JOURNAL OF SOCIAL COMPUTING
ISSN 2688-5255 02/06 pp112−124
Volume 4, Number 2, June 2023
DOI: 10.23919/JSC.2023.0013

Enhancing Next-Item Recommendation Through Adaptive User Group Modeling

Nengjun Zhu, Lingdan Sun, Jian Cao*, Xinjiang Lu, and Runtong Li

Abstract: Session-based recommender systems are increasingly applied to next-item recommendations. However, existing approaches encode the session information of each user independently and do not consider the interrelationship between users. This work is based on the intuition that dynamic groups of like-minded users exist over time. By considering the impact of latent user groups, we can learn a user's preference in a better way. To this end, we propose a recommendation model based on learning user embeddings by modeling long and short-term dynamic latent user groups. Specifically, we utilize two network units to learn users' long and short-term sessions, respectively. Meanwhile, we employ two additional units to determine the affiliation of users with specific latent groups, followed by an aggregation of these latent group representations. Finally, user preference representations are shaped comprehensively by considering all these four aspects, based on an attention mechanism. Moreover, to avoid setting the number of groups manually, we further incorporate an adaptive learning unit to assess the necessity for creating a new group and learn the representation of emerging groups automatically. Extensive experiments prove our model outperforms multiple state-of-the-art methods in terms of Recall, mean average precision (mAP), and area under curve (AUC) metrics.

Key words: session-based recommender; user group modeling; attention mechanism; adaptive learning

1 Introduction

Session-based recommender systems (SBRSs) have attracted increasing attention due to their highly practical value[2, 3]. A session in SBRSs specifies a scope of encapsulation of items, such as a set of products in a shopping cart, a set of viewed websites within a time window, and so forth. Different sessions reflect users' diverse preferences and requirements because users' interests keep changing in various periods[4]. Some SBRSs such as Refs. [5−7] distinguish the contribution of each session to depict users' current interests. Meanwhile, there are also some SBRSs[8, 9] exploiting the connections between sessions and transferring shared knowledge across similar sessions. These systems have demonstrated a decent improvement in recommendation performances compared to conventional approaches.

The stable long-term interests and dynamic short-term requirements are two key factors affecting user decisions. Some existing SBRSs assume the two factors are associated differently with long- and short-term sessions. For instance, to learn more complete representations of users, SHAN[7] adopts a hierarchical structure to fuse the pooling results of long- and short-

• Nengjun Zhu and Lingdan Sun are with the School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China. E-mail: {zhu_nj, sunld1127}@shu.edu.cn.
• Jian Cao is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: cao-jian@sjtu.edu.cn.
• Xinjiang Lu is with the Business Intelligence Lab, Baidu Research, Beijing 100085, China. E-mail: luxinjiang@baidu.com.
• Runtong Li is with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China. E-mail: lirt19@mails.tsinghua.edu.org.
• A preliminary version of this paper was published at Chinese CSCW 2022[1].
* To whom correspondence should be addressed.
Manuscript received: 2023-05-16; accepted: 2023-07-03

term sessions. Similarly, KA-MemNN[5, 10] encodes each session from two perspectives: users' intentions and preferences, followed by a more precisely weighted combination of long- and short-term session representations. Besides, AttRec[6] and CAN[11] combine conventional and sequential recommenders to model user interests by treating long- and short-term sessions differently. To further attain disentanglement of long- and short-term interests, CLSR[12] proposes a contrastive learning framework with self-supervision.

In all these approaches, user representations are summarized based on their sessions independently, causing the learned models to be built on a per-user basis. There is no explicit information sharing between the models of users. However, in real-world applications, groups of like-minded users exist in different contexts. The users in the same group usually share similar preferences and thus might behave similarly. Unfortunately, group information is often neglected in existing SBRSs.

If the data of all related items and users are treated indiscriminately, it is a global model. Instead, more emphasis can be put on some of the related items or users to make the model more targeted, and it is a local model. For example, in Refs. [13, 14], to capture users' more specific preferences, user representations are learned from currently visited items. At the same time, many local non-session-aware recommender systems (NSRSs) have been widely explored. For instance, based on truncated singular value decomposition (SVD), the work in Ref. [15] learnt a global model for a shared aspect set as well as a set of user subset-specific models. CMN[16] took users' neighbors as the values in memory network banks, and the values are further accumulated to model users' preferences. Although by considering the stable local influences NSRSs can improve recommendation performance, they still fail to capture users' dynamic preferences and evolving latent groups and thus are not effective for next-item recommendations. Figure 1 further exhibits the difference between local NSRSs and local SBRSs.

