

Self-supervised Graph Learning for Occasional Group Recommendation

Bowen Hao
Renmin University of China
China
jeremyhao@ruc.edu.cn

Hongzhi Yin
The University of Queensland
Australia
h.yin1@uq.edu.au

Jing Zhang
Renmin University of China
China
zhang-jing@ruc.edu.cn

Cuiping Li
Renmin University of China
China
licuiping@ruc.edu.cn

Hong Chen
Renmin University of China
China
chong@ruc.edu.cn

ABSTRACT

We study the problem of recommending items to occasional groups (a.k.a. cold-start groups), where the occasional groups are formed ad-hoc and have few or no historical interacted items. Due to the extreme sparsity issue of the occasional groups' interactions with items, it is difficult to learn high-quality embeddings for these occasional groups. Despite the recent advances on Graph Neural Networks (GNNs) incorporate high-order collaborative signals to alleviate the problem, the high-order cold-start neighbors are not explicitly considered during the graph convolution in GNNs. This paper proposes a self-supervised graph learning paradigm, which jointly trains the backbone GNN model to reconstruct the group/user/item embeddings under the meta-learning setting, such that it can directly improve the embedding quality and can be easily adapted to the new occasional groups. To further reduce the impact from the cold-start neighbors, we incorporate a self-attention-based meta aggregator to enhance the aggregation ability of each graph convolution step. Besides, we add a contrastive learning (CL) adapter to explicitly consider the correlations between the group and non-group members. Experimental results on three public recommendation datasets show the superiority of our proposed model against the state-of-the-art group recommendation methods.

CCS CONCEPTS

• Information systems → Recommendation.

KEYWORDS

Occasional group recommendation, self-supervised learning, graph neural network

ACM Reference Format:

Bowen Hao, Hongzhi Yin, Jing Zhang, Cuiping Li, and Hong Chen. 2018. Self-supervised Graph Learning for Occasional Group Recommendation. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Woodstock '18, June 03–05, 2018, Woodstock, NY

© 2018 Association for Computing Machinery.

ACM ISBN 978-1-4503-XXXX-X/18/06...\$15.00

<https://doi.org/10.1145/1122445.1122456>

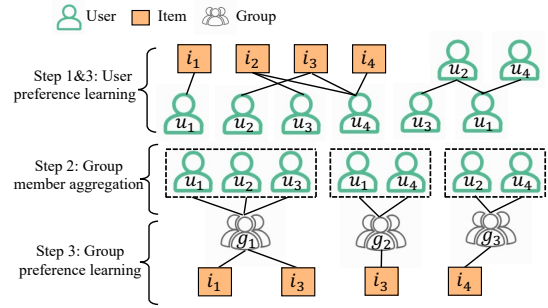


Figure 1: A toy example for group recommendation.

2018, Woodstock, NY. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/1122445.1122456>

1 INTRODUCTION

Recommender systems have played a crucial role in social media platforms, due to their promising ability in mitigating the information overload issue. With the recent advances in social platform services like Meetup and Facebook Event [23, 30], it is increasingly convenient for people with similar backgrounds (e.g., hobbies, locations) to form social groups to participate in activities such as group tours, class reunion, family dinners [9, 40]. The proliferation of groups in the social media platforms demands an effective way to perform group recommendation. This paper addresses the problem of recommending items to occasional groups (a.k.a. cold-start groups), where the occasional groups are formed ad-hoc and have few or no historical interacted items. Due to the extreme sparsity issue of the occasional groups' interactions with items, it is difficult to learn high-quality embeddings for these occasional groups.

To solve this problem, some early studies adopt heuristic predefined aggregation strategy such as average [2], least misery [1] and maximum satisfaction [3] to aggregate the user preference to obtain the group preference. However, due to the fixed aggregation strategies, these methods are insufficient to capture the complicated and dynamic process of group decision making, which results in the unstable recommendation performance [25]. Further, Cao et al. [4] propose to assign each user an attention weight, which denotes the influence of group member in deciding the group's choice on the target item. However, when some users in the occasional group

only interact with few items (a.k.a. cold-start users), the attention weight assigned for each user is diluted by these cold-start users, and thus results in biased group profile.

Recently, inspired by the development of graph neural networks (GNNs), a few GNN-based recommender models are proposed [12, 13, 29, 34]. The basic idea is to incorporate high-order neighbors to enhance the representations of the cold-start users, and then obtain refined group representation. As shown in Figure 1, the GNN model first conducts graph convolution multiple steps on the user-user and user-item interaction graphs to learn the preference of group members, and then performs average [12], summation and pooling [43] or attention mechanism [12] to aggregate the preferences of group members to obtain the group representation. Finally, based on the aggregated embeddings of groups and users, the likelihood of a group/user adopt an item is estimated, and the BPR loss [28] or cross entropy loss [15] is usually adopted to compare the likelihood and the true observations. Moreover, Zhang et al. [44] propose hypergraph convolution network (HHGR) with self-supervised node dropout strategy, which can model complex high-order interactions between groups and users. Through incorporating self-supervised signals, HHGR can alleviate the data sparsity issue in some extent.

However, the above methods still suffer from the following challenges. First, the group representation not only depends on the group members' preferences, but also relies on the group-level preferences towards items and collaborative group signals (the groups that share common users/items). Although some GNNs consider either group-level preferences [19] or collaborative group signals [12, 44] to form the group representation, they do not consider all these signals together. Second, the GNNs can not explicitly deal with the high-order cold-start neighbors when performing graph convolution. For example in Figure 1, for the target group g_2 , its group member u_1 and high-order neighbor i_1 only have few interactions. The embeddings of u_1 and i_1 are inaccurate, which will affect the embedding of g_2 when performing graph convolution. Third, existing GNNs only consider the correlations between the group and its members, but ignore the correlations between the group and non-group members. Thus, this inspires the following research problem: *how can we learn more accurate embeddings by GNNs for occasional group recommendation?*

To this end, motivated by the self-supervised learning (SSL) technique [16, 21, 26], which aims to spontaneously find the supervised signals from the input data itself and can further benefit the downstream tasks, we propose a new Self-supervised Graph learning paradigm for Group recommendation (SGG), which trains the backbone GNN model to jointly reconstruct the group/user/item embeddings from multiple interaction graphs under the meta-learning setting [33], such that it can directly improve the embedding quality. Specifically, we first pick groups/users/items with sufficient interactions as the target groups/users/items and then learn their ground truth embeddings from the observed abundant interactions. To simulate the cold-start scenarios, in each training episode, we randomly sample K neighbors for each target group/user/item in their counterpart interaction graphs¹, based on which we perform graph convolution multiple steps in each interaction graph, and

¹For each group, its counterpart graphs are group-group, group-item and group-user graphs; for each user, his counterpart graphs are user-user and user-item graphs; for each item, its counterpart graph is user-item graph.

fuse the corresponding refined embeddings to predict the target embedding. Finally, we jointly optimize the reconstruction losses between the predicted embeddings and the ground truth embeddings of groups/users/items, making the GNN model easily and rapidly being adapted to new cold-start groups/users/items. Through the above process, the reconstructed group representation can contain all the signals as presented in the first challenge.

