Ranking-based Group Identification via Factorized Attention on Social Tripartite Graph

Mingdai Yang
University of Illinois at Chicago
Chicago, USA
myang72@uic.edu

Zhiwei Liu
Salesforce AI Research
Palo Alto, USA
zhiweiliu@salesforce.com

Liangwei Yang
Xiaolong Liu
lyang84@uic.edu
xliu262@uic.edu
University of Illinois at Chicago
Chicago, USA

Chen Wang
University of Illinois at Chicago
Chicago, USA
cwang266@uic.edu

Hao Peng*
School of Cyber Science and
Technology, Beihang University,
Beijing, China
penghao@buaa.edu.cn

Philip S. Yu
University of Illinois at Chicago
Chicago, USA
psyu@uic.edu

ABSTRACT

Due to the proliferation of social media, a growing number of users search for and join group activities in their daily life. This develops a need for the study on the ranking-based group identification (RGI) task, i.e., recommending groups to users. The major challenge in this task is how to effectively and efficiently leverage both the item interaction and group participation of users' online behaviors. Though recent developments of Graph Neural Networks (GNNs) succeed in simultaneously aggregating both social and user-item interaction, they however fail to comprehensively resolve this RGI task. In this paper, we propose a novel GNN-based framework named **C**ontextualized **F**actorized **A**ttention for **G**roup identification (CFAG). We devise tripartite graph convolution layers to aggregate information from different types of neighborhoods among users, groups, and items. To cope with the data sparsity issue, we devise a novel propagation augmentation (PA) layer, which is based on our proposed factorized attention mechanism. PA layers efficiently learn the relatedness of non-neighbor nodes to improve the information propagation to users. Experimental results on three benchmark datasets verify the superiority of CFAG. Additional detailed investigations are conducted to demonstrate the effectiveness of the proposed framework.

CCS CONCEPTS

• **Information systems → Collaborative filtering**.

*Corresponding author

KEYWORDS

Recommender System; Graph Neural Network; Tripartite Graph; Group Recommendation

1 INTRODUCTION

Given the difficulties in information collection and decision-making among the overwhelming options for individuals, a growing number of customers prefer to join specific groups for suggestions in advance of consumption. For example, on the Steam video game platform[1], if a video game player hesitates about whether she should purchase a newly released game, she would seek suggestions from a group of players who have played that game for a period. Online groups offer spaces for users to share experiences, which in turn provides a reference to other group members. It assists them in locating their demands accurately and even influences their interests in items. We illustrate a toy example in Figure 1, Alex is interested in Motorcycle. Hence, he would like to join a Motorcyclists group. In terms of platforms, users' attachment to online groups can significantly increase the participation and retention rate of their users [38]. Compared to promoting content directly to potential users [25], recommending groups to users based on their interests is a more feasible way to help platforms build emotional bonds with users for maintaining long-term stickiness, which is however under-explored.

Most existing recommender systems have been widely adopted to discover relevant content [22], products [13] or services [16]. The recent successes of GNNs [11, 27] inspire graph-based recommender systems [14, 31]. These graph-based recommender systems
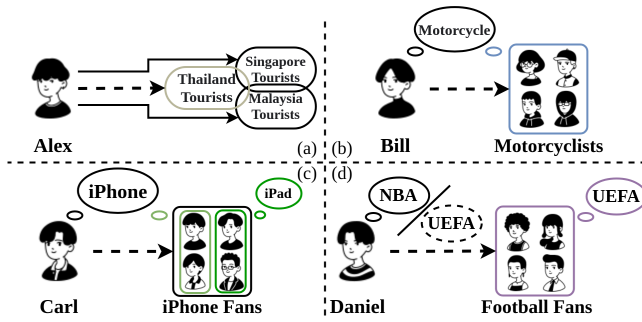
[1]https://store.steampowered.com/

Figure 1: Illustration of group identification.

focus on bipartite recommendation tasks, e.g., user-item recommendation [14, 31, 33]. These methods allow information to propagate over graphs through high-order connectivity.

However, directly applying those bipartite graph-based approaches to user-group interactions for prediction is far from comprehensive. In this paper, we argue that successfully recommending groups to users requires the combined modeling of both group participation and item interactions. On one hand, users may be interested in a group similar to the groups they joined before. We demonstrate this via a toy example in Figure 1(a). Alex visited Singapore and Malaysia before and became a member of corresponding tourist groups. Therefore, he would like to join a tourist group in Thailand as all of them are in Southeast Asia. On the other hand, users may join a new group because of item interests, even though never participating in any relevant groups before. For example, in Figure 1, Bill is fond of motorcycles. As such, we should recommend him a motorcycle riders group as Figure 1(b) demonstrated. Therefore, only when simultaneously characterizing both group participation and item interactions can we develop a satisfying group identification system.

In this work, we define recommending groups to users as ranking-based group identification (RGI) problem. RGI problem is distinguishable from group recommendation [1, 24, 37] and community detection [8, 17], where the former is to recommend items to groups and/or users by leveraging group information while the latter is to cluster users. As aforementioned, incorporating both item interactions and group participation is necessary for RGI problem, which is rather challenging from two perspectives.

Firstly, we should consider high-order connectivity in RGI problem. We demonstrate the high-order connectivity in Figure 1(c), where Carl is considering buying a newly-released iPhone. Thus, he joins the *iPhone Fans* group to discuss with others. In this group, other members also have interactions with *iPad*. As such, Carl may also have potential interests in groups relevant to iPad. Though Carl has no direct connection with iPad, we can still discover this potential interest through high-order connections, *i.e.*, user-group-user-item-group in this case. Existing works on recommender systems harness additional preferences (*i.e.* item interactions in this paper) as side information [1, 6, 12], which is unable to resolve high-order connectivity.

Secondly, the relatedness of items is crucial for identifying users' group interests. To be more specific, a user may be interested in a

group if its members share similar items' interests with this user, even though not exactly the same item that this user has interactions with. We illustrate this in Figure 1(d). Daniel subscribes to NBA basketball games stream[2]. He would be interested in a Football Fans group where the members are all football fans having UEFA football games stream[3] subscriptions as a result of the relatedness between NBA and UEFA.

To this end, we investigate the RGI problem upon a social tripartite graph, which includes user, group, and item as nodes and their associated interactions as edges. We decide to propagate information over this graph to learn representations of nodes. The designing reasons are threefold. First, this tripartite graph incorporates both group participation and item interactions. Second, propagating information over a social tripartite graph enables the high-order connectivity characterization. Third, similar representations are able to reveal the relatedness of RGI problem.