To this end, in this paper, we aim to propose a local

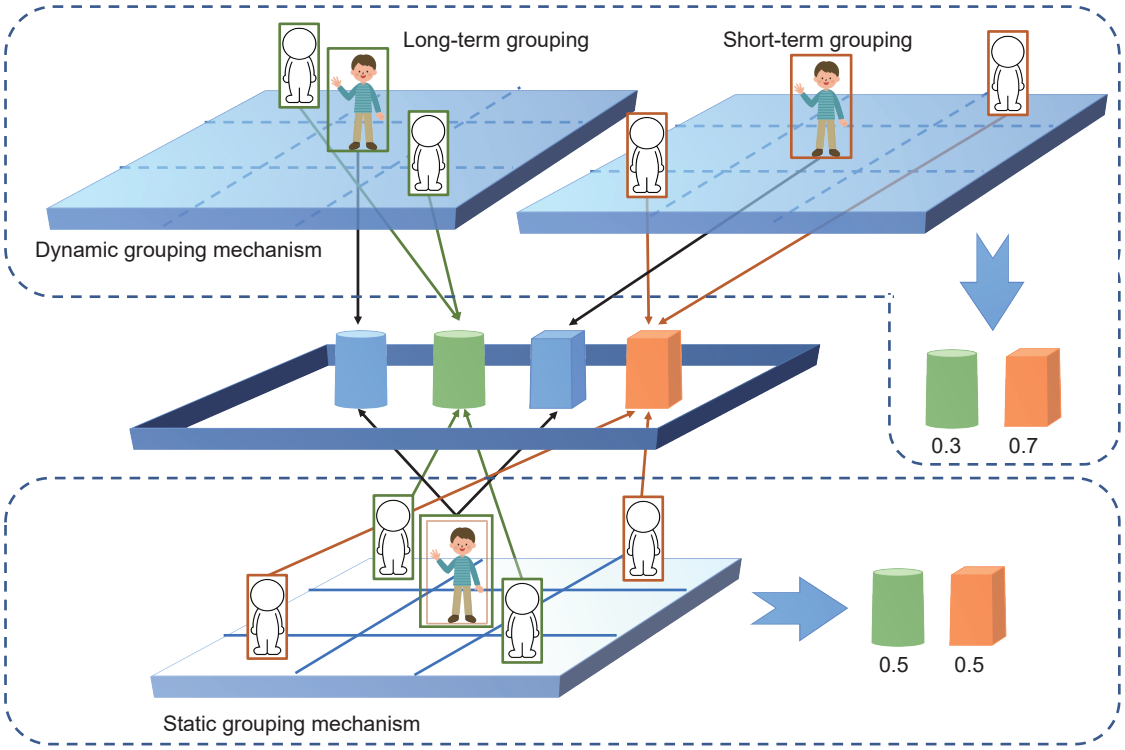


Fig. 1 An example of two different approaches of modeling influences of grouping: our dynamic grouping mechanism and conventional static grouping mechanism. The central user belongs to two latent groups represented by orange and green frames. In conventional static grouping mechanism, groups influence the user equally, and thus the visited items denoted by orange cuboid and green cylinder should be recommended with no difference. In our dynamic grouping mechanism, the two groups are treated differently since the central user has switched his group from the green one to the orange one. Up-to-date groups might have a larger impact on him, and thus items represented by orange cuboid are more in line with his taste.

SBRS. However, due to users' complicated behavior patterns, local SBRSs face several challenges. (1) Instead of being assigned a static group like conventional local approaches, in practice, a user might belong to multiple groups. For example, a user can be a cartoon fan and a tech fan simultaneously. In such a case, when he purchases a comic book, the taste of cartoon fans would have a more significant impact on him compared to that of tech fans, and vice versa. (2) The interest of each user group can be updated over time. For example, a hot cake can disturb the widespread tendency inside the group. (3) The number of groups is hard to determine. The emergence of new trends may form a new group. (4) Users can switch their groups for multiple reasons, such as the evolution of preferences and requirements. How to capture the dynamics of each user group and evolving grouping is not trivial.