Nevertheless, the above proposed pretext task still can not explicitly deal with the high-order cold-start neighbors. Besides, the correlations between the groups and non-group members are still not explicitly considered. To further deal with the high-order cold-start neighbors, we incorporate a meta aggregator to enhance the aggregation ability of each graph convolution step. Specifically, the meta aggregator learns cold-start node's embedding on its first-order neighbors in the counterpart interaction graphs by self-attention mechanism under the same meta-learning setting, which is then incorporated into each graph convolution step to enhance the aggregation ability. To explicitly consider the correlations between groups and non-group members, motivated by Contrastive Learning (CL) [7], which pulls the similar instances together while pulling away dissimilar instances, we propose a CL adapter, which contrasts the group embedding with its members' and non-members' embeddings to regularize the group and user embeddings under the same meta-learning setting. The contributions are as follows:

- We design a new self-supervised graph learning paradigm for group recommendation, which jointly trains the backbone GNN model to reconstruct the group/user/item embedding under the meta-learning setting. To deal with the cold-start neighbors, we further introduce a meta aggregator to enhance the aggregation ability of each graph convolution step.
- To explicitly consider the correlations between the group and non-group members, we further propose a CL adapter to regularize the group and user embeddings.
- Experimental results on the group recommendation task demonstrate the superiority of our proposed model against the state-of-the-art group recommendation models.

2 PRELIMINARIES

In this section, we first define the problem and then present a base GNN framework that can be used to solve the problem.

2.1 Problem Definition

There are three sets of entities in the group recommendation scenario: a user set $U=\{u_1, \dots, u_{|U|}\}$, an item set $I=\{i_1, \dots, i_{|I|}\}$ and a group set $G=\{g_1, \dots, g_{|G|}\}$. There are three kinds of observed interaction graphs among U , I and G : group-item subgraph \mathcal{G}_{GI} , user-item subgraph \mathcal{G}_{UI} and group-user subgraph \mathcal{G}_{GU} . Since the social connections of user-user and group-group are also important to depict the user and group profiles, we build two kinds of implicit interaction graphs based on \mathcal{G}_{GI} and \mathcal{G}_{UI} , namely user-user subgraph \mathcal{G}_{UU} and group-group subgraph \mathcal{G}_{GG} . In \mathcal{G}_{UU} , the two users are connected if they share with more than c_u items. Similarly, in \mathcal{G}_{GG} , the two groups are connected if they share with more than c_g items. Formally, we use notation $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ to denote the set of observed and implicit interaction graphs, i.e., $\mathcal{G} =$

$\mathcal{G}_{GI} \cup \mathcal{G}_{UI} \cup \mathcal{G}_{GU} \cup \mathcal{G}_{GG} \cup \mathcal{G}_{UU}$, where \mathcal{V} is the set of nodes $\{U, I, G\}$, and \mathcal{E} is the set of edges.

Definition 1. GNN for Group Recommendation. Given the interaction graph \mathcal{G} , we aim to train a GNN-based encoder f that can recommend top- k items for the target group g .

2.2 Base GNN for Group Recommendation

Although existing GNN-based group recommendation methods show their diversity in modelling group interactions with users and items [12, 13, 19, 34], we notice that they essentially share a general model structure. Based on this finding, we present a base GNN model, which consists of a representation learning module and a jointly training module. The representation learning module learns the representations of groups and users upon their counterpart interaction graphs, while the jointly training module optimizes the user/group preferences over items to compare the likelihood and the true user-item/group-item observations.

2.2.1 Representation Learning Module. This module first learns the user representation upon the user-item and user-user subgraphs, and then learns the group representation upon the group-group, group-item and group-user subgraphs. Specifically, for each user u , we first sample his first-order neighbors on \mathcal{G}_{UI} and \mathcal{G}_{UU} , and then perform graph convolution: $\mathbf{h}_{uUI}^l = \text{CONV}(\mathbf{h}_{uUI}^{l-1}, \mathbf{h}_{N(uUI)}^l)$, $\mathbf{h}_{uUU}^l = \text{CONV}(\mathbf{h}_{uUU}^{l-1}, \mathbf{h}_{N(uUU)}^l)$, where CONV can be instantiated into any GNN models, such as LightGCN [17] or GCN [22]. \mathbf{h}_{uUI}^l and \mathbf{h}_{uUU}^l denote the user embeddings calculated from \mathcal{G}_{UI} and \mathcal{G}_{UU} at the l -th graph convolution step, \mathbf{h}_{uUI}^0 and \mathbf{h}_{uUU}^0 are randomly initialized embeddings. $\mathbf{h}_{N(uUI)}^l$ and $\mathbf{h}_{N(uUU)}^l$ mean the averaged neighbor embeddings, where the neighbors are sampled from \mathcal{G}_{UI} and \mathcal{G}_{UU} , respectively. After performing L -th convolution steps, we can obtain the refined user embeddings \mathbf{h}_{uUI}^L and \mathbf{h}_{uUU}^L from the counterpart subgraphs. Finally we use attention mechanism [18] to aggregate these embeddings to form the final user embedding \mathbf{h}_u^L : $\mathbf{h}_u^L = \sum_{c \in \{UI, UU\}} a_c \mathbf{h}_{uc}^L$, $a_c = \frac{\exp(\mathbf{W}_c \mathbf{h}_{uc}^L)}{\sum_{c' \in \{UI, UU\}} \exp(\mathbf{W}_{c'} \mathbf{h}_{uc'}^L)}$, where $\{\mathbf{W}_c | c \in \{UI, UU\}\}$ are trainable parameters, $\{a_c | c \in \{UI, UU\}\}$ are the learned attention weight for each subgraph.

Then for each group g , we first sample its first-order neighbors from the \mathcal{G}_{GI} , \mathcal{G}_{GU} and \mathcal{G}_{GG} , and perform graph convolution:

$$\begin{aligned} \mathbf{h}_{gGI}^l &= \text{CONV}(\mathbf{h}_{gGI}^{l-1}, \mathbf{h}_{N(gGI)}^l), \\ \mathbf{h}_{gGU}^l &= \text{CONV}(\mathbf{h}_{gGU}^{l-1}, \mathbf{h}_{N(gGU)}^l), \\ \mathbf{h}_{gGG}^l &= \text{CONV}(\mathbf{h}_{gGG}^{l-1}, \mathbf{h}_{N(gGG)}^l), \end{aligned} \quad (1)$$

where \mathbf{h}_{gGI}^l , \mathbf{h}_{gGU}^l and \mathbf{h}_{gGG}^l denote the group embeddings calculated from \mathcal{G}_{GI} , \mathcal{G}_{GU} and \mathcal{G}_{GG} at the l -th graph convolution step, \mathbf{h}_{gGI}^0 , \mathbf{h}_{gGU}^0 and \mathbf{h}_{gGG}^0 are randomly initialized embeddings. $\mathbf{h}_{N(gGI)}^l$, $\mathbf{h}_{N(gGU)}^l$ and $\mathbf{h}_{N(gGG)}^l$ mean the averaged neighbor embeddings, where the neighbors are sampled from \mathcal{G}_{GI} , \mathcal{G}_{GU} and \mathcal{G}_{GG} , respectively. After performing L -th convolution steps, we can obtain the refined group embeddings \mathbf{h}_{gGI}^L , \mathbf{h}_{gGU}^L and \mathbf{h}_{gGG}^L

from these three subgraphs. Same as existing works [12, 19, 19, 44], we further average the first-order neighbors in \mathcal{G}_{GU} to obtain the aggregated group embedding $\mathbf{h}_{gGU'}^L$,

$$\mathbf{h}_{gGU'}^L = f_{agg}(\{\mathbf{h}_{uGU}^L | u \in \mathcal{N}(g_{GU})\}), \quad (2)$$

