Group-Aware Long- and Short-Term Graph Representation Learning for Sequential Group Recommendation

Wen Wang[1,2], Wei Zhang[1*], Jun Rao[2], Zhijie Qiu[2], Bo Zhang[2], Leyu Lin[2], Hongyuan Zha[3]
[1]East China Normal University
[2]WeChat Search Application Department, Tencent
[3]Georgia Institute of Technology
51164500120@stu.ecnu.edu.cn,zhangwei.thu2011@gmail.com,
{jamesrao,jackqiu,nevinzhang,goshawklin}@tencent.com
zha@cc.gatech.edu

ABSTRACT

Sequential recommendation and group recommendation are two important branches in the field of recommender system. While considerable efforts have been devoted to these two branches in an independent way, we combine them by proposing the novel sequential group recommendation problem which enables modeling group dynamic representations and is crucial for achieving better group recommendation performance. The major challenge of the problem is how to effectively learn dynamic group representations based on the sequential user-item interactions of group members in the past time frames. To address this, we devise a Group-aware Long- and Short-term Graph Representation Learning approach, namely GLS-GRL, for sequential group recommendation. Specifically, for a target group, we construct a group-aware long-term graph to capture user-item interactions and item-item co-occurrence in the whole history, and a group-aware short-term graph to contain the same information regarding only the current time frame. Based on the graphs, GLS-GRL performs graph representation learning to obtain long-term and short-term user representations, and further adaptively fuse them to gain integrated user representations. Finally, group representations are obtained by a constrained user-interacted attention mechanism which encodes the correlations between group members. Comprehensive experiments demonstrate that GLS-GRL achieves better performance than several strong alternatives coming from sequential recommendation and group recommendation methods, validating the effectiveness of the core components in GLS-GRL.

CCS CONCEPTS

• **Information systems → Recommender systems**; *Personalization*; Temporal data.

*Corresponding author and having equal contribution with the first author.

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 the author(s) 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.
SIGIR '20, July 25–30, 2020, Virtual Event, China
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401136

KEYWORDS

Sequential Group Recommendation; Graph Representation Learning; User Modeling

ACM Reference Format:
Wen Wang[1,2], Wei Zhang[1*], Jun Rao[2], Zhijie Qiu[2], Bo Zhang[2], Leyu Lin[2], Hongyuan Zha[3]. 2020. Group-Aware Long- and Short-Term Graph Representation Learning for Sequential Group Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401136

1 INTRODUCTION

Modern recommender systems play a pivotal role in shaping the way of user-item interactions. This is especially crucial in the information explosion era where users commonly suffer from the information overload issue (e.g., millions of products in Amazon and large-scale streaming user-generated content in Twitter). As a result, large research efforts have been devoted to developing effective recommendation models and algorithms for different recommendation problem settings. Sequential recommendation [17, 33] is a hot research problem in this regard, which aims to predict the next item that a target user prefers to interact with. It poses a major challenge of learning dynamic user preference representations based on sequential user-item interactions. Recurrent neural networks (RNNs) [8], convolutional neural networks (CNNs) [19], powerful attention mechanism [10], and recently graph neural networks (GNNs) [26] have been applied to this problem.

From another perspective, some studies investigate the situations where recommended items are served to a target group of users, instead of an individual user in general sequential recommendation. This problem is called group recommendation [15]. The form of user groups is widely prevalent in online social medias, such as Meetup[1] where users are organized as groups to participate some offline activities, Facebook[2] in which groups are interest clubs, and WeChat[3] wherein users easily create groups for chatting. Recent studies for group recommendation center on how to automatically quantify the relative importance of individual preferences in forming group-level preference, beyond using empirical strategies such as averaging preference scores of group members [2]. Two representative technique branches are probabilistic models [13] and attention-based approaches [3]. Nevertheless, researches to date

[1]https://www.meetup.com/
[2]https://www.facebook.com/groups/
[3]https://www.wechat.com/

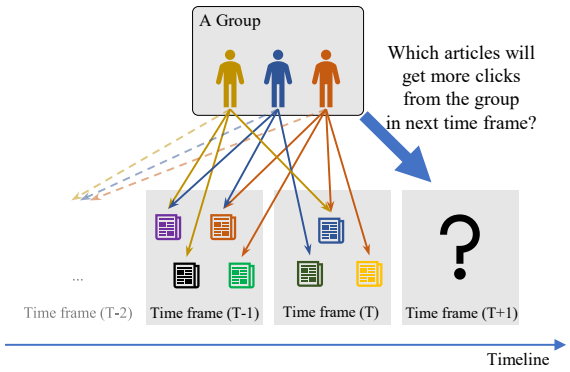


Figure 1: Illustration of sequential group recommendation with news articles as items. Arrow lines denote the interactions between items and group members.

formulate the problem as static recommendation, overlooking the sequential nature of group members' behaviors.

In this paper, we formulate a novel problem called Sequential Group Recommendation (SGR) which lies at the intersection of sequential recommendation and group recommendation. As Figure 1 shows, this problem aims to leverage the sequential item interactions of the target group members in the past time frames to predict which items will get more interactions from the target group members in the next time frame. Since new groups formed by existing users may emerge on social platforms from time to time, the ability of recommending items to both existing groups and new groups is required by the problem. Compared to the previous group recommendation problem settings, sequential group recommendation is very instrumental, enabling modeling group dynamic representations inspired by sequential recommendation and be promising for boosting group recommendation performance. Note that although the studies [14, 27] involve the concepts of sequential recommendation and groups, they are fundamentally different from our study because: (1) the former study is tailored for conversational recommendation and its method is fully empirical, without having a model learning process; (2) the latter actually utilizes group preference to facilitate the sequential recommendation for individual users, aiming to overcome the sparsity issue.

To deal with this problem, a fundamental challenge is how to effectively learn dynamic group representations based on the sequential user-item interactions of group members in the past time frames. In actuality, the group representations are reflected by the group members' dynamic representations, which is suitable for both existing groups and new groups. As such, user representations are indispensable, bridging the gap between specific user-item interactions and group representations. Then the overall fundamental challenge is decomposed into specific challenges:

- How to leverage group membership and sequential user-item interactions to learn user representations?
- How to utilize the obtained user representations to represent group preference?