However, directly adopting existing methods for the RGI problem is not suitable since we observe severe sparsity issues. In other words, many users have few group/item interactions. Under this circumstance, aggregation over the tripartite graph is less likely to learn high-quality representations of nodes due to the unavailability of adequate propagation paths.

To this end, we propose a novel framework, named Contextualized Factorized Attention for Group identification (CFAG). CFAG is a GNN over our proposed social tripartite graph. To resolve the sparsity issue, we propose to endow the model with the ability of discovering more potential interactions of users during training via factorized attention mechanism. More concretely, we propose learning weights between users and *all* other groups/items that they have no interaction with. In this way, the information from potential groups and items is able to propagate to users. The weights reflect the potential impacts of those groups/items and construct the propagation paths. Nonetheless, training all the pair-wise weights is of high time- and space- complexity. Instead, we learn contextual embeddings for both groups and items. Then, the weights between a user and all candidate groups and items are inferred based on the embedding similarity between candidates and users' participated groups and interacted items, respectively. Finally, the weights between this user and candidates are calculated as the attention score. To train a CFAG, besides the contextual embedding for items and groups, all nodes have personalized embeddings. We iteratively update the personalized embedding and contextual embedding via propagation over the graph. The rankings score between a user and a group is based on their final personalized embeddings from the output layer. And we optimize the framework based on BPR loss [23]. We highlight our key contributions as follows:

- We propose a novel framework CFAG, a GNN-based model for group identification, which can propagate the information on the social tripartite graph and conduct recommendation.
- We devise novel propagation augmentation layers with factorized attention mechanism in CFAG to cope with the sparsity issue, which explores non-existing interactions and enhances the propagation ability on graphs with high sparsity.

---

[2]https://www.espn.com/nba/
[3]https://www.uefa.com/uefaeuropaleague/

- We collect and release one large dataset for RGI task. We conduct extensive experiments on this dataset and two other public datasets. The significant improvements of CFAG on all datasets indicate its superiority in tackling the RGI problem.

2 RELATED WORKS

Since there is no previous work targeting the exact same task, i.e., recommending groups to users based on graphs, we will introduce some closely related work: (1) Community detection, which shares similar goals with our task, is introduced in Section 2.1. (2) Recommender systems that utilize users' social information are reviewed in Section 2.2. (3) With similar deep learning techniques, GNN-based methods are introduced in Section 2.3. In each subsection, we discuss their relationships to the proposed RGI task.

2.1 Group Recommender Systems

Group recommender systems refer to recommending groups to their potential members. Traditional group recommender systems apply various algorithms to recover user-group membership matrices with available side information. For example, semantic information from descriptions of groups [3] and visual information from photos shared by users [28] can be incorporated with a collaborative filtering framework to perform personalized group recommendations. User behaviors in different time periods [30, 34], such as joining groups, can also be leveraged for recommending groups to users. However, the requirement for side information degrades the performance of those methods when recommending groups to users with only interaction information. Some recent works [18, 19] investigate recommending groups to users with only user-group interactions. They directly characterize the bipartite structure between users and groups, while item interaction information is ignored.

Besides recommending groups to users, the term *group recommendation* in literature also refers to recommending items to a group of users [1, 24], which differs from the focus in this paper.

2.2 Social Information-based RS

Social relations between users have been applied to the recommender system to alleviate the data sparsity problem [15, 35]. Research in social recommendation combines a user-user graph and the bipartite user-item graph to better understand users' preferences on items [10, 36]. Friend recommendation is another research topic based on the homogeneous graph formed by social relations to find possible social links between users [5, 9]. For these RS involving social relations among users, the main challenge lies in how to model the influence between users [29]. Typical approaches addressing this challenge contain random walk [7], GAT [6] and graph embedding [32].

In contrast to using social information to predict possible links among user-user pairs or user-item pairs, our work focuses on predicting links between users and groups. Different from user-user or user-item relation, user-group relation could be entangled promiscuously [1, 37].

2.3 Graph-Based RS

GNN has been widely leveraged to address the most important challenges in RS nowadays given its powerful capability of learning informative representations in graph data. The GNN-based methods broadly fall into three classes from the model perspective: (i) Graph Convolutional Network based RS [14, 31, 33]; (ii) Graph Attention Network based RS [26, 27]; and (iii) Gated Graph Neural Network based RS [4].

Most previous works on Graph-based RS only focus on user-item bipartite graph [14]. It is difficult to directly apply these works to group identification tasks with three relations to be managed: user-item, group-item and user-group relations.

Some graph-based works in group-item recommendation also model different interactions between users, items and groups [1]. However, the key challenge of those works is how to aggregate the preference of group members [1, 24]. Our task differs significantly in that our goal is to predict the preference of individual users on groups he/she never interacted with.

3 PRELIMINARIES

In this section, we first define the social tripartite graph and then formulate the problem of rank-based group identification (RGI).

DEFINITION 1. **(Social Tripartite Graph).** *Given three disjoint nodes, i.e, user nodes set $\mathcal{U}$, group nodes set $\mathcal{G}$, and item nodes set $\mathcal{I}$, and given their interactive edges, i.e. user-group edges $E_{\mathcal{U},\mathcal{G}}$, user-item edges $E_{\mathcal{U},\mathcal{I}}$, and group-item edges $E_{\mathcal{G},\mathcal{I}}$, we define a social tripartite graph as $\mathcal{T} = \{V, E\}$, where $V = \mathcal{U} \cup \mathcal{I} \cup \mathcal{G}$ and $E = E_{\mathcal{U},\mathcal{G}} \cup E_{\mathcal{U},\mathcal{I}} \cup E_{\mathcal{G},\mathcal{I}}$.*

The social tripartite graph is an undirected and unweighted heterogeneous graph. In the following sections, we denote user nodes as u, group nodes as g, and item nodes as i. An edge $(u_j, g_k) \in E_{\mathcal{U},\mathcal{G}}$ in the graph represents that user $u_j \in \mathcal{U}$ is a member of a group $g_k \in \mathcal{G}$. Similarly, an edge in $E_{\mathcal{U},\mathcal{I}}$ or $E_{\mathcal{G},\mathcal{I}}$ represents that a user or a group shows preference on an item. Note that if we have no item set available, *i.e.* $\mathcal{I} = \emptyset$, this tripartite graph degenerates to a bipartite graph, which is utilized in existing recommender systems.