where \mathbf{h}_{uGU}^L is obtained by performing graph convolution L steps in \mathcal{G}_{GU} , $\mathcal{N}(g_{GU})$ denotes the first-order user set sampled from \mathcal{G}_{GU} , f_{agg} is the aggregate function such as average [12], summation and pooling [43], or attention mechanism [32]. In our experiments, we find attention mechanism leads to the best performance. Finally, we use attention mechanism to aggregate the above embeddings to form the final group embedding \mathbf{h}_g^L :

$$\begin{aligned} \mathbf{h}_g^L &= \sum_{c \in \{GI, GU, GU', GG\}} a_c \mathbf{h}_{gc}^L, \\ a_c &= \frac{\exp(\mathbf{W}_c \mathbf{h}_{gc}^L)}{\sum_{c' \in \{GI, GU, GU', GG\}} \exp(\mathbf{W}_{c'} \mathbf{h}_{gc'}^L)}, \end{aligned} \quad (3)$$

where $\{\mathbf{W}_c | c \in \{GI, GU, GU', GG\}\}$ are trainable parameters, $\{a_c | c \in \{GI, GU, GU', GG\}\}$ are the learned attention weights.

2.2.2 Jointly Training Module. This module jointly optimizes the user preferences over items with the user-item loss \mathcal{L}_u and the group preferences over items with the group-item loss \mathcal{L}_g , i.e., $\mathcal{L}_{main} = \mathcal{L}_g + \lambda \mathcal{L}_u$, where \mathcal{L}_{main} is the final recommendation loss with a balancing hyper-parameter λ . Here, we use BPR loss [28] to calculate \mathcal{L}_u and \mathcal{L}_g :

$$\begin{aligned} \mathcal{L}_u &= \sum_{(u,i) \in \mathcal{E}_{UI}, (u,j) \notin \mathcal{E}_{UI}} -\ln \sigma(y(u,i) - y(u,j)), \\ \mathcal{L}_g &= \sum_{(g,i) \in \mathcal{E}_{GI}, (g,j) \notin \mathcal{E}_{GI}} -\ln \sigma(y(g,i) - y(g,j)), \end{aligned} \quad (4)$$

where $y(u,i) = \mathbf{h}_u^L \mathbf{h}_i^L$, $y(g,i) = \mathbf{h}_g^L \mathbf{h}_i^L$, σ is an activation function, \mathcal{E}_{UI} and \mathcal{E}_{GI} represent the edges in \mathcal{G}_{UI} and \mathcal{G}_{GI} .

Although the above presented GNNs can address the occasional groups through incorporating high-order collaborative signals, they still can not deal with the groups/users/items with few interactions, and thus can not learn high-quality embeddings for them.

3 THE PROPOSED MODEL

We present the proposed self-supervised graph learning paradigm for group recommendation (SGG). We first describe the process of embedding reconstruction with GNN, and then explain a meta aggregator and a CL adapter that are incorporated in the GNN model to further improve the embedding quality. Finally, we explain how SGG is trained and analyze its time complexity. The overall framework of SGG is shown in Figure 2.

3.1 Embedding Reconstruction with GNN

We propose embedding reconstruction with GNN, which jointly reconstructs the groups/users/items embeddings from multiple subgraphs under the meta-learning setting. Here we take the group embedding reconstruction as an example. User and item embedding reconstruction can be explained in the same way. To achieve this

$$\begin{aligned}
\mathbf{h}_{gGI}^L &= \text{CONV}(\hat{\mathbf{h}}_{gGI}, \mathbf{h}_{gGI}^{L-1}, \mathbf{h}_{N(gGI)}^L), \\
\mathbf{h}_{gGU}^L &= \text{CONV}(\hat{\mathbf{h}}_{gGU}, \mathbf{h}_{gGU}^{L-1}, \mathbf{h}_{N(gGU)}^L), \\
\mathbf{h}_{gGG}^L &= \text{CONV}(\hat{\mathbf{h}}_{gGG}, \mathbf{h}_{gGG}^{L-1}, \mathbf{h}_{N(gGG)}^L).
\end{aligned} \tag{8}$$

For a target group g , Eq. (8) is repeated L steps to obtain the embeddings $\mathbf{h}_{gGI}^L, \mathbf{h}_{gGU}^L, \mathbf{h}_{gGG}^L$, Eq. (2) is used to obtain the aggregated group embedding $\mathbf{h}_{gGU'}^L$, and Eq. (3) is applied to get the final embedding \mathbf{h}_g^L . Finally, the same cosine similarity in Eq. (5) is used to optimize the model parameters, which include the parameters Θ_f of the GNN model and $\Theta_{f_{meta}}$ of the meta aggregator. Similarly, f_{meta} can obtain the meta user embedding on \mathcal{G}_{UI} and \mathcal{G}_{UU} , and the meta item embedding on \mathcal{G}_{UI} .

3.3 CL Adapter

To further consider the correlations between the group and non-group members, motivated by the contrastive learning technique [7], which pulls the similar instances together while pulling away dissimilar instances, we propose a CL adapter, which contrasts the group embedding with its members' / non-members' embeddings to regularize the group and user embeddings under the same meta-learning setting. The group and its members form the positive pair, while the group and its non-group members form the negative pair. The CL loss is defined as:

$$\mathcal{L}_M = \sum_g \sum_{u \in N(g_{GU})} -\log \frac{\exp(\cos(\mathbf{h}_g^L, \mathbf{h}_u^L) / \tau)}{\sum_{u' \sim U \setminus N(g_{GU})} \exp(\cos(\mathbf{h}_g^L, \mathbf{h}_{u'}^L) / \tau)}, \tag{9}$$

where $N(g_{GU})$ denotes the group member set of group g , u denotes the group member, u' denotes non-group member, where for each u , we sample at most K instances for u' . \setminus defines set subtraction operation, $\mathbf{h}_g^L, \mathbf{h}_u^L, \mathbf{h}_{u'}^L$ are the reconstructed embeddings, and τ is a temperature parameter.

3.4 Multi-task Training

To improve recommendation with the SSL task, we leverage a multi-task training strategy [36] to jointly optimize the classic recommendation task (cf. Eq. (4)) and the self-supervised learning task (cf. Eq. (6) and Eq. (9)):

$$\mathcal{L} = \mathcal{L}_{main} + \lambda_1(\mathcal{L}_R + \mathcal{L}_M) + \lambda_2 \|\Theta\|_2^2, \tag{10}$$

where $\Theta = \{\Theta_f, \Theta_{f_{meta}}\}$ is the model parameters, λ_1 and λ_2 are hyperparameters to control the strengths of SGG and L2 regularization, respectively. We also consider the alternative optimization — pre-training on $\mathcal{L}_R + \mathcal{L}_M$ and fine-tuning on \mathcal{L}_{main} . We detail its recommendation performance in Section 4.4.3.

3.5 Complexity Analysis

Here we present the time and space complexity of SGG. Same as LightGCN [17], we implement SGG as the matrix form. Suppose the number of edges in the interaction graph \mathcal{G} is $|\mathcal{E}|$. The number of edges in the masked interaction graph $\hat{\mathcal{G}}$ is $|\hat{\mathcal{E}}|$. Since we mask a large proportion neighbors for each node in \mathcal{G} to simulate the cold-start scenario, the size of masked edge set is far less than the original edge set, i.e., $|\hat{\mathcal{E}}| \ll |\mathcal{E}|$. Let s denote the number of epochs, d denote the embedding size and L denote the number of

Table 1: The comparison of analytical time complexity between GNN and SGG.