To address the mentioned problems, we design LSUG, which is a next-item recommendation system based on Long- and Short-term latent User Group modeling. Specifically, we employ a hierarchical neural network to build an end-end representation learning mechanism. We first split the sessions into long- and short-term sessions, then we embed each item in sessions into a dense representation. Based on item embeddings, we abstract critical information to form long- and short-term session representations by a pooling layer. These representations reflect users' preferences and requirements. By analyzing the relations between them and latent user group embeddings, we can assign the target user to multiple user groups with different probabilities. Groups with higher probability have a greater impact on user preferences. Considering that users' interests may be updated over time, as well as their long- and short-term preferences, the probabilities of users in groups are dynamic. Then, we can summarize the influences from different user groups by a weighted combination of group embeddings. Finally, the long- and short-term session representations and the user group influences are further aggregated to more comprehensive user representations based on an attention model. These representations replace user latent vectors in a pairwise model, i.e., Bayesian personalized ranking (BPR)[17], to estimate the probability of an item being the next visited one.

We also utilize adaptive learning to assess the necessity for creating a new group. If it is necessary, then the representation of emerging groups can be learned automatically. Thus, the model can avoid setting the number of groups manually.

Our contributions are summarized as follows:

● We reveal that dynamic groups of like-minded users exist over time, and users in the same group might share a similar preference. By considering the impact of latent user groups, we can capture a user's preference in a better way.

● We propose a recommendation model for next-item recommendations, i.e., LSUG, which learns user embeddings by modeling long- and short-term latent user groups.

● We propose an adaptive learning unit to create new groups. Thus rather than being tuned by hand, the number of groups can be increased automatically according to modeling requirements.

● The extensive experiments prove the superiority of our model compared to multiple state-of-the-art approaches in terms of Recall, area under curve (AUC), and mean average precision (mAP) metrics.

2 Related Work

Traditional approaches, e.g., collaborative filtering (CF), model the relations between users and items in a static way. They neglect the sequential dependencies inside the user-item interactions. To tackle such a problem, Markov chain based (MC-based) methods, such as Ref. [18], are developed. The work in Ref. [19] incorporated hidden Markov models into matrix factorization to deal with temporal dynamics in recommender systems. However, MC-based models only consider the first-order dependency. Thus, they usually fail to capture more complex high-order user sequential patterns.

Neural networks (NNs) are widely explored and applied in recent years thanks to their ability to handle highly complex users' behaviors. Different from MC-based methods, RNN-based technologies like HRNN[20] can model higher-order sequential dependencies while avoiding the exponential growth of parameters existing in higher-order MCs[18]. However, RNN models suppose that items in a session follow a rigid order, which dose not match the real-world session based settings, as a user might buy or look through these items randomly in a short time. Attention mechanism is incorporated into the neural network to distinguish the

importance of different items within a session. CoCoRec[21] leverages category information to capture the context-aware action dependence and uses a self-attention network to capture item-to-item transition patterns within each category-specific subsequence. DPAN[22] applies an attention network to model both the collective and sequential information within sessions. There are some works[23, 24] regarding the last item as the user's main preference at the present period, so the last item is utilized to calculate the weight of other items. However, they have limited ability to explicitly learn user preference, as the selected item could be a noisy click.

The effectiveness of graph neural networks (GNNs) has been reported[25, 26] in recommendation domains. MixGCF[27] utilizes the underlying GNN-based recommenders to synthesize negative samples. GHCF[28] uses graph convolutional network (GCN) to explore the high-hop user-item interactions. MB-GMN[29] is an integrative neural architecture with a meta-knowledge learner and a meta-graph neural network to capture the personalized multi-behavior characteristics. MDSR[30] aims to improve the diversification of the recommendation by exploring the multi-intent of users. These models are usually trained in a pairwise manner, i.e., one item has priority over the other. But in reality, items in the same session can not always have such partial order relations. Thus, these approaches might lead to false dependencies. Besides, RNN- and MC-based approaches are apt to forget long-term information and are biased to recently visited items according to their structures.

Recently, many SBRSs rely on NNs to model users' long- and short-term interests. They pursue this task from two perspectives: (1) using an attention mechanism to learn explicit session-specific weights[5, 7, 31] and (2) exploiting different prototypes to model long- and short-term sessions[9, 14, 15]. Both of these two techniques have been proven very successful in SBRSs. Our work follows the first pipeline. However, these SBRSs treat each user independently and learn from users' personal behaviors to make recommendations, causing no explicit information sharing between similar users. Therefore, user and item representations are learned from a global view. In reality, there might be strong local associations inside users and items, which has been validated in many local NSRSs such as CMN[16] and r(s)GLSVD[15]. Unfortunately, NSRSs do not take temporal orders of user behaviors into account, which limits their performances. To this end, we propose local SBRSs to combine the advantages of SBRSs and local fusion settings.