During computation, we use adjacency matrices to represent the edges. Specifically, we define the user-group adjacency matrix $\mathbf{X} = [x_{jk}] \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{G}|}$ to represent the user-group interactions, where $x_{jk} = 1$ if an edge (u_j, g_k) exists in the social tripartite graph, i.e., user u_j is a member of group g_k, and otherwise $x_{jk} = 0$. Analogously, we define user-item and group-item adjacency matrices as $\mathbf{Y} \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{I}|}$ and $\mathbf{Z} \in \mathbb{R}^{|\mathcal{G}| \times |\mathcal{I}|}$, respectively. Then, the target is to conduct ranking-based edge prediction, which is defined as:

DEFINITION 2. **(Ranking-based Group Identification).** *Given a social tripartite graph $\mathcal{T}$, the ranking-based group identification (RGI) for a user u is to predict a ranking list of groups $\{g_1, g_2, \ldots, g_k\}$, with which this user has no interactions.*

In other words, we recommend a list of groups that this user u is of potential interest in RGI. Note that we distinguish the group as another entity rather than a simple union of users due to its special characteristics, e.g. group information.

4 METHOD

In this section, we present the proposed CFAG model for the group identification task. The framework of CFAG is shown in Figure 2.
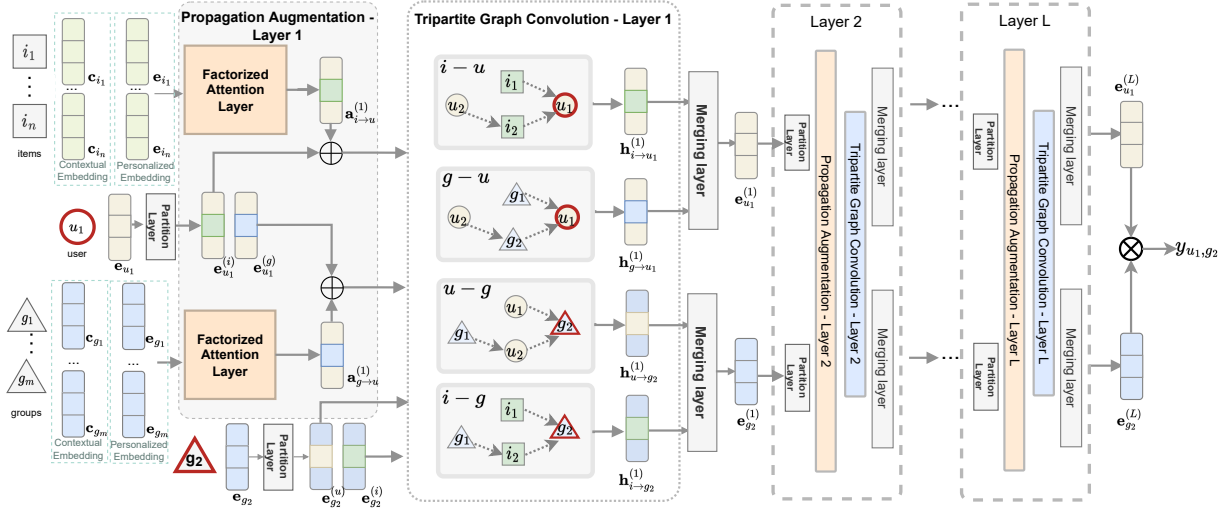
Figure 2: The framework of CFAG. This is the computation process when inferring the score between user u_1 and group g_2.

We start by introducing all embedding layers to be trained in this framework. Next, we demonstrate how to aggregate the interactions from different neighborhoods on the social tripartite graph. Specifically, we adopt tripartite graph convolution networks to learn the user and group personalized embeddings for recommending groups. We further propose a Factorized Attention module for user-item interactions and user-group interactions to infer the relevance of each group and each item to the target user.

4.1 Embedding layer

We maintain an embedding layer $\mathbf{E} \in \mathbb{R}^{d \times (|\mathcal{U}| + |\mathcal{G}| + |\mathcal{I}|)}$, where each column represents the trainable personalized embedding for each node $v \in V$ in the graph. In addition to the personalized embedding, we also have contextual embedding layers for both items and groups, denoted as $\mathbf{C}_i \in \mathbb{R}^{d \times |\mathcal{I}|}$ and $\mathbf{C}_g \in \mathbb{R}^{d \times |\mathcal{G}|}$, respectively. They are used and trained for Factorized Attention to infer the influence between groups (items) for each user, respectively. More details will be presented in Sec. 4.3.

4.2 Tripartite Graph Convolution

GNNs learn node embeddings by propagating information from neighbors to center nodes. However, existing GNN layers are not suitable due to the heterogeneity of the social tripartite graph, i.e. the distinctions of different types of nodes. Hence, we devise a novel tripartite graph convolution to propagate information between users, groups and items. Instead of directly aggregating embeddings of neighbors, we employ *partition layers* to divide neighbor information as two branches for aggregation. In the following part of this section, we explain each module of tripartite graph convolution over the center node group g. The tripartite graph convolution over user and items can be derived in analogy. To be more specific, partition layers divide group information $\mathbf{e}_g$ as group-user information $\mathbf{e}_g^{(u)}$ and group-item information $\mathbf{e}_g^{(i)}$ for the following

propagation to users and items, respectively:

$$\mathbf{e}_g^{(u)}, \mathbf{e}_g^{(i)} = \mathbf{PT}(\mathbf{e}_g), \quad (1)$$

where $\mathbf{PT}()$ denotes the partition layer. We justify various partition layers, for example $\mathbf{PT}()$ can be two linear transformations, i.e., $\mathbf{e}_g^{(u)} = \mathbf{W}_g^{(u)} \mathbf{e}_g$, $\mathbf{e}_g^{(i)} = \mathbf{W}_g^{(i)} \mathbf{e}_g$. Moreover, it can be a simple division over the dimension, i.e., $[\mathbf{e}_g^{(u)} \| \mathbf{e}_g^{(i)}] = \mathbf{e}_g$, where $\|$ is the concatenation of two embeddings. In fact, experiments in Sec. 5.3 justify the superiority of later simple division over the dimension. The partition layers distinguish the impacts of different types of neighbors. Examples of dividing the information of group g_2 and user u_1 via partition layers are illustrated in Figure 2.