Component	GNN	SGG
Adjacency Matrix	$O(2 \mathcal{E})$	$O(10 \mathcal{E} s + 2 \mathcal{E})$
Graph Convolution	$O(2 \mathcal{E} Lds \frac{ \mathcal{E} }{B})$	$O(2(\mathcal{E} + 5 \hat{\mathcal{E}})Lds \frac{ \mathcal{E} }{B})$
BPR Loss	$O(2 \mathcal{E} ds)$	$O(2 \mathcal{E} ds)$
Self-supervised Loss	-	$O(24 \hat{\mathcal{E}} Lds)$

GCN layers. Since SGG introduces meta embedding to enhance the aggregation ability, the space complexity of SGG is twice than that of the vanilla GNN model. The time complexity comes from four parts, namely normalization of adjacency matrix, graph convolution, recommendation loss and self-supervised loss. As there is no change on the model structure and inference process, the time complexity of SGG in the graph convolution and recommendation loss is the same as the vanilla GNN model. We present the main differences between the vanilla GNN and SGG models as follows:

- **Normalization of adjacency matrix.** Since we generate five independent subgraphs per epoch, given the fact that the number of non-zero elements in the adjacency matrices of full training graph and the five subgraphs are $2|\mathcal{E}|, 2|\hat{\mathcal{E}}_{UU}|, 2|\hat{\mathcal{E}}_{UI}|, 2|\hat{\mathcal{E}}_{GG}|, 2|\hat{\mathcal{E}}_{GU}|$ and $2|\hat{\mathcal{E}}_{GI}|$, respectively, its total complexity is $O((2|\hat{\mathcal{E}}_{UU}| + 2|\hat{\mathcal{E}}_{UI}| + 2|\hat{\mathcal{E}}_{GG}| + 2|\hat{\mathcal{E}}_{GU}| + 2|\hat{\mathcal{E}}_{GI}|)s + 2|\mathcal{E}|) \approx 10|\hat{\mathcal{E}}|s + 2|\mathcal{E}|$.³
- **Self-supervised loss.** We evaluate the self-supervised tasks upon the masked subgraphs. For the user or item embedding reconstruction task, the time complexity is $O(2d * (2|\hat{\mathcal{E}}_{UU}| + 2|\hat{\mathcal{E}}_{UI}|) * s * L) \approx 8|\hat{\mathcal{E}}|Lds$. For the group embedding reconstruction task, the time complexity is $O(2d * (2|\hat{\mathcal{E}}_{GG}| + 2|\hat{\mathcal{E}}_{GU}| + 2|\hat{\mathcal{E}}_{GI}|) * s * L) \approx 12|\hat{\mathcal{E}}|Lds$, where $2d$ represents the concatenated embedding size, as we incorporate the meta embedding to the graph convolution process. For the mutual information maximization task, the time complexity is $O(2d * 2|\hat{\mathcal{E}}_{GU}| * s * L) \approx 4|\hat{\mathcal{E}}|Lds$. Thus the total time complexity of self-supervised loss is $24|\hat{\mathcal{E}}|Lds$.

We summarize the time complexity between the vanilla GNNs and SGG in Table 1, from which we observe that the time complexity of SGG is in the same magnitude with the vanilla GNNs, which is totally acceptable, since the increased time complexity of SGG is only from the self-supervised loss. The details are shown in Section 4.4.1.

4 EXPERIMENTS

To verify the superiority and effectiveness of SGG, we conduct extensive experiments and answer the following research questions:

- **RQ1:** How does SGG perform occasional group recommendation compared with the state-of-the-art GNN models?
- **RQ2:** What are the benefits of performing pretext tasks in occasional group recommendation?
- **RQ3:** How do different settings influence the effectiveness of the proposed SGG model?

³We use notation $|\hat{\mathcal{E}}|$ to denote the masked edge length $|\hat{\mathcal{E}}_{UU}|, |\hat{\mathcal{E}}_{UI}|, |\hat{\mathcal{E}}_{GG}|, |\hat{\mathcal{E}}_{GU}|$ and $|\hat{\mathcal{E}}_{GI}|$ for simplicity.

Table 2: Statistics of the Datasets.

	Weeplaces	CAMRa2011	Douban
Users	8,643	602	70,743
Items	25,081	7,710	60,028
Groups	22,733	290	109,538
U-I Interactions	1,358,458	116,314	3,422,266
G-I Interactions	180,229	145,068	164,153
U-I Sparsity	6.27%	2.51%	0.081%
G-I Sparsity	0.03%	6.49%	0.002%

4.1 Experimental Settings

4.1.1 Datasets. We evaluate on three public datasets including Weeplaces [29], CAMRa2011 [4] and Douban [39]. Table 1 illustrates the statistics of these datasets.

4.1.2 Baselines. We select three types of baselines including the state-of-the-art attention based model, the general GNN models and the hypergraph GNN models for group recommendation:

- **MoSAN [31]:** adopts sub-attention mechanism to model the group-item interactions.
- **AGREE [4]:** adopts attention mechanism for jointly modelling user-item and group-item interactions.
- **SIGR [39]:** further incorporates social relationships of groups and users to model the attentive group and user representations.
- **GroupIM [29]:** further regularizes group and user representations by maximizing the mutual information between the group and its members.
- **GAME [19]:** performs graph convolution only based on the first-order neighbors from the group-group, group-user and group-item graphs for group recommendation.
- **GCMC [37]** employs the standard GCN [22] model to learn the node embeddings.
- **NGCF [35]** adds second-order interactions upon the neural passing based GNN model [10].
- **LightGCN [17]:** devises the light graph convolution upon NGCF.
- **HHGR [44]:** designs coarse- and fine-grained node dropout strategies upon the hypergraph for group recommendation.

We discard potential baselines like Popularity [8], COM [42], and CrowdRec [27], since previous works [4, 19, 29, 31, 39] have validated the superiority over the compared ones. For the GNN model GCMC, NGCF and LightGCN, we extend it to address group recommendation as proposed in Section 2.2; Besides, we use notation GNN* to denote the corresponding proposed model SGG. We further evaluate three variants of SGG, named Basic-GNN, Meta-GNN and CL-GNN, which are equipped with basic embedding reconstruction with GNNs (Section 3.1), meta aggregator (Section 3.2) and only CL adapter (Section 3.3), respectively.

4.1.3 Training Settings. We present the details of dataset segmentation, model training process and hyper-parameter settings.

Dataset Segmentation. We first select the groups/users with sufficient interactions as the target groups/users in the meta-training set D_T^g/D_T^u , and leave the rest groups/users in the meta-test set D_N^g/D_N^u , as we need the true embeddings of groups/users inferred

from the sufficient interactions. In order to avoid information leakage, we further select items with sufficient interactions from D_T^g/D_T^u , and obtain the meta-training set D_T^i . For simplicity, we use D_T and D_N to denote these meta-training and meta-test sets. For each group/user in D_N , we further select top $c\%$ of its/his interacted items in chronological order into the training set $Train_N$, and leave the rest items into the test set $Test_N$.

For addressing the occasional groups, we divide the groups with the number of direct interacted items more than n_g into D_T and leave the rest groups into D_N . We select n_g as 10 for Weeplaces and Douban. Similarly, we divide the users (or items) with the number of direct interacted items (users) more than n_u (n_i) into D_T and leave the rest users (items) into D_N , where both n_u and n_i is set as 10 for Weeplaces and Douban. In CAMRa2011, since the groups, users and items have abundant interactions, we randomly select 70% groups, users and items in D_T and leave the rest in D_N . For each group and user in D_N , we only keep top 10 interacted items in chronological order to simulate the real cold-start scenario. Similarly, for each item in D_N , we only keep its first 5 interacted groups/users.