In order to handle the above-mentioned two challenges well, we propose the GLS-GRL model (short for Group-aware Long- and Short-term Graph Representation Learning). It is welcomed by its ability of enabling group membership to impact user representation learning and achieving group representations based on the obtained user representations. To be specific, in each time frame, we first build group-aware long- and short-term graphs, both of which share all the users belonging to the same target group. The long-term graph contains user-item interactions and item-item co-occurrence in the whole history, while the short-term graph contains the information regarding only the current time frame. Based on the two graphs, GLS-GRL employs graph representation learning on the two graphs to learn user long- and short-term representations, respectively. These two types of representations are fused by a simple gating mechanism to obtain integrated user representations. In this way, the first challenge is addressed. For the second challenge, GLS-GRL further develops a constrained user-interacted attention inspired by the sub-attention network [20]. It encodes the correlations between group members by representing a user w.r.t. the representations of other selected group members who are required to have at least one co-interacted item. The group representations are finally achieved by integrating the user representations.

To sum up, the main contributions are as follows:

- We formulate a novel problem named sequential group recommendation which requires to model the sequential dynamics of group representations overlooked by existing studies for group recommendation.
- We develop the model GLS-GRL with the innovations of learning long- and short-term user representations through the corresponding group-aware long- and short-term graphs, as well as the coupling of group representation learning and user representation learning.
- We conduct comprehensive experiments on two real-world datasets and demonstrate GLS-GRL achieves superior performance compared to strong alternatives, validating some key designs of the model.

2 RELATED WORK

In this section, we review relevant studies from three aspects: sequential recommendation, group recommendation, and GNNs for recommendation.

2.1 Sequential Recommendation

In contrast to general recommendation task settings, sequential recommendation has its characteristics in addressing the sequence nature of user behaviors and predicting what users will prefer in the near future (e.g., next time frame). Early studies on sequential recommendation rely on first-order Markov assumption that the next interaction only depends on the current interaction of the same user, including transition counting-based methods [5] and latent factor models [17]. Inspired by the widespread success of deep learning, recent years have witnessed the applications of RNNs, CNNs, attention mechanism, and GNNs to sequential recommendation. Specifically, the pioneering study [8] leverages RNNs in this domain by regarding interacted items as words. CNNs [19], the infrastructure for image processing, are also verified to be effective in this regard to a certain extent, by mapping item sequences to embedding matrices. To quantify the different importance of

past interactions on the next prediction, attention mechanism [10–12, 18] is adopted. From a different perspective, GNNs model the interactions as a graph, which will be discussed in Section 2.3.

Most of the existing sequential recommendation approaches serve for an individual user. A straightforward idea is to directly utilize these methods to learn user representations and then aggregate them through some fusion approaches. However, it cannot leverage group membership to gain the user representations, as emphasized in the first challenge. It is worth noting that we could not map group IDs to embeddings for enhancing input representations since new groups that do not appear in a training stage should be handled.

2.2 Group Recommendation

Group recommendation requires to fuse individual preferences of all the members in a user group. To this end, some empirical and simple strategies are first adopted. O'Connor et al. [15] used the least satisfied user's preference to represent group-level preference (a.k.a. least misery strategy). [2] compares a few preference aggregation strategies, including the simple average aggregation strategy, and find: (1) the results of these strategies are similar, and (2) the group recommendation is harder than individual recommendation. Nevertheless, these strategies are a little too empirical without having a learning procedure to guide the aggregation.

To automatically measure the influence of individuals, probabilistic models [6, 13, 28, 31] are proposed to characterize item recommendation as a generation process. The basic procedure shared in these models is to first select a user for a target group (or members in the same group) and then generate items based on the user and associated hidden topics. However, they suffer a limitation that users' distributions overs topics or items, also regarded as user representations, are independent of groups [20]. Recently, deep representation learning-based models [3, 20, 29] are proposed. All of them leverage the attention mechanism [1] to calculate the individual influence weight w.r.t. specific group for effectively fusing user representations. It is demonstrated that they perform better than probabilistic models.

All the above group recommendation methods do not consider the sequence nature of user behaviors which constitute the dynamics of groups. This motivates the proposal of sequential group recommendation and the development of tailored models for the problem. Note that some studies [32] formulate group recommendation as recommending groups to a specific user to join, while some other researches [9, 27] utilize group information to promote the recommendation performance for individuals, both of which are conceptually different from the studied problem.

2.3 GNNs for Recommendation

Graph neural networks model the user-item interactions as graphs, which are welcomed for the ability of encoding high-order correlations into low-dimensional user and item representations. PinSage [30] is put forward for propagating representations on item-item graph through graph convolutional layers. NGCF [24] models user-item bipartite graph to learn from collective user behaviors. RippleNet [22] leverages knowledge graphs to propagate user preference revealed by the interacted items to candidate items that might be recommended. These methods are tailored for general recommendation settings and do not address the sequence nature of user behaviors.

In the domain of sequential recommendation, the latest studies [16, 23, 25, 26] model the current session of a target user as an item-item graph, or convert multiple sessions of different users into a globally shared item-item graph. The next item recommendation is achieved by calculating the similarities between candidate items and the overall representation of the session. The above studies conduct recommendation for individuals by capturing user short-term preference.

By contrast, we aim to capture both long- and short-term user preference for sequential group recommendation, i.e., predicting which items a target group will prefer in the next time frame. We also inherit the idea of GNN-based modeling by building group-aware long- and short-term graphs. This enables the user representation learning to be guided by both group membership and sequential user-item interactions.

3 PROBLEM FORMULATION

We utilize $U = \{u_1, u_2, ...u_{|U|}\}$ and $V = \{v_1, v_2, ..., v_{|V|}\}$ to denote the two most basic elements in recommender system, i.e., user set and item set, respectively. A user group g is associated with a user set $M_g \subseteq U$, which can be either an existing group or a newly emerged group. As such, the number of groups might change along with time. For each user $u \in U$, we denote the user's long-term history behaviors as $H_u^l = (v_0^l, v_1^l, v_2^l, ...)$ corresponding to the whole history, and short-term history behavior as $H_u^s = (v_0^s, v_1^s, v_2^s, ...)$ corresponding to the current time frame T. We further utilize $\Omega(H_u^l)$ $\left(\Omega(H_u^l) \subseteq V\right)$ and $\Omega(H_u^s)$ $\left(\Omega(H_u^s) \subseteq V\right)$ to denote the set of contained items in the history behaviors, respectively.

