256.

[17] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *SIGIR*. 355–364.

[18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*.

[19] Xiangnan He, Hanwang Zhang, Min Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *SIGIR*. 549–558.

[20] Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Wei Cao. 2014. Deep Modeling of Group Preferences for Group-Based Recommendation.. In *AAAI*. 1861–1867.

[21] Wenjun Jiang, Guojun Wang, Md Zakirul Alam Bhuiyan, and Jie Wu. 2016. Understanding graph-based trust evaluation in online social networks: Methodologies and challenges. *Comput. Surveys* 49, 1 (2016), 10.

[22] Wenjun Jiang and Jie Wu. 2017. Active opinion-formation in online social networks. In *INFOCOM*. 1–9.

[23] Wenjun Jiang, Jie Wu, Feng Li, Guojun Wang, and Huanyang Zheng. 2016. Trust evaluation in online social networks using generalized network flow. *TOC* 65, 3 (2016), 952–963.

[24] Peiguang Jing, Yuting Su, Liqiang Nie, Xu Bai, Jing Liu, and Meng Wang. 2017. Low-rank Multi-view Embedding Learning for Micro-video Popularity Prediction. *TKDE* (2017).

[25] Peiguang Jing, Yuting Su, Liqiang Nie, and Huimin Gu. 2017. Predicting Image Memorability Through Adaptive Transfer Learning From External Sources. *TMM* 19, 5 (2017), 1050–1062.

[26] Santosh Kabbur, Xia Ning, and George Karypis. 2013. FISM: Factored Item Similarity Models for top-N Recommender Systems. In *KDD*. 659–667.

[27] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *Computer Science* (2014).

[28] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD*. 426–434.

[29] Meng Liu, Liqiang Nie, Meng Wang, and Baoquan Chen. 2017. Towards Micro-video Understanding by Joint Sequential-Sparse Modeling. In *MM*. 970–978.

[30] Xingjie Liu, Yuan Tian, Mao Ye, and Wang-Chien Lee. 2012. Exploring personal impact for group recommendation. In *CIKM*. 674–683.

[31] Liqiang Nie, Meng Wang, Yue Gao, Zheng-Jun Zha, and Tat-Seng Chua. 2013. Beyond text QA: multimedia answer generation by harvesting web information. *TMM* 15, 2 (2013), 426–441.

[32] Liqiang Nie, Luming Zhang, Yi Yang, Meng Wang, Richang Hong, and Tat-Seng Chua. 2015. Beyond doctors: Future health prediction from multimedia and multimodal observations. In *MM*. ACM, 591–600.

[33] Elisa Quintarelli, Emanuele Rabosio, and Letizia Tanca. 2016. Recommending new items to ephemeral groups using contextual user influence. In *RecSys*. 285–292.

[34] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. 452–461.

[35] Shunichi Seko, Takashi Yagi, Manabu Motegi, and Shinyo Muto. 2011. Group recommendation using feature space representing behavioral tendency and power balance among members. In *RecSys*. 101–108.

[36] Xuemeng Song, Fuli Feng, Jinhuan Liu, Zekun Li, Liqiang Nie, and Jun Ma. 2017. NeuroStylist: Neural Compatibility Modeling for Clothing Matching. In *MM*. 753–761.

[37] Xuemeng Song, Liqiang Nie, Luming Zhang, Mohammad Akbari, and Tat-Seng Chua. 2015. Multiple social network learning and its application in volunteerism tendency prediction. In *SIGIR*. 213–222.

[38] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *JMLR* 15, 1 (2014), 1929–1958.

[39] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item Silk Road: Recommending Items from Information Domains to Social Users. In *SIGIR*. 185–194.

[40] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks. In *IJCAI*. 3119–3125.

[41] Quan Yuan, Gao Cong, and Chin-Yew Lin. 2014. COM: a generative model for group recommendation. In *SIGKDD*. 163–172.

[42] Shuangfei Zhai, Keng-hao Chang, Ruofei Zhang, and Zhongfei Mark Zhang. 2016. Deepintent: Learning attentions for online advertising with recurrent neural networks. In *SIGKDD*. 1295–1304.

[43] Qi Zhang, Jiawen Wang, Haoran Huang, Xuanjing Huang, and Yeyun Gong. 2017. Hashtag Recommendation for Multimodal Microblog Using Co-Attention Network. In *IJCAI*. 3420–3426.