Hereafter, we learn node embeddings by aggregating corresponding neighboring information from partition layers. The group information aggregated from both neighbor users and items are as follows:

$$\begin{aligned} \mathbf{h}_{u \to g} &= \mathbf{AGG}(\{\mathbf{e}_u^{(g)} | u \in \mathcal{N}_g^{(u)}\}) \\ \mathbf{h}_{i \to g} &= \mathbf{AGG}(\{\mathbf{e}_i^{(g)} | i \in \mathcal{N}_g^{(i)}\}), \end{aligned} \quad (2)$$

where $\mathbf{h}_{u \to g}$ and $\mathbf{h}_{i \to g}$ denote the user and item information propagated to group g, respectively. The $\mathbf{AGG}()$ represents the aggregation layers, such as the GCN [21] aggregation. These aggregation layers propagate associated information to group g. For example, user-to-group information $\mathbf{h}_{u \to g}$ is aggregated from the group partition of user embeddings, denoted as $\mathbf{e}_u^{(g)}$. $\mathcal{N}_g^{(u)}$ represents all the neighbor users of group g. In analogy, item-to-group information $\mathbf{h}_{i \to g}$ is aggregated from the group partition of neighbor item embeddings, i.e., $\mathbf{e}_i^{(g)}$.

Next, we combine the information from two branches as one embedding for group g via *merging layers* $\mathbf{MG}()$ as follows:

$$\mathbf{e}_g^* = \mathbf{MG}(\mathbf{h}_{u \to g}, \ \mathbf{h}_{i \to g}), \quad (3)$$

where $\mathbf{e}_g^*$ is the output embedding for group g. We demonstrate the merging layers in Figure 2. In this paper, we investigate different types of $\mathbf{MG}(\cdot, \cdot)$ layers, including direct concatenation,
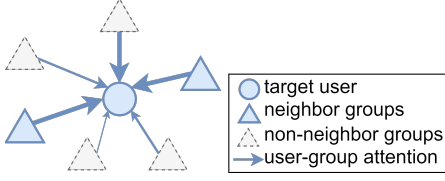
Figure 3: Illustration of propagation augmentation on group-user neighbors. We aggregate all groups via learned group contextualization attention.

concatenation before fully-connection, and concatenation after fully-connection.

Multi-layer propagation. By stacking multiple those layers in Eq. (1), (2) and (3), we construct the multi-layer propagation pattern of tripartite graph convolution and rewrite the Eq. (3) as follows:

$$\mathbf{e}_g^{(l)} = \mathbf{MG}(\mathbf{h}_{u \to g}^{(l)}, \ \mathbf{h}_{i \to g}^{(l)}), \tag{4}$$

where $\mathbf{h}_{u \to g}^{(l)}$ and $\mathbf{h}_{i \to g}^{(l)}$ denote the information propagation in Eq. (2) on the l-th layer. In Figure 2, we stack L layers and yield the final embedding $\mathbf{e}_{g_2}^{(L)}$ and $\mathbf{e}_{u_1}^{(L)}$ for group g_2 and user u_1, respectively.

4.3 Propagation Augmentation

In RGI problem, we observe severe sparsity issues in the graph, *e.g.* few group participation for users. Those sparsity issues impair the propagation over the graph, and thus, those nodes with few neighbors are unable to be well-trained. To this end, we propose a propagation augmentation (PA) layer before the tripartite graph convolution. Intuitively, PA layers aggregate the information from all *non-neighbor* nodes to the center node. As such, it augments the propagation paths for sparse graphs. An illustration of the PA layer on group-to-user propagation is in Figure 3. The dash triangles represent those unconnected groups of the target user. We assign them the user-group attention weights as in the solid lines. PA layer constructs additional context for the target user by exploring non-neighbor groups. Hence, we have additional user-to-group information as follows:

$$\mathbf{a}_{g \to u} = \sum_{g \in \mathcal{G}} \alpha_{gu} \mathbf{e}_g^{(u)}, \tag{5}$$

where $\mathbf{a}_{g \to u}$ denotes the addition information propagated to the target user from all groups. The α_{gu} is the attention weight from group g to user u. And $\mathbf{e}_g^{(u)}$ represents the user partition for group personalized embedding $\mathbf{e}_g$. Analogously, we also have additional user-to-item information as follows:

$$\mathbf{a}_{i \to u} = \sum_{i \in \mathcal{I}} \alpha_{iu} \mathbf{e}_i^{(u)}, \tag{6}$$

where α_{iu} is the attention weight from item i to user u. And $\mathbf{e}_i^{(u)}$ represents the user partition for item personalized embedding $\mathbf{e}_i$.

However, it is problematic if we directly learn these pair-wise attention weights. For example, the attention weights between users and groups α_{gu} increase the parameter complexity by $O(|\mathcal{G}| \times |\mathcal{U}|)$, which is rather large and unable to scale. Inspired by matrix factorization [23], we propose a novel **factorized attention** mechanism to learn those attention weights. Basically, groups and items have additional *contextual embeddings* as $\mathbf{C}_g$ and $\mathbf{C}_i$, respectively, which

are utilized to infer corresponding attention weights. Next, we will introduce the factorized attention mechanism. For simplicity, we only explain how to infer group-user attention weight α_{gu} via context embedding of groups. Item-user attention weight α_{iu} can be derived in analogy.

We calculate the attention weight α_{gu} based on the relatedness between this group g and all the neighboring groups of the target user u. The intuition is that users tend to be more interested in groups similar to users' previously participated groups. We formulate the calculation as follows:

$$\alpha_{gu} = \frac{\exp(\sigma(\sum_{m \in \mathcal{N}_u^{(g)}} \mathbf{R}_{mg}))}{\sum_{k \in \mathcal{G}} \exp(\sigma(\sum_{m \in \mathcal{N}_u^{(g)}} \mathbf{R}_{mk}))}, \tag{7}$$

where $\mathbf{R}_{mg}$ denotes the relatedness between group m and group g and group m is from user participated groups $\mathcal{N}_u^{(g)}$. We use LeakyRelu as the nonlinear function $\sigma(\cdot)$. The group contextual attention matrix $\mathbf{R} \in \mathbb{R}^{|\mathcal{G}| \times |\mathcal{G}|}$ enables us to aggregate non-direct group information thus augmenting the propagation. We calculate elements $\mathbf{R}_{mg} \in \mathbf{R}$ as follows:

$$\mathbf{R}_{mg} = \frac{\exp(\mathbf{c}_m \cdot \mathbf{c}_g)}{\sum_{k \in \mathcal{G}} \exp(\mathbf{c}_k \cdot \mathbf{c}_g)}, \tag{8}$$

where $\mathbf{c}_m \in \mathbb{R}^d$ is the contextual embedding for group m. In fact, we calculate the relatedness matrix directly through the product between the group contextual embeddings and its transpose as follows:

$$\mathbf{R} = \mathrm{softmax}(\mathbf{C}_g^\top \mathbf{C}_g), \tag{9}$$

where $\mathbf{C}_g$ is the contextual embeddings for groups. As observed, we decompose the calculation of pair-wise attention weights, *i.e.* the α_{gu} in Eq. (5), into the product of a d-rank matrix and its transpose. Hence, it is a factorized attention mechanism. Our experiments also demonstrate its better performance compared with existing graph attention layers [27]. We present the factorized attention layer in Figure 2. By stacking PA layer with the tripartite convolution layer, we enhance the group and item partition embedding of user u with this additional information propagation as follows:

$$\hat{\mathbf{e}}_u^{(g)} = \mathbf{e}_u^{(g)} + \beta \mathbf{a}_{g \to u}, \ \hat{\mathbf{e}}_u^{(i)} = \mathbf{e}_u^{(i)} + \beta \mathbf{a}_{i \to u}, \tag{10}$$

where $\beta \in [0, 1]$ is a scalar hyper-parameter to control the intensity of PA. Note that when $\beta = 0$, there is no augmentation for the propagation. Then, we aggregate these enhanced embeddings in tripartite graph convolution and substitute the $\mathbf{e}_u^{(g)}$ and $\mathbf{e}_u^{(i)}$ in Eq. (2) with $\hat{\mathbf{e}}_u^{(g)}$ and $\hat{\mathbf{e}}_u^{(i)}$, respectively. We present this PA layer process in Figure 2.

4.4 Ranking Prediction and Optimization

The final prediction process is demonstrated in the right part of Figure 2. After stacking L layers, we obtain the user embedding $\mathbf{e}_u^{(L)}$ and group embedding $\mathbf{e}_g^{(L)}$. The ranking score of the user-group pair (u, g) is calculated by the inner product:

$$y_{u,g} = \mathbf{e}_u^{(L)} \cdot \mathbf{e}_g^{(L)}. \tag{11}$$

Then we employ the pairwise Bayesian Personalized Ranking (BPR) loss [23] as our loss function:

$$\mathcal{L} = \sum_{(u,g,g') \in \mathcal{D}} -\log \sigma(\hat{y}_{u,g} - \hat{y}_{u,g'}) + \lambda \|\Theta\|_2^2, \tag{12}$$

where $\mathcal{D} = \{(u, g, g') | g \in \mathcal{G}_u^+, g' \in \mathcal{G} \backslash \mathcal{G}_u^+\}$ is the training data with positive interactions and random negative samples. Θ is all trainable parameters in the framework, which is regularized by λ. Adam [20] is chosen as the optimizer.

5 EXPERIMENT

5.1 Experimental Setup

5.1.1 Datasets. We conduct on three real-world datasets: Mafengwo, Weeplaces and Steam. Both Mafengwo and Weeplaces datasets contain the user's travel history with a location-based social network. The history of creating or joining group travel for a user is recorded in Mafengwo [1]. For Weeplaces, we construct group interactions with venues by check-in time and users' social networks in the same way as GroupIM [24]. Both Mafengwo and Weeplaces have limited users as shown in Table 2. Therefore, we release a new dataset Steam which includes 11,099 users and 57,654 group participation records on Steam online game platform. The statistics of three datasets is shown in Table 2. For Mafengwo, we randomly select 70% of all groups joined by each user for training and validation and the remaining 30% for testing. For Weeplaces and Steam, the split ratio is 80% for training and validation and 20% for the testing set. Our implementation is available online[4].

5.1.2 Baselines. To justify the effectiveness of our work, we compare the following baselines:

- **AGREE** [1]. This model is designed to recommend items to groups and users, which integrates the user, item and group information. We adapt it to RGI task by endowing AGREE with user-group pairwise BPR loss instead of the original item prediction loss.
- **MF-BPR** [23]. This is the classical pair-wise matrix factorization based recommendation model optimized by the BPR loss.
- **ENMF** [2]. This model based on a neural matrix factorization architecture leverages mathematical optimization to train the model efficiently without sampling data.
- **NGCF** [31]. This method is a variant of standard GCN [21] leveraging high-order connectivity in a user-item bipartite graph for collaborative filtering.
- **LightGCN** [14]. This is a method based on NGCF with optimization in training efficiency and generation ability by removing feature transformation and nonlinear activation.
- **SGL** [33]. This work performs contrastive learning on LightGCN to augment node representations for user-item recommendation.

Since those user-item recommendation baseline methods are not designed for tripartite graphs, we deploy them with only user-group interactions such that they are adapted to RGI task.

5.1.3 Parameter Settings. We apply a grid search for hyperparameters tuning in our model. We searched embedding size in {128, 256, 512, 1024, 2048}, learning rate in {0.0001, 0.0005, 0.001, 0.005,

---

[4]https://github.com/mdyfrank/CFAG

0.01}, regularization parameter λ_Θ in {0.001, 0.005, 0.01, 0.05, 0.1}, and the hyperparameter λ_α to control the strength of attention in {0.01, 0.05, 0.1, 0.5, 1}. We set both personalized embeddings and contextualized embeddings in the same embedding size and leave the exploration of different embedding sizes in future work. A simple division over the dimension is used as partition layer, and direct concatenation is used as merging layer. We use one convolutional layer with batch size = 2048 for Mafengwo and Weeplaces, and two convolutional layers with batch size = 8196 for the larger Steam dataset. Early stopping is utilized in all experiments to cope with the over-fitting problem.

5.1.4 Evaluation Metrics. We evaluate RGI task by ranking the test groups with all non-interacted groups of users. And we adopt Recall@{10, 20} and NDCG@{10, 20} as evaluation metrics.

5.2 Overall Performance Comparison

We present the overall comparison results in Table 1. The best results among all methods are in boldface, and the second best results are underlined. We summarize the following key observations:

- The proposed CFAG method achieves the best results on all three datasets. Especially, it outperforms all the baseline methods significantly in Weeplaces dataset by more than 60%. We hypothesize these large gains result from the abundant user-group interactions as Table 1 shows. This demonstrates that CFAG is able to well characterize the user-group interactions. The performance gain in the other two datasets is from 8.53% to 25.67%, which demonstrates the superiority of the proposed framework.
- The bipartite graph convolutional networks, such as LightGCN and NGCF, are better than a simple matrix factorization method MF-BPR on all the datasets, which indicates the benefits of using graph propagation to learn embeddings. However, they are still unable to incorporate the tripartite graph information, thus being worse than CFAG. This observation justifies the necessity of tripartite graph convolution for RGI task.
- Although AGREE integrates both the user-item and user-group interaction information, its poor performance compared to other baseline methods indicates that such methods designed for group-item recommendation tasks cannot be directly applied to RGI task. Compared with it, CFAG is specifically designed for RGI task with tripartite graph convolution, and propagation augmentation via factorized attention, which is a better framework.