Model Training Process. We train each of the baseline methods to obtain the ground-truth embeddings on D_T , as these methods are good enough to learn high-quality embeddings from the abundant interactions. For MoSAN, AGREE, SIGR, GroupIM and GAME, we directly fetch the trained embeddings as ground-truth embeddings; For HHGR, GCMC, NGCF and LightGCN, we combine the embeddings obtained at each layer to form the ground-truth embeddings, take the group embedding as an example, $\mathbf{h}_g = \mathbf{h}_g^0 + \dots + \mathbf{h}_g^L$. User and item embeddings can be explained in the similar way.

The SSL tasks is trained on D_T , while the recommendation task is trained on D_T and $Train_N$. Both the SSL and the recommendation tasks are evaluated in $Test_N$. We adopt Recall@ \mathcal{K} and NDCG@ \mathcal{K} as the metrics to evaluate the items ranked by the relevance scores.

Hyper-parameter Settings. For fair comparison, all models are trained from scratch which are initialized with the Xavier method [11]. The learning rate is 0.001 and mini-batch size is 256. We tune K , L , $c\%$ within the ranges of {3, 4, 5, 6, 7, 8, 9, 10, 11, 12}, {1,2,3,4} and {0.1, 0.2, 0.3}, respectively. We tune λ_1 with the ranges of {0.01, 0.1, 0.5, 1.0, 1.2}, and empirically set λ and λ_2 as 1 and 1e-6, respectively. We tune c_u and c_g with the ranges of {10, 20, 30}. By default, we set L as 3, K as 5, $c\%$ as 0.1, τ as 0.2, c_u and c_g as 20, and \mathcal{K} as 20.

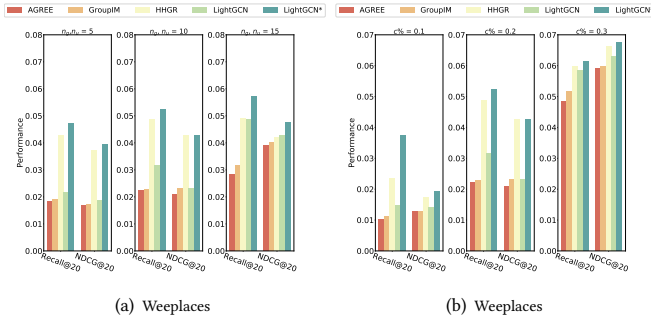
4.2 Performance Comparison (RQ1)

4.2.1 Overall Performance Comparison. We report the overall recommendation performance in Table 3. The results show that compared with other baselines, our proposed SGG (denoted as GNN*) significantly improves the recommendation performance, which indicates the proposed SSL tasks are useful to learn high-quality embeddings, and can further benefit the recommendation task. Besides, SGG is better than the most competitive baseline method HHGR, which indicates the superiority of the proposed SSL tasks in dealing with high-order cold-start neighbors.

4.2.2 Interacted Number and Sparse Rate Analysis. It is still unclear how does SGG handle the cold-start groups and users with different interacted items (n_g and n_u) and different sparse rate $c\%$. To this end,

Table 3: Overall recommendation performance with sparse rate $c\%=0.1$, layer size $L=3$ and neighbor size $K=5$.

Methods	Weeplaces		CAMRa2011		Douban	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
MoSAN	0.0223	0.0208	0.0214	0.0166	0.0023	0.0019
AGREE	0.0266	0.0233	0.0237	0.0168	0.0024	0.0018
SIGR	0.0276	0.0223	0.0278	0.0169	0.0028	0.0021
GroupIM	0.0228	0.0283	0.0277	0.0169	0.0034	0.0026
GAME	0.0283	0.0216	0.0499	0.0173	0.0031	0.0027
GCMC	0.0312	0.0083	0.0348	0.0171	0.0036	0.0025
NGCF	0.0336	0.0093	0.0288	0.0177	0.0043	0.0028
LightGCN	0.0316	0.0233	0.1036	0.0183	0.0118	0.0032
HHGR	0.0488	0.0422	0.1494	0.0376	0.0154	0.0045
GCMC*	0.0513	0.0448	0.1112	0.0394	0.0053	0.0038
NGCF*	0.0486	0.0413	0.1256	0.0342	0.0133	0.0043
LightGCN*	0.0523	0.0426	0.1634	0.0353	0.0237	0.0063

**Figure 3: Recommendation performance under different interacted numbers n_u and n_g (shown in Fig 3(a)), and under different sparse rate $c\%$ (shown in Fig 3(b)). Results on CAMRa2011 and Douban show the same trend which are omitted for space.**

we change n_g and n_u in the range of $\{5, 10, 15\}$ while keeping $c\%$ as 0.1, L as 3 and K as 5; and change $c\%$ in the range of $\{0.1, 0.2, 0.3\}$ while keeping n_g and n_u as 5, L as 3 and K as 5. We compare our proposed model SGG (denoted as LightGCN*, in which we select LightGCN as the backbone GNN model) with competitive baseline methods AGREE, GroupIM, HHGR and LightGCN, and report the recommendation performance in Figure 3. The smaller n_g , n_u and $c\%$ are, the groups and users in D_N have fewer interactions. Based on the results, we find that: (1) LightGCN* is consistently superior to all the other baselines, which justifies the superiority of SGG in handling cold-start recommendation with different n_g , n_u and $c\%$. (2) When n_g and n_u decrease from 15 to 5, and when $c\%$ decreases from 0.3 to 0.1, SGG all has a larger improvement compared with other baselines, which verifies its capability to solve the cold-start groups/users with extremely sparse interactions.

4.3 Ablation Study (RQ2)

It is still not clear which part of the pretext tasks is responsible for the good performance in SGG. To answer this question, we conduct an ablation study to investigate the recommendation performance of SGG and its variant models in Table 4. We find that: (1) Basic-GNN, Meta-GNN and CL-GNN is consistently superior than the vanilla GNNs, which indicates the effectiveness of the proposed

Table 4: The comparison of different SGG variants with sparse rate $c\%=0.1$, layer size $L=3$ and neighbor size $K=5$.

Methods	Weeplaces		CAMRa2011		Douban	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
GCMC	0.0312	0.0083	0.0348	0.0171	0.0036	0.0025
CL-GCMC	0.0333	0.0127	0.0375	0.0216	0.0038	0.0026
Basic-GCMC	0.0412	0.0293	0.0561	0.0278	0.0045	0.0028
Meta-GCMC	0.0441	0.0328	0.0826	0.0319	0.0048	0.0031
GCMC*	0.0513	0.0448	0.1112	0.0394	0.0053	0.0038
NGCF	0.0336	0.0093	0.0288	0.0177	0.0043	0.0028
CL-NGCF	0.0357	0.0116	0.0310	0.0129	0.0048	0.0028
Basic-NGCF	0.0390	0.0241	0.0971	0.0233	0.0068	0.0041
Meta-NGCF	0.0480	0.0382	0.1172	0.0319	0.0121	0.0042
NGCF*	0.0486	0.0413	0.1256	0.0342	0.0133	0.0043
LightGCN	0.0316	0.0233	0.1036	0.0183	0.0118	0.0032
CL-LightGCN	0.0310	0.0243	0.1053	0.0235	0.0136	0.0033
Basic-LightGCN	0.0373	0.0318	0.1252	0.0210	0.0181	0.0038
Meta-LightGCN	0.0475	0.0415	0.1556	0.0312	0.0232	0.0049
LightGCN*	0.0523	0.0426	0.1634	0.0353	0.0237	0.0063

Table 5: Recommendation performance, training time per epoch and convergent epochs w/o meta-learning setting.