Based on these notations, we first define the group-aware long-term graph and group-aware short-term graph as follows:

Definition 1 (**Group-Aware Long-Term Graph**). *$\mathcal{G}_g^l = \{\mathcal{V}_g^l, \mathcal{E}_g^l\}$ is a group-aware long-term graph for the given group g if it satisfies: (1) $\mathcal{V}_g^l = \bigcup \Omega(H_u^l)|_{u \in M_g} \bigcup M_g$; (2) $\mathcal{E}_g^l$ contains user-item and item-item edges which are constructed by the later introduced graph construction procedure with $\{H_u^l\}_{u \in M_g}$ as input.*

Definition 2 (**Group-Aware Short-Term Graph**). *$\mathcal{G}_g^s = \{\mathcal{V}_g^s, \mathcal{E}_g^s\}$ is a group-aware short-term graph for the given group g if it satisfies: (1) $\mathcal{V}_g^s = \bigcup \Omega(H_u^s)|_{u \in M_g} \bigcup M_g$; (2) $\mathcal{E}_g^s$ contains user-item and item-item edges which are constructed by the later introduced graph construction procedure with $\{H_u^s\}_{u \in M_g}$ as input.*

The group-aware long-term graphs capture group members' long-term preference, while the group-aware short-term graphs pay more attention to their dynamic preferences. As such, the two types of the graphs are the cornerstones for near-future prediction. Formally, we define the sequential group recommendation problem as follows:

Problem 1 (**Sequential Group Recommendation**). *Given a target group g, and its group-aware long- and short-term graphs $\mathcal{G}_g^l$ and $\mathcal{G}_g^s$ depending on the current time frame T, this problem aims to learn a function $f(g, \mathcal{G}_g^l, \mathcal{G}_g^s, v) \rightarrow s_i^g$ for for a candidate item v,*

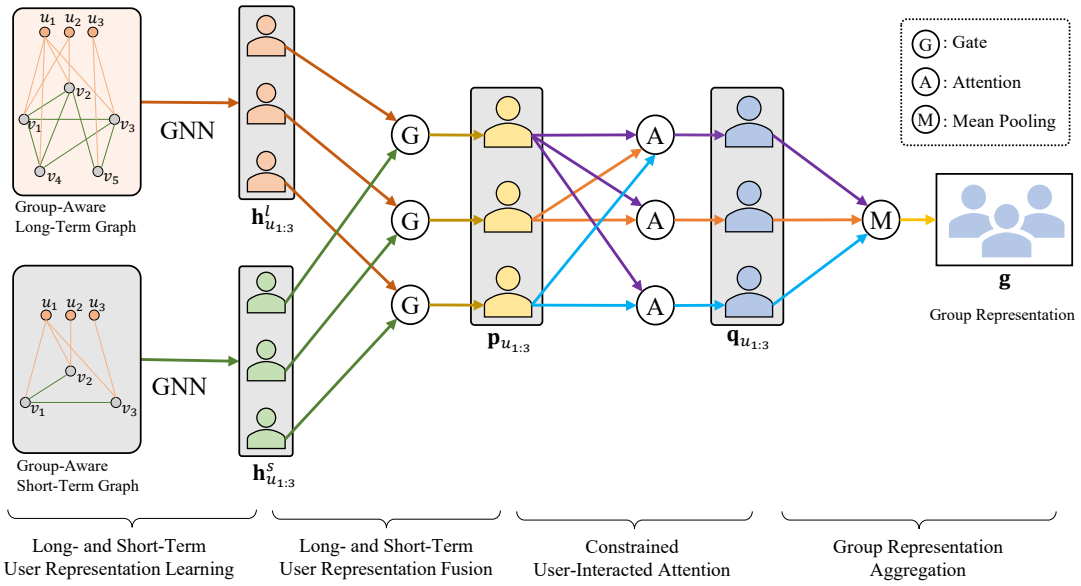


Figure 2: The architecture of GLS-GRL. The colors used in the left two components of the model correspond to the long- and short-term aspects, respectively. And each color used in the right part indicates a specific user.

where s_i^g denotes the group-level preference score for that item in the next time frame $T+1$.

Without causing ambiguity, we omit the superscripts and subscripts related to the group g for simplicity in the remaining of this paper.

4 PROPOSED MODEL

The overall architecture of our model GLS-GRL is shown in Figure 2. It takes a group's long-term graph and short-term graph as input, and outputs the group-level representation for similarity calculation with candidate items for recommendation. The long-term (or short-term) graph is constructed based on group members' interactions with items in the whole history (or in the current time frame). Figure 3 shows a toy example to illustrate the construction of the two graphs. GLS-GRL consists of four key components: (1) long- and short-term user representation learning, (2) long- and short-term user representation fusion, (3) constrained user-interacted attention, and (4) group representation aggregation. A hybrid ranking-based loss function is proposed to optimize GLS-GRL. Before elaborating the above key components and the objective function, we illustrate the graph construction procedure.

4.1 Group-Aware Graph Construction

In common sequential recommendation scenarios, users' sequential interactions with items are recorded, which are utilized for constructing group-aware long- and short-term graphs. We take an example, containing group members' behaviors (shown in 3(a)) and all users' behaviors (shown in 3(b)), to illustrate the procedure for graph construction. As defined in the Definition 1 and 2, the two graphs share the same user nodes and the item nodes are what

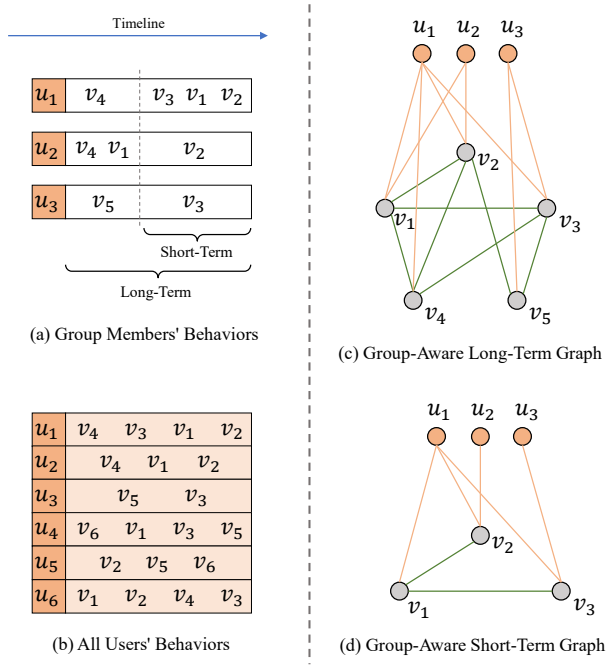


Figure 3: A toy example to illustrate graph construction.

the group members have interacted in the whole history (for the long-term graph) or in the current time frame (for the short-term graph). As such, the nodes in the graphs could be easily determined.

Then it comes to introduce how to construct edges in the two graphs. There are two types of edges in the constructed graphs, i.e., user-item edges and item-item edges. On the one hand, the user-item edges naturally capture the dynamic preference of group members. If a user has interacted with an item in the past, there will be an edge connecting the user and the item. As Figure 3(a) shows, user u_1 has interacted with items v_3, v_1, and v_2 in the current time frame, then there are edges between user u_1 and these items in the group-aware short-term graph, as shown in Figure 3(d). Similarly, since user u_3 has interactions with items v_5 and v_3 in the whole history, there are edges the user and these items in the group-aware long-term graphs, as shown in Figure 3(c). On the other hand, the item-item edges encode relations between different items, which are considered to be more stable than user preferences. Thus we construct item-item edges with a shared strategy for both of the two graphs. Specifically, if two items have been interacted with any user continuously, we assume there is an edge between the two items. For example, as shown in the behavior sequence of user u_1, since v_1 and v_2 are interacted with the user continuously, v_1 and v_2 should have an edge.