5.3 Ablation study

In this section, we conduct two types of ablation study to justify the effectiveness of those modules in CFAG, which are partition/merging layers in tripartite graph convolution and PA layers.

5.3.1 Partition and Merging Layer Settings. We demonstrate the performance of CFAG with different partition and merging layer settings on the three datasets in Figure 4. CFAG employs a simple division over the dimension as partition layer and a direct concatenation as merging layer. In addition, we investigate three other settings: (P1) Fully-connected layer as partition layer; (M1) concatenation before a fully-connected layer as merging layer; and (M2) concatenation after a fully-connected layer as merging layer. We only show the result on NDCG since the pattern on Recall is

Table 1: RGI performance comparison on three datasets.

Dataset	Mafengwo				Weeplaces				Steam			
Metric	R@10	R@20	N@10	N@20	R@10	R@20	N@10	N@20	R@10	R@20	N@10	N@20
AGREE	0.0287	0.0495	0.0104	0.0157	0.0174	0.0322	0.0081	0.0119	0.0497	0.0683	0.0239	0.0283
MF-BPR	0.1930	0.2388	0.1185	0.1305	0.2349	0.2826	0.1479	0.1625	0.2034	0.2859	0.1138	0.1356
ENMF	0.3038	0.3568	0.1805	0.1959	0.2283	0.3260	0.1258	0.1577	0.2424	0.3153	0.1415	0.1605
NGCF	0.2012	0.2723	0.1102	0.1288	0.1742	0.2332	0.1068	0.1249	0.2356	0.3448	0.1286	0.1573
LightGCN	0.2531	0.3561	0.1385	0.1839	0.2585	0.3293	0.1479	0.1636	0.2388	0.3509	0.1294	0.1611
SGL	0.2586	0.3075	0.1454	0.1576	0.2650	0.3210	0.1563	0.1704	0.2221	0.3030	0.1230	0.1437
CFAG	**0.3540**	**0.4484**	**0.1959**	**0.2184**	**0.4264**	**0.5323**	**0.2590**	**0.2790**	**0.2851**	**0.3856**	**0.1599**	**0.1824**
Improvement	16.52%	25.67%	8.53%	11.49%	60.87%	61.65%	65.71%	63.73%	17.62%	9.88%	13.00%	13.22%

Table 2: The statistics of datasets.

Dataset	Mafengwo	Weeplaces	Steam
# users	1,269	1,501	11,099
# groups	972	4,651	1,085
# items	999	6,406	2,351
# user-group edges	5,574	12,258	57,654
# user-item edges	8,676	43,942	444,776
# group-item edges	2,540	6,033	22,318
Avg. # groups/user	4.39	8.17	5.19
Avg. # items/user	6.84	29.28	40.07
Avg. # items/group	2.61	1.29	20.57

Table 3: Ablation study on PA layers.

Dataset	Mafengwo		Weeplaces		Steam	
Metric	R@10	N@10	R@10	N@10	R@10	N@10
w/o PA	0.3338	0.1881	0.4065	0.2419	0.2620	0.1389
w/o item	0.3379	0.1913	0.4116	0.2438	0.2838	0.1560
w/o group	0.3381	0.1899	0.4198	0.2504	0.2835	0.1538
GAT att.	0.3190	0.1747	0.4172	0.2484	0.2800	0.1520
CFAG	**0.3540**	**0.1959**	**0.4264**	**0.2590**	**0.2851**	**0.1599**

compared with w/o PA, which indicates the benefits of PA layers. However, w/o item and w/o group PA are both worse than CFAG which uses both PA. This also justifies the necessity of augmenting both user-item interactions and user-group participation. Additionally, we observe that using the GAT layer to learn attention weights is not able to improve the performance. GAT att. even yields the worst performance on Mafengwo dataset. We argue that GAT layer introduces redundant parameters to learn the attention weights. Therefore, we believe that the proposed factorized attention mechanism is a better way to infer the attention weights for PA layers.

5.4 Contextual Embedding Analysis

As aforementioned, contextual embeddings characterize the similarity of groups and items, and thus we can construct augmented propagation paths. To verify this, we conduct analyses of those learned contextual embeddings of CFAG from two perspectives.

Firstly, we investigate the distribution of all the values in the relatedness matrix $\mathbf{R}$. Since values in it are too small, we instead present the pair-wise value before passing to softmax, *i.e.*, the $\mathbf{c}_m \cdot \mathbf{c}_g$. Also, due to the space limitation, we only present the results regarding group contextual embeddings. Item contextual embeddings have similar patterns. The distributions on three datasets are present in Figure 5. Two obvious peaks appear on all datasets. The first peak centers at 0 and is much higher than the second peak, which suggests that most of the groups are not related based on contextual embedding. The second peak centers at a distinct value. The second peak is actually those correlated groups and those similar groups constructs augmented propagation paths from groups to users. Therefore, we conclude that the learned contextual embedding is able to reveal the similarity of groups and benefits the propagation augmentations.

Secondly, we retrieve all pairs of groups. Then, we calculate their relatedness and corresponding common user ratio. The ratio for
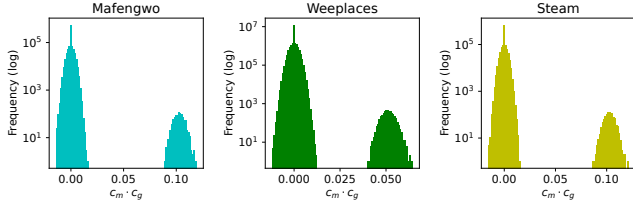
the same. We observe that CFAG consistently performs the best on three datasets. This justifies the superiority of CFAG with a simple division as partition layer and a direct concatenation as merging layer performs best among all the settings. The reason is that those fully connected layers increase the complexity of the model with redundant parameters and also degrade model effectiveness.
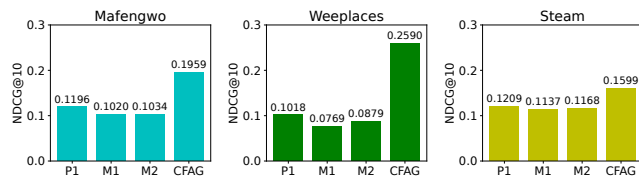
Figure 4: Performance of CFAG w.r.t. partition/merging layer settings.

5.3.2 PA layer variants. Another ablation study is further made to investigate layers. We design four variants of PA layers. 1) *w/o PA* is without using PA layer in CFAG, which only employs tripartite graph convolution layer. 2) *w/o group* is the PA layers without the user-to-group propagation augmentation. 3) *w/o item* is the PA layers without the item-to-user propagation augmentation. And 4) *GAT att.* is changing the factorized attention mechanism to GAT attention layers in [27]. We demonstrate the results in Table 3.