Dataset	Weeplace				CAMRa2011			
Method	Recall	NDCG	Time	Epoch	Recall	NDCG	Time	Epoch
GCMC	0.0312	0.0083	188.6s	31	0.0348	0.0171	51.8s	30
GCMC*-M	0.0509	0.0453	721.3s	30	0.1021	0.0382	172.6s	22
GCMC*	0.0513	0.0448	499.6s	12	0.1112	0.0394	112.3s	10
NGCF	0.0336	0.0093	182.6s	30	0.0288	0.0177	58.7s	26
NGCF*-M	0.0465	0.0403	700.2s	20	0.1123	0.0325	166.3s	13
NGCF*	0.0486	0.0413	489.7s	8	0.1256	0.0342	100.8s	8
LightGCN	0.316	0.0233	179.2s	30	0.1036	0.0183	48.3s	30
LightGCN*-M	0.0511	0.0402	683.6s	18	0.1435	0.0329	153.8s	18
LightGCN*	0.0523	0.0426	483.4s	10	0.1634	0.0353	99.3s	6

SSL tasks. (2) Among all the variant models, Meta-GNN performs the best, which indicates enhancing the cold-start neighbors' embedding quality is much more important. (3) GNN* performs the best, which verifies the superiority of combining these SSL tasks.

4.4 Study of SGG (RQ3)

4.4.1 Effectiveness of Meta-Learning Setting. As mentioned in Section 3.1, we train SGG under the meta-learning setting. To explore whether the meta-learning setting can benefit the recommendation performance and have satisfactory time complexity, we compare SGG and the vanilla GNN model with a variant model SGG-M, which removes the meta-learning setting. More Concretely, in SGG-M, for each group/user/item, we do not sample K neighbors, but instead directly using their first-order and high-order neighbors to perform graph convolution. We report the average recommendation performance, the average training time in each epoch and the average convergent epoch in Table 5. Based on the results, we find that SGG is consistently superior than SGG-M, and has much more smaller training time in each epoch, much faster converges speed. This indicates training SGG in the meta-learning setting can not only improve the model performance, but also improve the training efficiency and make the model easily and rapidly being adapted to new occasional groups.

4.4.2 Sensitive of Ground-truth Embedding. As mentioned in Section 3.1, when performing embedding reconstruction with GNNs, we select any group recommendation models to learn the ground-truth embeddings. One may consider whether the recommendation performance is sensitive to the ground-truth embeddings obtained

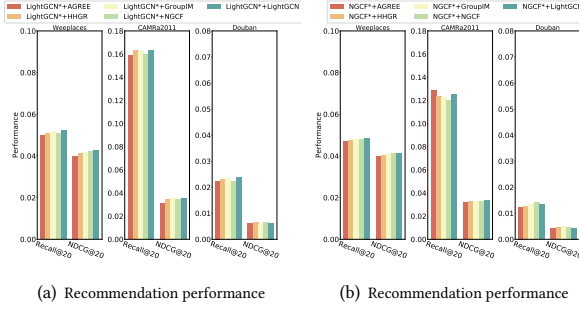


Figure 4: Sensitive analysis of ground-truth embeddings.

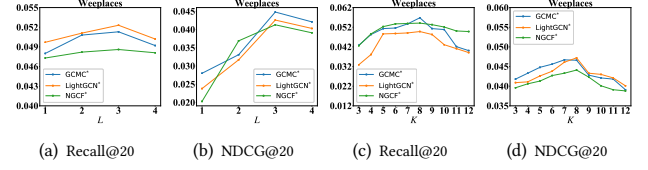
Table 6: Recommendation performance, training time for each epoch and convergent epochs under the multi-task learning or pre-training paradigms.

Dataset	Weeplaces				CAMRa2011			
	Recall	NDCG	Time	Epoch	Recall	NDCG	Time	Epoch
GCMC*-P	0.0428	0.0409	201.0s	28	0.1008	0.0317	56.29s	26
GCMC*	0.0513	0.0448	499.6s	12	0.1112	0.0394	112.3s	10
NGCF*-P	0.0411	0.0388	203.1s	28	0.1182	0.0318	60.2s	28
NGCF*	0.0486	0.0413	489.7s	8	0.1256	0.0342	100.8s	8
LightGCN*-P	0.0487	0.0386	189.3s	28	0.1525	0.0327	50.1s	28
LightGCN*	0.0523	0.0426	483.4s	10	0.1634	0.0353	99.3s	6

by different models. To this end, we use competitive baselines to learn the ground-truth embeddings as proposed in Section 4.1.3, and report the performance of NGCF* and LightGCN* in Figure 4. Notation NGCF*-AGREE denotes SGG is equipped with the ground-truth embeddings, which are obtained by AGREE. Other notations are defined in a similar way. The results show that all the models that equipped with different ground-truth embeddings achieve almost the same performance, which indicates SGG is not sensitive to the ground-truth embeddings, as the baselines are good enough to learn high-quality embeddings from the abundant interactions.

4.4.3 Multi-task Learning Vs Pre-training. Here we would like to answer the question in Section 3.4: Can the recommendation performance benefit from the pre-training or the multi-task learning paradigm? Towards this goal, we first pre-train the SSL tasks on D_T to obtain the model parameters, use them to initialize SGG, and then fine-tune SGG on D_N via optimizing the main task. We term this variant as SGG-P. We report the recommendation performance, the average training time in each epoch and the average convergent epoch in Table 6. Based on the results, we find that: (1) SGG-P performs worse than SGG, but still better than other baselines (cf. Table 3). As jointly training in SGG admits that the representations in the main and SSL tasks are mutually enhanced with each other, which is superior than only offer a better initialization for the GNNs in SGG-P. This finding is consistent with the observation in previous studies [36]. (2) Compared with SGG-P, SGG has faster convergence speed. Although SGG-P has smaller training time per epoch, it still has larger total training time than SGG. This verifies the multi-task learning paradigm can speed up the model convergence.

4.4.4 Hyper-parameter analysis. We move on to study different designs of the layer depth L and the neighbor size K . We select LightGCN*, NGCF* and GCMC*, and report their performances

Figure 5: Recommendation performance under different layer L (Fig. 5(a) and Fig. 5(b)), and under different neighbor size K (Fig. 5(c) and Fig. 5(d)). Results on CAMRa2011 and Douban show the same trend with Weeplaces which are omitted for space.

under different layer depth L in Figure 5(a) and Figure 5(b), under different neighbor size K in Figure 5(c) and Figure 5(d). The results show that: (1) The performance first increases and then drops when increasing L from 1 to 4. The peak point is 3 at most cases. This indicates GNN can only capture short-range dependencies, which is consistent with LightGCN's [17] finding. (2) The performance first increases and then drops when increasing K from 3 to 12. The peak point is 8 at most cases. This indicates incorporating proper size of neighbors can benefit the recommendation task.

5 RELATED WORK

5.1 Group Recommendation

Existing works on group recommendation can be generally divided into two categories: score aggregation and profile aggregation.