4.2 Long- and Short-Term User Representation Learning

Before learning to encode graph relations into low-dimensional user representations, the first step is to build input representations for users and items. To achieve this, we use $\bar{\mathbf{e}}_v \in \mathbb{R}^{|V|}$ and $\bar{\mathbf{f}}_u \in \mathbb{R}^{|U|}$ denote one-hot representations of each item $v \in V$ and each user $u \in U$. Afterwards, we use two trainable embedding matrices, $\mathbf{E} \in \mathbb{R}^{|V| \times d}$ and $\mathbf{F} \in \mathbb{R}^{|U| \times d}$, to convert each of them into low-dimensional dense vectors, $\mathbf{e}_v \in \mathbb{R}^d$ and $\mathbf{f}_u \in \mathbb{R}^d$, respectively:

$$\mathbf{e}_v = \mathbf{E}^\top \bar{\mathbf{e}}_v \,, \tag{1}$$

$$\mathbf{f}_u = \mathbf{F}^\top \bar{\mathbf{f}}_u \,. \tag{2}$$

The mathematical formulas are almost the same for graph representation learning on the two graphs, thus we take the long-term graph $\mathcal{G}^l$ as an example for illustration. We assume $\mathbf{h}_v^{l,(0)} = \mathbf{e}_v$ is the initial representation of item node $v \in \mathcal{V}^l$, and $\mathbf{h}_u^{l,(0)} = \mathbf{f}_u$ is the initial representation of user node $u \in \mathcal{V}^l$. Then we adopt a commonly used two-stage procedure [30] to conduct representation propagation on graphs. The first step is to aggregate representations of neighborhoods for a target node. Since there are two types of nodes in the graph, we first define the way of representation aggregation for a user node as follows:

$$\bar{\mathbf{h}}_u^{l,(k)} = \frac{\sum_{v \in N_u^I} \mathbf{h}_v^{l,(k-1)}}{|N_u^I|} \,, \tag{3}$$

where N_u^I denotes the item neighborhoods of user u, and k denote the k-th time of aggregation. By contrast, an item node is not only involved in user-item edges, but also associated with item-item edges. As a result, it has two types of neighborhoods and the corresponding aggregated representation is computed by:

$$\bar{\mathbf{h}}_v^{l,(k)} = \frac{\sum_{u \in N_v^U} \mathbf{h}_u^{l,(k-1)}}{|N_v^U|} + \frac{\sum_{v' \in N_v^I} \mathbf{h}_{v'}^{l,(k-1)}}{|N_v^I|} \,, \tag{4}$$

where N_v^U and N_v^I denote the user neighborhoods and item neighborhoods of item v, respectively. Through the above manner, the item representations are simultaneously influenced by user and item representations. Note that we also consider the situation that edges in the graphs take continuous values. As illustrated later in Section 5, we adopt an intuitive way of incorporating continuous edge weights into the aggregation process. However, currently no improvements are observed in the experiments.

The second step of representation propagation is to update the target user and item representations by the aggregated representations:

$$\mathbf{h}_u^{l,(k)} = \bar{\mathbf{h}}_u^{l,(k)} + \mathbf{h}_u^{l,(k-1)} \,, \tag{5}$$

$$\mathbf{h}_v^{l,(k)} = \bar{\mathbf{h}}_v^{l,(k)} + \mathbf{h}_v^{l,(k-1)} \,. \tag{6}$$

We assume the total number of propagation is K. Therefore the obtained long-term user representation is $\mathbf{h}_u^{l,(K)}$.

Thanks to the constructed group-aware short-term graph $\mathcal{G}^s$, we can gain the short-term user representation denoted as $\mathbf{h}_u^{s,(K)}$ in a similar manner. It is worth noting that the initial representation of user node u in the short-term graph is initialized as a zero vector: $\mathbf{h}_u^{s,(0)} = \mathbf{0}$. This makes sense because the aim of graph representation learning on this graph is to learn short-term user representations mainly based on their recently interacted items. And initialization with a zero vector could eliminate the influence of long-term user representations.

4.3 User Representation Fusion

So far we have the long- and short-term user representations available. We add a nonlinear fully-connected (nFC) layer on top of user representations to endow them with richer expressive ability. The computations are defined as below:

$$\mathbf{p}_u^l = \text{nFC}(\mathbf{h}_u^{l,(K)}) + \mathbf{h}_u^{l,(K)} \,, \tag{7}$$

$$\mathbf{p}_u^s = \text{nFC}(\mathbf{h}_u^{s,(K)}) + \mathbf{h}_u^{s,(K)} \,, \tag{8}$$

where nFC uses tanh as its nonlinear activation function. The idea of short-cut [7] is also contained in the above equations.

As introduced, the long-term user representations correspond to long-term preference, while the short-term user representations reveal the dynamic and recent preference. The two types of representations are complementary to each other and the fusion of them could have a stronger expressive ability. To ensure an adaptive fusion, we adopt a simple gating mechanism to calculate the relative importance of each representation type for a specific user. Mathematically, it is defined as:

$$\mathbf{p}_u = a_u \mathbf{p}_u^l + (1 - a_u)\mathbf{p}_u^s \,, \tag{9}$$

where a_u is a scalar value denoting the ratio that the long-term representation occupies in forming the integrated user representation $\mathbf{p}_u$, which is computed by:

$$a_u = \sigma(\mathbf{w}^{g\top}[\mathbf{p}_u^l; \mathbf{p}_u^s]) \,, \tag{10}$$

where σ is a Sigmoid function, $\mathbf{w}^g \in \mathbb{R}^{2d}$ is trainable parameter vector, and $[\cdot;\cdot]$ denotes concatenate operation.

4.4 Group Representation Learning

Based on integrated user representations, GLS-GRL performs group representation learning through constrained user-interaction attention, which learns the hidden correlations between group members, and group representation aggregation, deriving group-level representations.

4.4.1 Constrained user-interacted attention. The recently proposed sub-attention network [20] achieves the state-of-the-art performance in the static group recommendation problem. This network utilizes the self-attention technique [21] to represent one user with the weighted combination of other group members' representations. The weights for combination are obtained by attention computation. However, one issue of the sub-attention network is that the attention computation for each user is conducted over all the other group members. This is time-consuming and may not be so necessary since in real situations, not all users in the same group have direct interactions. With this intuition, we provide a simple modification over the self-attention technique by constraining the group members to be attended for each target user, which is named constrained user-interacted attention mechanism.