From the results, we can observe that without the PA layer, the performance is the worst on both Weeplaces and Steam datasets. This verifies the effectiveness of PA layers. Moreover, if we use only group-to-user or item-to-user PA, the performance is improved

Figure 5: Distribution of the dot-product values on group contextualized embeddings, *i.e.*, $c_m \cdot c_g$.

Figure 7: Cold-start performance of different methods.

each group pair (g_a, g_b) is computed as follows:

$$r_{ab} = \frac{|\mathcal{N}_{g_a}^{(u)} \cap \mathcal{N}_{g_b}^{(u)}|}{|\mathcal{N}_{g_a}^{(u)} \cup \mathcal{N}_{g_b}^{(u)}|}, \tag{13}$$

where $\mathcal{N}_{g_a}^{(u)}$ and $\mathcal{N}_{g_b}^{(u)}$ is the set of users in group g_a and g_b, respectively. Hence, the nominator in Eq.(13) denotes the common users for group pair (g_a, g_b) while the denominator denotes the total number of users in this pair. For a simple illustration purpose, we sort all group pairs w.r.t. the relatedness scores and split them into 10 equal size subsets, and represent each subset as the average relatedness scores. Then, we calculate the average common user ratio in each subset. The scatter plots between relatedness and common user ratio sharing ratio on three datasets are shown in Figure 6. We also draw a regression line and compute the Pearson correlation coefficient p for each dataset.
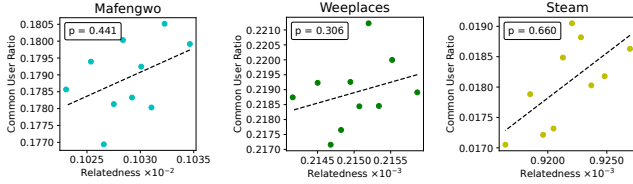

Figure 6: Common user ratio w.r.t. relatedness of group pairs based on contextual embeddings.

We have the following observations: Firstly, an overall tendency is that the common user ratio increases with the growing of relatedness, especially in Mafengwo and Steam datasets. This tendency indicates that two groups sharing more common members can have higher relatedness based on contextual embeddings, which justifies the efficacy of learned contextual embeddings. Secondly, the highest p value in the largest Steam dataset implies that the propagation augmentation ability of contextual embeddings can be more effective on larger datasets.

5.5 Cold-start Group Recommendation

As aforementioned, the cold-start issue in RGI is severe. Hence, we conduct a detailed analysis regarding the ability of CFAG to tackle cold-start group recommendation. We randomly remove some user-group edges for each user in the training set such that the number of neighbor groups of each user is no greater than a threshold k. 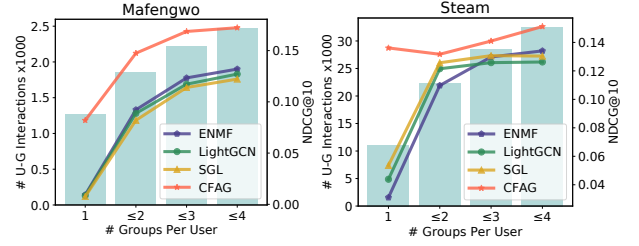For comparison, we choose ENMF, LightGCN, and SGL as the three baseline models, and perform the experiments with threshold $k \in \{1, 2, 3, 4\}$. The threshold indicates the maximum number of groups per user. The results are in Figure 7. We report the NDCG performance with respect to different thresholds as the solid line and the number of user-group interactions in the background histograms. We observe that CFAG significantly outperforms other baselines. The reasons are twofold. First, CFAG can leverage both the item and group interactions to learn node embeddings. Though few group participation of users, item interactions complement the cold-start issue. The other reason is that the PA layers in CFAG resolve the cold-start issue by discovering potential unconnected neighbors, thus being able to improve the cold-start performance.

Additionally, we observe that on the large-scale Steam datasets, CFAG even performs better when $k = 1$ than on $k = 2$. We hypothesize that when the number of groups per user is few, CFAG can well characterize the group interests of users from their item interactions. Therefore, it verifies the CFAG is a better framework to comprehensively integrate item and group information for users and can successfully complete the RGI task.

6 CONCLUSION

In this paper, we formulate the Ranking-based Group Identification (RGI) problem, with the goal of recommending new groups to a target user based on the user's group and item interactions. In RGI problem, it is challenging to effectively harness both item interactions and group participation on a social tripartite graph. To address these challenges, we propose a novel framework CFAG, which is able to (i) effectively aggregate information from different types of neighborhoods via the tripartite graph convolution, and (ii) augment the propagation paths to resolve the data sparsity issue via PA layers. We conduct extensive experiments and detailed analyses on three datasets to verify the effectiveness of CFAG. In the future, we may explore how to select the optimal neighbors for aggregation and different aggregator layers. As such, we can improve the functionality of tripartite graph convolution layers.

7 ACKNOWLEDGEMENTS

REFERENCES

[1] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang Hong. 2018. Attentive Group Recommendation. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 645–654.

[2] Chong Chen, Min Zhang, Yongfeng Zhang, Yiqun Liu, and Shaoping Ma. 2020. Efficient Neural Matrix Factorization without Sampling for Recommendation. *ACM Trans. Inf. Syst.* 38, 2, Article 14 (Jan 2020), 28 pages.

[3] Wen-Yen Chen, Dong Zhang, and Edward Y. Chang. 2008. Combinational Collaborative Filtering for Personalized Community Recommendation. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada, USA) *(KDD '08)*. Association for Computing Machinery, New York, NY, USA, 115–123. https://doi.org/10.1145/1401890.1401909

[4] Zeyu Cui, Zekun Li, Shu Wu, Xiao-Yu Zhang, and Liang Wang. 2019. Dressing as a Whole: Outfit Compatibility Learning Based on Node-Wise Graph Neural Networks. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 307–317.

[5] Daizong Ding, Mi Zhang, Shao-Yuan Li, Jie Tang, Xiaotie Chen, and Zhi-Hua Zhou. 2017. BayDNN: Friend Recommendation with Bayesian Personalized Ranking Deep Neural Network. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management* (Singapore, Singapore) *(CIKM '17)*. Association for Computing Machinery, New York, NY, USA, 1479–1488.

[6] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 417–426.