Score Aggregation. This strategy pre-defines a scoring function to obtain the preference score of all members in a group on the target item. The scoring functions include average [2], least misery [1] and maximum satisfaction [3]. However, due to the static recommendation process of the predefined functions, these methods easily fall into local optimal solutions.

Profile Aggregation. This strategy aggregates the profiles of group members and feeds the fused group profile into individual recommendation models. Essentially, probabilistic generative models and deep learning based models are proposed to aggregate the group profile. The generative model first selects group members for a target group, and then generates items based on the selected members and their associated hidden topics [24, 38, 42]. The deep learning based model conducts attention mechanism to assign each user an attention weight, which denotes the influence of group member in deciding the group's choice on the target item [4, 5, 39]. However, both of these two methods suffer from the data sparsity issue. Recently, researches propose GNN-based recommendation models, which incorporate high-order collaborative signals in the built graph [13, 19, 34] or hypergraph [12, 41]. Moreover, Zhang et al. [44] propose hypergraph convolution network (HHGR) with self-supervised node dropout strategy to alleviate the data sparsity issue. However, the GNNs still can not deal with the cold-start neighbors when performing graph convolution, and do not explicitly capture the correlations between the group and non-group members. Motivated by the SSL technique, we propose to jointly reconstruct the group/user/item embeddings under the meta-learning setting, and

further incorporate a meta aggregator and a CL adapter to improve the embedding quality.

6 CONCLUSION

We design a new self-supervised graph learning paradigm for group recommendation, which trains the GNN model to reconstruct the group embedding under the meta-learning setting. To deal with the cold-start neighbors during the graph convolution process, we further introduce a meta aggregator to enhance the aggregation ability of each graph convolution step. To explicitly consider the correlations between the group and non-group members, we further propose a CL adapter to regularize the group and user embeddings. Experimental results demonstrate the superiority of our proposed model against the state-of-the-art group recommendation models.

REFERENCES

- [1] Sihem Amer-Yahia, Senjuti Basu Roy, Ashish Chawla, Gautam Das, and Cong Yu. 2009. Group Recommendation: Semantics and Efficiency. *Proc. VLDB Endow.* 2, 1 (2009), 754–765.
- [2] Linas Baltrunas, Tadas Makcinskas, and Francesco Ricci. 2010. Group Recommendations with Rank Aggregation and Collaborative Filtering. In *RecSys '10*. 119–126.
- [3] Ludovico Boratto and Salvatore Carta. 2011. State-of-the-Art in Group Recommendation and New Approaches for Automatic Identification of Groups. In *Information Retrieval and Mining in Distributed Environments*. Vol. 324. 1–20.
- [4] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang Hong. 2018. Attentive Group Recommendation. In *SIGIR '18*. ACM, 645–654.
- [5] Da Cao, Xiangnan He, Lianhai Miao, Guangyi Xiao, Hao Chen, and Jia Xu. 2021. Social-Enhanced Attentive Group Recommendation. *IEEE Trans. Knowl. Data Eng.* 33, 3 (2021), 1195–1209.
- [6] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling. In *ICLR '18*. OpenReview.net.
- [7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML '20 (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 1597–1607.
- [8] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *RecSys '10*. ACM, 39–46.
- [9] Li Gao, Jia Wu, Zhi Qiao, Chuan Zhou, Hong Yang, and Yue Hu. 2016. Collaborative Social Group Influence for Event Recommendation. In *CIKM '16*. ACM, 1941–1944.
- [10] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural Message Passing for Quantum Chemistry. In *ICML '17*. Vol. 70. PMLR, 1263–1272.
- [11] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS '10 (JMLR Proceedings, Vol. 9)*. JMLR.org, 249–256.
- [12] Lei Guo, Hongzhi Yin, Tong Chen, Xiangliang Zhang, and Kai Zheng. 2021. Hierarchical Hyperedge Embedding-based Representation Learning for Group Recommendation. *ACM TOIS* (2021).
- [13] Lei Guo, Hongzhi Yin, Qinyong Wang, Bin Cui, Zi Huang, and Lizhen Cui. 2020. Group Recommendation with Latent Voting Mechanism. In *ICDE '20*. IEEE, 121–132.
- [14] William L. Hamilton, Zitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NeurIPS '17*. 1024–1034.
- [15] Bowen Hao, Jing Zhang, Cuiping Li, Hong Chen, and Hongzhi Yin. 2020. Recommending Courses in MOOCs for Jobs: An Auto Weak Supervision Approach. In *ECML-PKDD '20 (Lecture Notes in Computer Science, Vol. 12460)*. Springer, 36–51.
- [16] Bowen Hao, Jing Zhang, Hongzhi Yin, Cuiping Li, and Hong Chen. 2021. Pre-Training Graph Neural Networks for Cold-Start Users and Items Representation. In *WSDM '21*. 265–273.
- [17] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR '20*. ACM, 639–648.
- [18] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Trans. Knowl. Data Eng.* 30, 12 (2018), 2354–2366.
- [19] Zhixiang He, Chi-Yin Chow, and Jia-Dong Zhang. 2020. GAME: Learning Graphical and Attentive Multi-view Embeddings for Occasional Group Recommendation. In *SIGIR '20*. ACM, 649–658.
- [20] Ziniu Hu, Ting Chen, Kai-Wei Chang, and Yizhou Sun. 2019. Few-Shot Representation Learning for Out-Of-Vocabulary Words. In *ACL '19*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, 4102–4112.
- [21] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020. GPT-GNN: Generative Pre-Training of Graph Neural Networks. In *SIGKDD '20*. ACM, 1857–1867.
- [22] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR '17*. OpenReview.net.
- [23] Chenghao Liu, Xin Wang, Tao Lu, Wenwu Zhu, Jianling Sun, and Steven C. H. Hoi. 2019. Discrete Social Recommendation. In *AAAI '19*. AAAI Press, 208–215.
- [24] Xingjie Liu, Yuan Tian, Mao Ye, and Wang-Chien Lee. 2012. Exploring personal impact for group recommendation. In *CIKM '12*. ACM, 674–683.
- [25] Toon De Pessemier, Simon Doooms, and Luc Martens. 2014. Comparison of group recommendation algorithms. *Multim. Tools Appl.* 72, 3 (2014), 2497–2541.
- [26] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In *SIGKDD '20*. ACM, 1150–1160.
- [27] Vineeth Rakesh, Wang-Chien Lee, and Chandan K. Reddy. 2016. Probabilistic Group Recommendation Model for Crowdfunding Domains. In *WSDM '16*. ACM, 257–266.
- [28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18–21, 2009*. AUAI Press, 452–461.
- [29] Aravind Sankar, Yanhong Wu, Yuhang Wu, Wei Zhang, Hao Yang, and Hari Sundaram. 2020. GroupIM: A Mutual Information Maximization Framework for Neural Group Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25–30, 2020*. ACM, 1279–1288.
- [30] Peijie Sun, Le Wu, and Meng Wang. 2018. Attentive Recurrent Social Recommendation. In *SIGIR '18*. ACM, 185–194.
- [31] Lucas Vinh Tran, Tuan-Anh Nguyen Pham, Yi Tay, Yiding Liu, Gao Cong, and Xiaoli Li. 2019. Interact and Decide: Medley of Sub- on Networks for Effective Group Recommendation. In *SIGIR '19*. ACM, 255–264.
- [32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NeurIPS '17*. 5998–6008.
- [33] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching Networks for One Shot Learning. In *NeurIPS '16*, Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (Eds.). 3630–3638.
- [34] Wen Wang, Wei Zhang, Jun Rao, Zhijie Qiu, Bo Zhang, Leyu Lin, and Hongyuan Zha. 2020. Group-Aware Long- and Short-Term Graph Representation Learning for Sequential Group Recommendation. In *SIGIR '20*. ACM, 1449–1458.
- [35] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR '19*. ACM, 165–174.
- [36] 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*. ACM, 726–735.
- [37] Yuxin Wu, Hanxiao Liu, and Yiming Yang. 2018. Graph Convolutional Matrix Completion for Bipartite Edge Prediction. In *KDD '18*. SciTePress, 49–58.
- [38] Mao Ye, Xingjie Liu, and Wang-Chien Lee. 2012. Exploring social influence for recommendation: a generative model approach. In *SIGIR '12*. ACM, 671–680.
- [39] Hongzhi Yin, Qinyong Wang, Kai Zheng, Zhixu Li, Jiali Yang, and Xiaofang Zhou. 2019. Social Influence-Based Group Representation Learning for Group Recommendation. In *ICDE '19*. IEEE, 566–577.
- [40] Hongzhi Yin, Lei Zou, Quoc Viet Hung Nguyen, Zi Huang, and Xiaofang Zhou. 2018. Joint Event-Partner Recommendation in Event-Based Social Networks. In *ICDE '18*. IEEE Computer Society, 929–940.
- [41] Junliang Yu, Hongzhi Yin, Jundong Li, Qinyong Wang, Nguyen Quoc Viet Hung, and Xiangliang Zhang. 2021. Self-Supervised Multi-Channel Hypergraph Convolutional Network for Social Recommendation. In *WWW '21*.
- [42] Quan Yuan, Gao Cong, and Chin-Yew Lin. 2014. COM: a generative model for group recommendation. In *KDD '14*. ACM, 163–172.
- [43] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. 2017. Deep Sets. In *NeurIPS '17*. 3391–3401.
- [44] Junwei Zhang, Min Gao, Junliang Yu, Lei Guo, Jundong Li, and Hongzhi Yin. 2021. Double-Scale Self-Supervised Hypergraph Convolutional Network for Group Recommendation. In *CIKM '21*. ACM.