Specifically, we first use $co\text{-}interact(u, u')$ to denote the number of items that are interacted with both user u and u' in the whole history. Then we employ N_u^U to denote the neighborhoods of user u in the group, which satisfies that: $\forall u' \in N_u^U, co\text{-}interact(u, u') \geq 1$. Based on this, we define the computational details of the constrained user-interacted attention as follows:

$$\beta_{u,u'} = \frac{\exp(\mathbf{p}_u^\top \mathbf{W}^a \mathbf{p}_{u'})}{\sum_{u' \in N_u^U} \exp(\mathbf{p}_u^\top \mathbf{W}^a \mathbf{p}_{u'})}, \tag{11}$$

where a bilinear method is applied to measure the similarities between user u and u', and $\mathbf{W}^a$ is the trainable matrix of the method. Given the attention weights, the updated user representation is calculated by:

$$\mathbf{q}_u = \sum_{u' \in N_u^U} \beta_{u,u'} \mathbf{p}_{u'}. \tag{12}$$

4.4.2 Group representation aggregation. Finally, we aggregate the updated user representations to obtain the group-level representation for the group. As the study [20] demonstrates, an additional attention computation over user representations could not yield improvements in performance. Thus we adopt the simple mean-pooling technique defined as follows:

$$\mathbf{g} = \frac{\sum_{u \in M} \mathbf{q}_u}{|M|}. \tag{13}$$

So far, GLS-GRL takes the group g, its long- and short-term graphs $\mathcal{G}_g^l$ and $\mathcal{G}_g^s$ as input, and can output the graph representation $\mathbf{g}$, which is later used for computing group preference scores for candidate items.

4.5 Model Prediction and Training

4.5.1 Model Prediction. In this part, we clarify the way of calculating the group-level preference score s to the candidate item v, so as to show the whole picture of the function f (mentioned in Problem 1) in our approach. We argue the preference score to a candidate item v consists of two parts: (1) the relevance between the group and the item, measured by the cosine similarity; (2) the popularity of the item, independent of specific groups. Then the score s is computed by:

$$s = \frac{\mathbf{g}^\top \mathbf{e}_v}{||\mathbf{g}|| \cdot ||\mathbf{e}_v||} + \text{pop}_v. \tag{14}$$

where pop_v could be regarded as the popularity bias of item v, which is a trainable parameter of our model. We initialize the value to be 0 for training.

4.5.2 Model Training. For group recommendation, we present a hybrid loss function, consisting of an explicit ranking loss and an implicit ranking loss. We utilize $Y^E \subseteq V$ to denote the item set in which all the items have been interacted with group members, and $Y^I \subseteq V$ to denote the negative item set where each item has not been interacted with any user in the group. Based on this, we formally define the explicit ranking loss as follows:

$$\mathcal{L}_E = \sum_{v \in Y^E} \sum_{\hat{v} \in Y^E} \mathbb{I}(\text{num\_m}_v > \text{num\_m}_{\hat{v}}) \max(0, s_{\hat{v}} + \gamma - s_v), \tag{15}$$

where $\mathbb{I}(*)$ is an indicator function which takes value of 1 if a condition $*$ is satisfied. num_m_v represents the number of group members which have interacted with item v. γ is the margin and set to 0.1 in the experiments. We further define the implicit ranking loss, commonly adopted by previous group recommendation methods:

$$\mathcal{L}_I = \sum_{v \in Y^E} \sum_{\hat{v} \in Y^I} \max(0, s_{\hat{v}} + \gamma - s_v). \tag{16}$$

In the end, we combine the two loss functions by a linear interpolation to obtain the hybrid loss function:

$$\mathcal{L} = \alpha \mathcal{L}_E + (1 - \alpha) \mathcal{L}_I, \tag{17}$$

where α is a hyper-parameter to control the relative importance of each loss function.

5 EXPERIMENTS

In this section, we conduct experiments to answer the following research questions:

- **Q1:** How does GLS-GRL perform compared with existing sequential recommendation approaches, group recommendation methods, and some combined approaches for sequential group recommendation?
- **Q2:** Do the key components of GLS-GRL contribute to the recommendation performance? Are the core designs of GLS-GRL rational and effective compared with alternatives?

To achieve this, we first clarify the experimental setup and then analyze the experimental results in depth.

5.1 Experimental Setup

5.1.1 Dataset. We adopt two real-world datasets named WeChat and MovieLens. The WeChat dataset is collected from *Top Stories* (看一看) of WeChat, wherein users can browse articles posted by official accounts of WeChat. We select about twenty thousand groups with at least 3 members from WeChat, and collect all group members' click records for a duration of one week. We take one day as a time frame for splitting the dataset. The MovieLens dataset is obtained from the MovieLens 20M Data[4]. It contains about 20

[4]https://grouplens.org/datasets/movielens/

Table 1: Basic statistics of the datasets.

Data	WeChat	MovieLens
#Users	61,170	10,551
#Groups	21,908	58,021
#Items	73,297	2,549
Avg. group size	11.32	4.86
Time duration	2019/10/18~24	1999/03/27~2000/05/20
#Training data	39,663	56,470
#Validation data	3,584	2,664
#Test data	7,170	5,328

million ratings generated by 138,000 users for 27,000 movies. In this paper, we assume that a user has an interaction with a movie if the user gives it a rating larger than 3. To make the sparse dataset suitable for sequential recommendation, we collect user rating records in 60 weeks and take 10 weeks as a time frame. Since there is no explicit group structure in the dataset, we follow the strategy used in [2] to generate groups with similar users. This is realized by firstly computing the user-user similarity matrix by Pearson correlation coefficients and then aggregating similar users into groups, requiring that the similarity values of all user pairs in each group are larger than 0.27. In total, we generate about 60,000 groups with at least 3 members.

For each time frame, we regard the ground-truth of a target group as items that have been interacted with at least two members in the group. We use users' interaction records in the whole history as their long-term behaviors and in the current time frame as short-term behaviors, and ensure that long-term behaviors cover at least four time frames. We take all the data except the last time frame as training data. 1/3 data from the last time frame is taken as as validation data to determine optimal hyper-parameter setting and early stopping, and the left 2/3 data of the last time frame as test data to evaluate performance of different approaches. The item-item co-occurrence relationships used to construct graphs are only based on training data. In summary, the basic statistics of both datasets are summarized in Table 1.