[7] Wenqi Fan, Yao Ma, Dawei Yin, Jianping Wang, Jiliang Tang, and Qing Li. 2019. Deep Social Collaborative Filtering. In *Proceedings of the 13th ACM Conference on Recommender Systems* (Copenhagen, Denmark) *(RecSys '19)*. Association for Computing Machinery, New York, NY, USA, 305–313.

[8] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3-5 (2010), 75–174.

[9] Linke Guo, Chi Zhang, and Yuguang Fang. 2015. A Trust-Based Privacy-Preserving Friend Recommendation Scheme for Online Social Networks. *IEEE Transactions on Dependable and Secure Computing* 12, 4 (2015), 413–427.

[10] Zhiwei Guo and Heng Wang. 2021. A Deep Graph Neural Network-Based Mechanism for Social Recommendations. *IEEE Transactions on Industrial Informatics* 17, 4 (2021), 2776–2783.

[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[12] Xiaotian Han, Chuan Shi, Senzhang Wang, Philip S. Yu, and Li Song. 2018. Aspect-Level Deep Collaborative Filtering via Heterogeneous Information Networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 3393–3399.

[13] Junheng Hao, Tong Zhao, Jin Li, Xin Luna Dong, Christos Faloutsos, Yizhou Sun, and Wei Wang. 2020. P-Companion: A Principled Framework for Diversified Complementary Product Recommendation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. ACM, 2517–2524.

[14] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. *LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation*. Association for Computing Machinery, New York, NY, USA, 639–648.

[15] Liang Hu, Songlei Jian, Longbing Cao, Zhiping Gu, Qingkui Chen, and Artak Amirbekyan. 2019. HERS: Modeling Influential Contexts with Heterogeneous Relations for Sparse and Cold-Start Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (Jul. 2019), 3830–3837.

[16] F.O. Isinkaye, Y.O. Folajimi, and B.A. Ojokoh. 2015. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal* 16, 3 (2015), 261–273.

[17] Di Jin, Ziyang Liu, Weihao Li, Dongxiao He, and Weixiong Zhang. 2019. Graph Convolutional Networks Meet Markov Random Fields: Semi-Supervised Community Detection in Attribute Networks. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (Jul. 2019), 152–159.

[18] Davod Karimpour, Mohammad Ali Zare Chahooki, and Ali Hashemi. 2021. GroupRec: Group Recommendation by Numerical Characteristics of Groups in Telegram. In *2021 11th International Conference on Computer Engineering and Knowledge (ICCKE)*. 115–120. https://doi.org/10.1109/ICCKE54056.2021.9721494

[19] Davod Karimpour, Mohammad Ali Zare Chahooki, and Ali Hashemi. 2021. Telegram group recommendation based on users' migration. In *2021 26th International Computer Conference, Computer Society of Iran (CSICC)*. 1–6. https://doi.org/10.1109/CSICC52343.2021.9420581

[20] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[21] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[22] Keunchan Park, Jisoo Lee, and Jaeho Choi. 2017. Deep Neural Networks for News Recommendations. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*. ACM, 2255–2258.

[23] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (Montreal, Quebec, Canada) *(UAI '09)*. AUAI Press, Arlington, Virginia, USA, 452–461.

[24] Aravind Sankar, Yanhong Wu, Yuhang Wu, Wei Zhang, Hao Yang, and Hari Sundaram. 2020. *GroupIM: A Mutual Information Maximization Framework for Neural Group Recommendation*. Association for Computing Machinery, New York, NY, USA, 1279–1288.

[25] Terence A Shimp and J Craig Andrews. 2012. *Advertising promotion and other aspects of integrated marketing communications*. Cengage Learning.

[26] Zhulin Tao, Yinwei Wei, Xiang Wang, Xiangnan He, Xianglin Huang, and Tat-Seng Chua. 2020. MGAT: Multimodal Graph Attention Network for Recommendation. *Information Processing and Management* 57, 5 (2020), 102277.

[27] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

[28] Jingdong Wang, Zhe Zhao, Jiazhen Zhou, Hao Wang, Bin Cui, and Guojun Qi. 2012. Recommending Groups with Social Topic Model. *Inf. Retr.* 15, 3–4 (jun 2012), 278–295. https://doi.org/10.1007/s10791-012-9193-0

[29] Shoujin Wang, Liang Hu, Yan Wang, Xiangnan He, Quan Z. Sheng, Mehmet A. Orgun, Longbing Cao, Francesco Ricci, and Philip S. Yu. 2021. Graph learning based recommender systems: a review. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence, 4644–4652. 30th International Joint Conference on Artificial Intelligence, IJCAI 2021 ; Conference date: 19-08-2021 Through 27-08-2021.

[30] Xin Wang, Roger Donaldson, Christopher Nell, Peter Gorniak, Martin Ester, and Jiajun Bu. 2016. Recommending Groups to Users Using User-Group Engagement and Time-Dependent Matrix Factorization. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence* (Phoenix, Arizona) *(AAAI'16)*. AAAI Press, 1331–1337.

[31] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 165–174.

[32] Yufei Wen, Lei Guo, Zhumin Chen, and Jun Ma. 2018. Network Embedding Based Recommendation Method in Social Networks. In *Companion Proceedings of the The Web Conference 2018* (Lyon, France) *(WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 11–12.

[33] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised Graph Learning for Recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 726–735.

[34] Lu Yang, Yezheng Liu, Yuanchun Jiang, Le Wu, and Jianshan Sun. 2021. Predicting personalized grouping and consumption: A collaborative evolution model. *Knowledge-Based Systems* 228 (2021), 107248. https://doi.org/10.1016/j.knosys.2021.107248

[35] Liangwei Yang, Zhiwei Liu, Yingtong Dou, Jing Ma, and Philip S. Yu. 2021. ConsisRec: Enhancing GNN for Social Recommendation via Consistent Neighbor Aggregation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 2141–2145.

[36] Liangwei Yang, Zhiwei Liu, Yu Wang, Chen Wang, Ziwei Fan, and Philip S. Yu. 2022. Large-scale Personalized Video Game Recommendation via Social-aware Contextualized Graph Neural Network. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. ACM, 3376–3386.

[37] Quan Yuan, Gao Cong, and Chin-Yew Lin. 2014. COM: A Generative Model for Group Recommendation. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, New York, USA) *(KDD '14)*. Association for Computing Machinery, New York, NY, USA, 163–172.

[38] Mingli Zhang, Lingyun Guo, Mu Hu, and Wenhua Liu. 2017. Influence of customer engagement with company social networks on stickiness: Mediating effect of customer value creation. *International Journal of Information Management* 37, 3 (2017), 229–240.