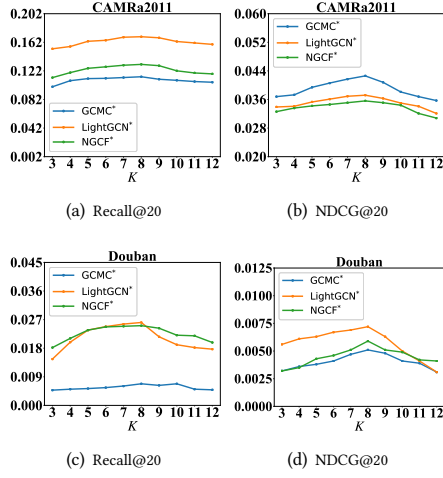


Figure 7: Recommendation performance under different neighbor size K . $c\%=0.1$ and $L=3$.

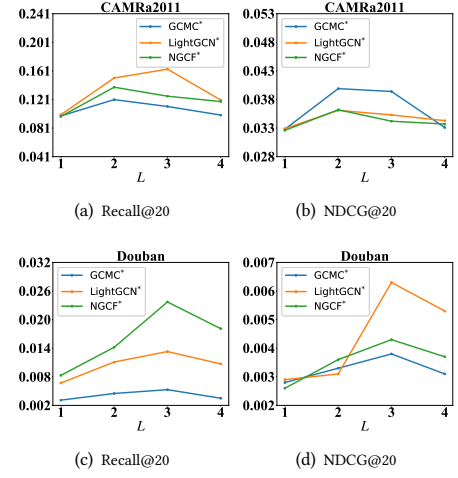


Figure 6: Recommendation performance under different layer L . $c\%=0.1$ and $K=5$.

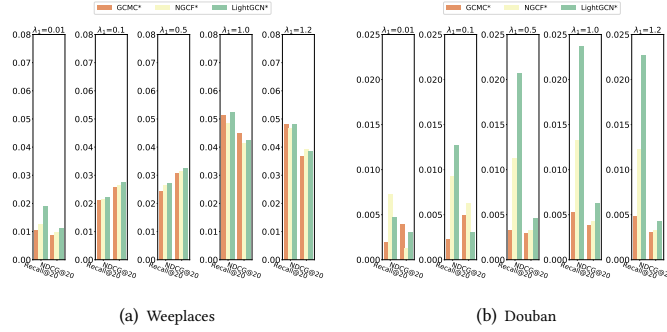
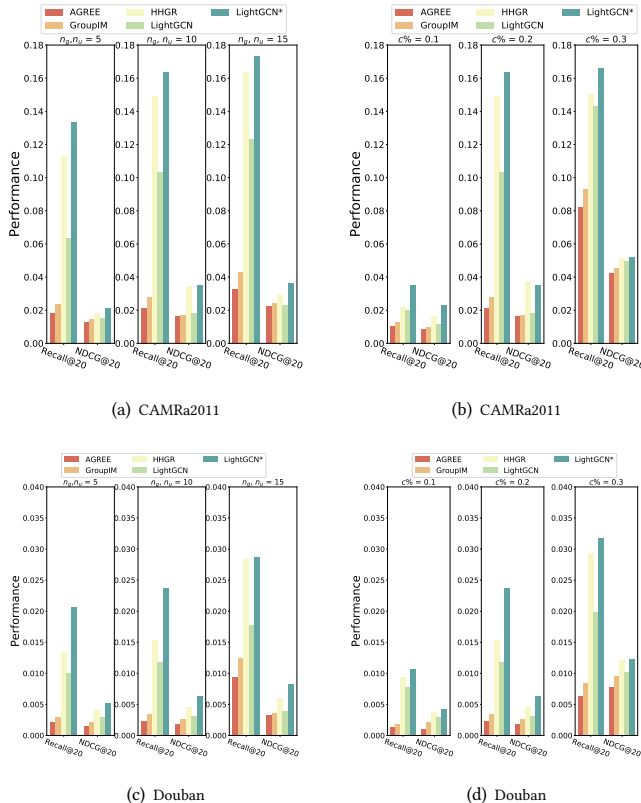


Figure 8: Recommendation performance under different balancing parameter λ_1 . $K=5$, $c\%=0.1$ and $L=3$.



A APPENDIX

A.1 Interacted Number and Sparse Rate Analysis

We report the recommendation performance under different interacted number and sparse rate on CAMRa2011 and Douban in Figure 9. The results are consistent with the findings in Section 4.2.2.

A.2 Hyper-parameter analysis.

- In terms of the balancing parameter λ_1 , we report the recommendation performance of LightGCN*, NGCF* and GCMC* on CAMRa2011 and Douban in Figure 8. The results show the performance first increases and then drops when increasing λ_1 from 0.01 to 1.2. The peak point is 1 at all cases. This indicates the auxiliary SSL tasks are as important as the main recommendation task.
- In terms of the layer depth L , we report the recommendation performance of LightGCN*, NGCF* and GCMC* on CAMRa2011 and Douban in Figure 6. The results are consistent with the findings of Section 4.4.4.
- In terms of the neighbor size K , we report the recommendation performance of LightGCN*, NGCF* and GCMC* on CAMRa2011

and Douban in Figure 7. The results are consistent with the findings of Section 4.4.4.