5.1.2 Baselines. We adopt different kinds of recommendation methods in the experiments. To ensure a fair comparison, we make all the trainable baselines to adopt the same score generation manner (Eq. 14) and hybrid loss function (Eq. 17) as ours, which is an extension of the general pairwise-ranking loss utilized by them. The effectiveness of the hybrid loss is demonstrated later.

- **POP** [4]: It is the simplest one which always recommends popular items in training data to groups, regardless of other information.
- **MF+AVG** [20]: Matrix factorization learns latent factors for user and item identifiers. In our implementation, since there is no available group identifier, we use the average embeddings of group members in groups as its latent factor.
- **AGREE** [3]: It aggregates the preferences of group members based on attention computation over group members and group preference embedding associated with group identifiers. In our implementation, we only consider the modeling of group-item interactions.
- **MoSAN** [20]: This is the state-of-the-art group recommendation model which first obtains user preference embedding with respect to all other members by self-attention mechanism, and then summarizes all group members' preference as group preference.
- **GRU4Rec+AVG**: GRU4Rec [8] is an RNN-based deep sequential model for sequential recommendation. To adapt it to group recommendation, we apply mean pooling for aggregating each group member representations obtained by GRU networks as group representations.
- **STAMP+AVG**: STAMP [12] learns users' preferences from their last behaviors and long-term behaviors to obtain user representation. Similarly, we also apply mean pooling for aggregating each group member representation obtained by STAMP as group representation.
- **Combined models for SGR (GRU4Rec+AGREE, STAMP+AGREE, GRU4Rec+MoSAN, STAMP+MoSAN).** These models combine a sequential recommendation model and a group recommendation model with end-to-end learning. Specifically, they feed user representations obtained from sequential models (e.g. GRU4Rec or STAMP) into group recommendation models (e.g. AGREE or MoSAN) to get the group representations.
- **SR-GNN**: SR-GNN [16] is a graph-based session-based recommendation model which leverages graphs to encode item relations and obtain the session representation through an attention network. In our implementation, user embeddings and the obtained session representations are taken as user representations. We attempted different methods to aggregate group members' representations to group representation and report the best results.
- **NGCF**: NGCF [24] learns user and item representations via GNN on a user-item bipartite-graph. Same as SR-GNN, we attempted different methods to aggregate group member's representations and report the best results.

5.1.3 Metrics. We adopt four widely used metrics as reference and provide the computational details for a single example. The first two ranking-based metrics: Mean Average Precision (MAP) and Normalised Discounted Cumulative Gain (NDCG). Among them, MAP is defined as:

$$MAP@N = \frac{\sum_{n=1}^{N} P@n \times hit(n)}{\text{number of relevant items @N}} \times 100\%, \tag{18}$$

where $hit(n)$ denotes whether the item ranked at the position of n is true, $P@n$ is the top-n precision, and N is the number of item recommendations be evaluated. NDCG is computed by:

$$DCG@N = \sum_{n=1}^{N} \frac{\text{num\_m}_n}{\log_2(n+1)}, \tag{19}$$

$$NDCG@N = \frac{DCG@N}{IDCG@N} \times 100\%, \tag{20}$$

where num_m_n is the number of group members interacted with the n-th recommended item, and IDCG@N is the ideal DCG@N.

The last two are classification metrics: Recall and Precision. They are defined as:

$$R@N = \frac{\text{number of hit items}}{\text{number of relevant items}} \times 100\%, \tag{21}$$

Table 2: Overall performance on the two datasets. Results are denoted by percentages (% is omitted).

Methods	WeChat				MovieLens			
	MAP@10	NDCG@10	R@3	P@3	MAP@10	NDCG@10	R@3	P@3
POP	5.6028	8.1389	6.2788	7.1827	1.0029	2.4055	0.3962	0.4942
MF+AVG	6.5893	10.5072	6.8205	8.4226	1.9722	3.2306	2.2316	1.0928
GRU4Rec+AVG	7.4696	11.7308	7.9073	8.7355	2.6348	4.7653	3.9497	2.1178
STAMP+AVG	6.6999	10.8623	7.2624	8.3806	2.8253	4.2415	3.1286	1.7121
AGREE	8.3988	13.2372	8.7693	8.9850	1.9756	3.2786	2.2396	1.1949
MoSAN	8.4789	13.2325	9.0813	9.2004	2.1503	3.3146	2.3113	1.2012
GRU4Rec+AGREE	9.0862	14.0851	9.4360	9.9463	3.5300	5.3445	4.8794	2.3236
STAMP+AGREE	8.6498	13.3924	9.1022	9.3429	3.4271	5.0587	3.9037	1.9182
GRU4Rec+MoSAN	9.1412	14.1740	9.5957	10.0192	3.2251	5.2410	4.1547	2.1083
STAMP+MoSAN	8.5313	13.3801	9.1452	9.2376	2.9560	4.5534	3.4722	1.7101
SR-GNN	9.4878	14.6005	9.6917	10.8910	3.4921	5.4665	3.7255	2.2856
NGCF	9.1369	14.0697	9.6037	9.8683	4.2490	5.5359	5.0417	2.4066
Ours (GLS-GRL)	**10.1615**	**15.6782**	**10.4713**	**11.1344**	**4.6078**	**5.9109**	**5.1406**	**2.4668**

$$P@N = \frac{\text{number of hit items}}{N} \times 100\% . \tag{22}$$

The final results are obtained by averaging over the test sets. We use MAP@10 and NDCG@10, and R@3 and P@3 for evaluation, focusing more on the performance of top-ranked answers.

5.1.4 Implementation Details. We implement our model based on Tensorflow and adopt Adam with default parameter setting except learning rate of 1e-4 and mini-batch size of 64 to optimize the model. The dimension of latent vectors is set to 64. The depth of GNN is set to 3 for both datasets, and the hyper-parameter α of loss function is set to 0.4 for WeChat dataset and 0.8 for MovieLens dataset. We use early stopping strategy to terminate the learning process when the best performance on the validation data keeps unchanged for more than 20 batches. We randomly sample 20 items as negative samples for each training instance.

5.2 Experimental Results

*5.2.1 Model Comparison (**Q1**).* Table 2 shows the overall results of all the adopted models, from which we can observe:

- POP performs worst among all the methods on the two datasets. This indicates that personalized preference of group members is still crucial for the problem. MF+AVG outperforms POP significantly, but is inferior compared to other baselines. This might be due to the fact that the model capacity and nonlinear modeling ability of MF is relatively limited.
- GRU4Rec+AVG and STAMP+AVG achieve better results than MF+AVG, indicating even with simple aggregation strategies, stronger models could boost performance. One interesting phenomenon is that the two sequential models performs better than the two group recommendation models i.e. AGREE and MoSAN on MovieLens but performs worse on WeChat. It reveals that group behaviors and sequential behaviors show different importance for recommendation depending on specific datasets.
- The combined models (e.g., *+AGREE and *+MoSAN) present significantly better results than the models of *+AVG. The reason is that more advanced and effective group aggregation techniques indeed bring benefits towards the SGR problem. Besides, MoSAN behaves better than AGREE as a single model, but does not show improvements when combined with sequential recommendation models. This shows self-attention mechanism used in MoSAN is effective, but carefully-designed sequential models are needed for better performance.
- The two graph-based recommendation models behave well, showing graph neural networks are powerful tools to learn from item-item relations. In summary, the proposed GLS-GRL achieves superior performance, by achieving 7.4% and 8.1% relative improvement over the second best results w.r.t. NDCG@10 on WeChat and MovieLens, respectively.

Table 3: Results of different depth of GNN on WeChat.

Depth	MAP@10	NDCG@10	R@3	P@3
1	9.8673	15.1601	10.2574	10.8024
2	10.0198	15.4829	10.4414	10.9363
3	10.1615	15.6782	10.4713	11.1344
4	10.1160	15.5681	10.4205	11.0046

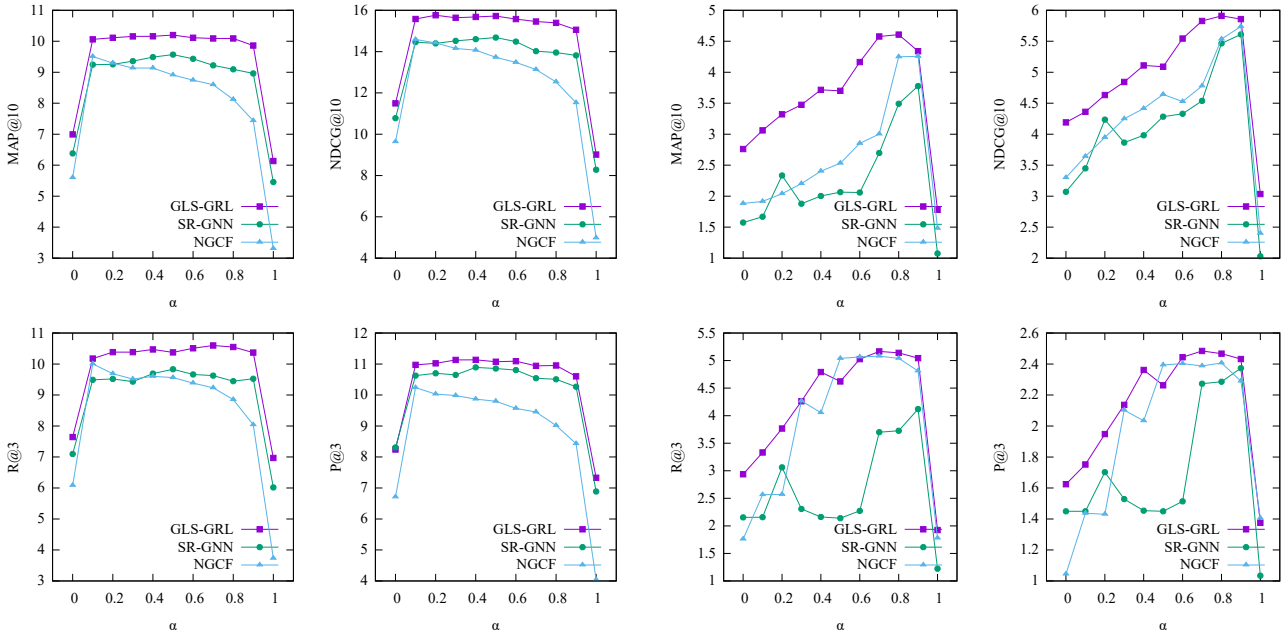
Table 4: Results of different depth of GNN on MovieLens.

Depth	MAP@10	NDCG@10	R@3	P@3
1	4.1453	5.6110	4.8767	2.3361
2	4.2042	5.7114	4.9820	2.4199
3	4.6078	5.9109	5.1406	2.4668
4	4.0165	5.4272	4.4315	2.1559

*5.2.2 Ablation Study (**Q2**).* To provide an in-depth analysis of GLS-GRL, we consider two groups of variants of the model, where "w/o" denotes removing corresponding components from GLS-GRL and "w/" means replacing some components of GLS-GRL with alternative ways. Particularly, for the first group,"w/o Graph-Aware LTG" and "w/o Graph-Aware STG" denote removing graph-aware long- and short-term graphs, respectively. "w/o User-Level Gate" discards gating mechanism shown in Eq. 9 and uses mean-pooling instead. "w/o Constrained UIA" removes the computations shown in Eq. 11 and 12. "w/o Popularity Bias" deletes the item popularity score in Eq. 14. For the second group, "w/ UIA" denotes using the same

Table 5: Ablation study of GLS-GRL on the two dataset.

Methods	WeChat				MovieLens			
	MAP@10	NDCG@10	R@3	P@3	MAP@10	NDCG@10	R@3	P@3
w/o Graph-Aware LTG	9.9318	15.2810	10.2224	10.9129	3.2101	5.4945	3.5918	2.2147
w/o Graph-Aware STG	9.3674	14.6154	9.6656	10.0925	4.1406	5.6032	4.5617	2.2485
w/o User-Level Gate	10.0106	15.5360	10.3786	11.0744	4.1152	5.6362	4.8677	2.3699
w/o Constrained UIA	9.9091	15.2985	10.1661	10.8415	4.1945	5.5955	5.0321	2.4162
w/o Popularity Bias	9.7962	15.1317	10.1230	10.7638	2.7077	4.4861	5.0260	2.3878
w/ UIA	9.9082	15.2979	10.2513	10.9233	4.2451	5.6610	4.7833	2.2792
w/ Global Graph	9.3480	14.3026	9.8214	10.2464	4.3888	5.5364	4.8531	2.3774
w/ Continuous Edge Weights	10.1410	15.6523	10.3706	11.1134	4.2308	5.6255	5.0153	2.4418
w/ User-Init Short	9.8084	15.1279	10.3509	10.7661	4.3467	5.7422	5.0020	2.4124
Ours (GLS-GRL)	**10.1615**	**15.6782**	**10.4713**	**11.1344**	**4.6078**	**5.9109**	**5.1406**	**2.4668**



(a) WeChat (b) MovieLens

Figure 4: Effect of loss function's hyper-parameter α with different weights.

self-attention computation of MoSAN by deleting the constraint of users. "w/ Global Graph" replaces the two group-aware graphs with two global graphs containing all users. "w/ Continuous Edge Weights" utilizes continuous weights for weighted aggregation of user and item representations. At last, "w/ User-Init Short" represents using user embeddings to initialize user representations in the short-term graph.

Table 5 shows the performance of different variants. By first investigating the results in the first part of the table, we have the following observations. (1) Either removing long-term graph or short-term graph degrades the recommendation performance, showing they are complementary for each other. (2) Using gating mechanism for fusion leads to better user representations and group-level representations. (3) Constrained user-interacted attention is indispensable for improving performance as well. (4) Popularity bias of items makes positive contributes to the performance, especially for MovieLens. Note that as aforementioned, all the trainable baselines adopt the same score generation manner as ours to ensure fairness of comparison. We further compare the results of the variants and GLS-GRL in the second part and have the corresponding observations. (1) Using self-attention computation without constraint (same as MoSAN) suffers from performance drop. This indicates incorporating constraint for user attention computation is effective. (2) Encoding relations from global graphs into user and item representations leads to worse results. We attribute this to the lack of mechanism for leveraging group information to help learning

group members' representations. (3) Simply considering continuous edge weights by weighted aggregation could not boost the performance. (4) Initializing short-term user representations with user embeddings instead of zero vector causes inferior performance. The reason might be that the pure short-term user representations better complement long-term representations.

5.2.3 Impact of propagation number K. To evaluate the impact of the depth of GNN, we test different propagation numbers in the range of $[1, 2, 3, 4]$. Table 3 and 4 show the corresponding results on the two datasets, respectively. We can find that the performance becomes better with the depth grows from 1 to 3, which shows the advantages of modeling high-order relations between items and users through GNN. When further increasing the layer number, we observe no improvements and find the results are even worse on MoveiLens. This might be attributed to the overfitting issue of more layers.

5.2.4 Impact of α. In Figure 4, we depict how the change of α in the hybrid loss function affects the performance of GLS-GRL and two strong baselines, i.e., SG-GNN and NGCF. The left 4 subfigures correspond to the results on WeChat and the right 4 subfigures show the performance on MovieLens. Since the performance of all the three approaches declines notably when α is set 0 or 1, the rationality of combining explicit and implicit ranking loss functions is validated. Moreover, GLS-GRL outperforms SG-GNN and NGCF in most value cases of α, and its optimal results are significantly the best. This demonstrates our model performance is robust. Besides, the values of α and their variation trends are not exactly the same for the best results on the datasets, showing the relative importance of the two losses depends on the characteristics of specific datasets.

6 CONCLUSION

In this paper, we formulate the novel and important problem of sequential group recommendation. The fundamental challenge of learning dynamic group representations based on the sequential user-item interactions of group members is addressed by the proposal of GLS-GRL. The model has the major innovation of consolidating group representation learning and long- and short-term user representation learning in a unified framework. Experiments on real-world datasets verify the efficacy of the whole model and the contributions of its main components.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable comments. This work was supported in part by the National Natural Science Foundation of China (No. 61702190, No. U1609220) and in part by the foundation of Key Laboratory of Artificial Intelligence, Ministry of Education, P.R. China. Wen Wang is also supported by 2019 Tencent Rhino-Bird Elite Training Program.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.
[2] Linas Baltrunas, Tadas Makcinskas, and Francesco Ricci. 2010. Group recommendations with rank aggregation and collaborative filtering. In *RecSys*. 119–126.
[3] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang Hong. 2018. Attentive Group Recommendation. In *SIGIR*. 645–654.
[4] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *RecSys*. 39–46.
[5] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, and Dasarathi Sampath. 2010. The YouTube video recommendation system. In *RecSys*. 293–296.
[6] Jagadeesh Gorla, Neal Lathia, Stephen Robertson, and Jun Wang. 2013. Probabilistic group recommendation via information matching. In *WWW*. 495–504.
[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*. 770–778.
[8] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. *ICLR* (2016).
[9] Chong Li, Kunyang Jia, Dan Shen, C.-J. Richard Shi, and Hongxia Yang. 2019. Hierarchical Representation Learning for Bipartite Graphs. In *IJCAI*. 2873–2879.
[10] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *CIKM*. 1419–1428.
[11] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *WSDM*.
[12] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *KDD*. 1831–1839.
[13] Xingjie Liu, Yuan Tian, Mao Ye, and Wang-Chien Lee. 2012. Exploring personal impact for group recommendation. In *CIKM*. 674–683.
[14] Thuy Ngoc Nguyen and Francesco Ricci. 2018. Situation-dependent combination of long-term and session-based preferences in group recommendations: an experimental analysis. In *SAC*. 1366–1373.
[15] Mark O'Connor, Dan Cosley, Joseph A. Konstan, and John Riedl. 2001. PolyLens: A recommender system for groups of user. In *ECSCW*. 199–218.
[16] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. 2019. Rethinking the Item Order in Session-based Recommendation with Graph Neural Networks. In *CIKM*. 579–588.
[17] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW*. 811–820.
[18] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *CIKM*. 1441–1450.
[19] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *WSDM*. 565–573.
[20] Lucas Vinh Tran, Tuan-Anh Nguyen Pham, Yi Tay, Yiding Liu, Gao Cong, and Xiaoli Li. 2019. Interact and Decide: Medley of Sub-Attention Networks for Effective Group Recommendation. In *SIGIR*. 255–264.
[21] 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 *NIPS*. 5998–6008.
[22] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. In *CIKM*. 417–426.
[23] Wen Wang, Wei Zhang, Shukai Liu, Qi Liu, Bo Zhang, Leyu Lin, and Hongyuan Zha. 2020. Beyond Clicks: Modeling Multi-Relational Item Graph for Session-Based Target Behavior Prediction. In *WWW*. 3056–3062.
[24] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*. 165–174.
[25] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. In *AAAI*. 346–353.
[26] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation. In *IJCAI*. 3940–3946.
[27] YeongWook Yang, Danial Hooshyar, and Heuiseok Lim. 2019. GPS: Factorized group preference-based similarity models for sparse sequential recommendation. *Inf. Sci.* 481 (2019), 394–411.
[28] Mao Ye, Xingjie Liu, and Wang-Chien Lee. 2012. Exploring social influence for recommendation: a generative model approach. In *SIGIR*. 671–680.
[29] 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*. 566–577.
[30] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*. 974–983.
[31] Quan Yuan, Gao Cong, and Chin-Yew Lin. 2014. COM: a generative model for group recommendation. In *KDD*. 163–172.
[32] Wei Zhang, Jianyong Wang, and Wei Feng. 2013. Combining latent factor model with location features for event-based group recommendation. In *KDD*. 910–918.
[33] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential Click Prediction for Sponsored Search with Recurrent Neural Networks. In *AAAI*. 1369–1375.