3 LSUG Model

3.1 Problem formulation

In a recommender system, we have a user u and an item v in a user set $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$ and an item set $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$, respectively. Let $s = \{v_1, v_2, \ldots, v_{|s|}\} \subset \mathcal{V}$ be an item set clicked by a target user within a session, i.e., Δt. Throughout the history of user behaviors, we have a session sequence denoted by $\mathcal{S}_t^u = \{s_1^u, s_2^u, \ldots, s_t^u\}$ for each user, where t indicates the index of sessions following timestamps.

Formally, given a user u and his session sequence $\mathcal{S}_t^u$, we aim to build a model to predict the next items that have high probabilities belonging to the current session s_t, by taking the consideration of user u's long- and short-term sessions, as well as his long- and short-term latent groups' influences.

3.2 Overview

Our model (the framework based on Long- and Short-term latent User Group modeling, LSUG) shown in Fig. 2 is a hierarchical end-end framework. It splits user behaviors into long- and short-term ones. For each part, we aggregate item embeddings to form a user's preference representation. Then, based on the long- and short-term representations, we calculate the probability distribution of groups that users might belong to, followed by an aggregation of group features to capture the impact from users' neighbors as well as the differences in preference between subsets of like-minded users. Finally, we construct a hybrid user representation using users' long- and short-term session representations and group influences. Next, we give an introduction to each part of the model.

3.3 General embedding construction

We use two matrices $\boldsymbol{U} \in \mathbb{R}^{N \times K}$ and $\boldsymbol{V} \in \mathbb{R}^{M \times K}$ with fully-connected NNs to transform one-hot encoding of users and items to dense vectors, in which $N = |\mathcal{U}|$ (resp. $M = |\mathcal{V}|$) denotes the number of users (resp. items) and

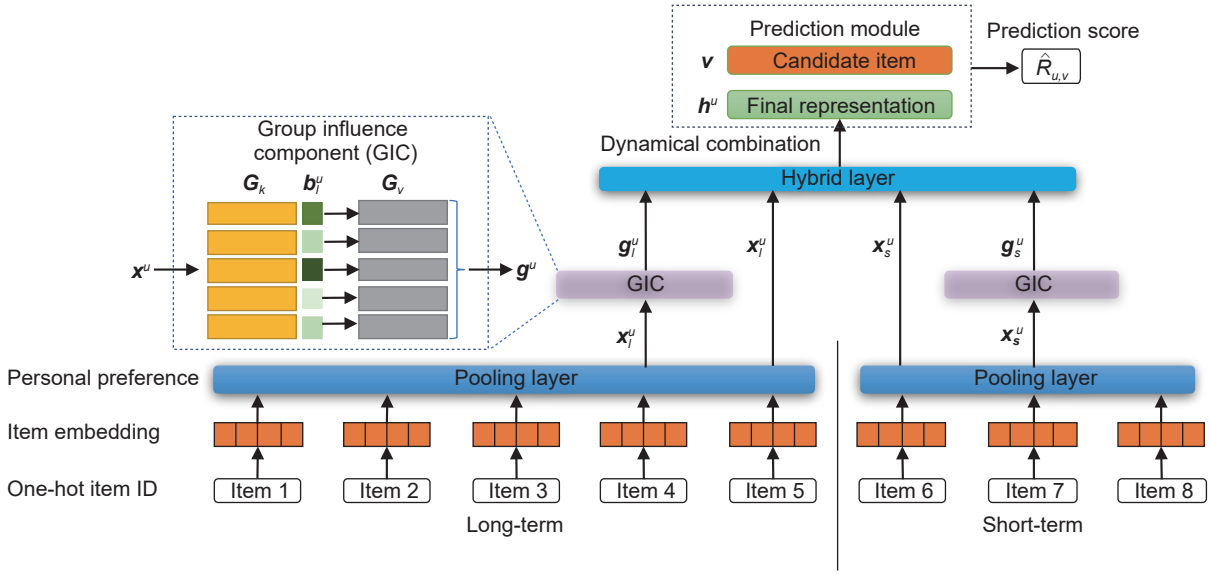


Fig. 2 Framework of LSUG.

K is the latent dimensionality. Let $\boldsymbol{u} \in \mathbb{R}^K$ and $\boldsymbol{v} \in \mathbb{R}^K$ represent an embedding vector of user u and item v, respectively. They capture static features since they do not change with time.

Inspired by key-value memory networks (KV-MemNN)[32], to determine the influence of groups to each user, we assume L user groups with L latent anchor representations denoted by $\boldsymbol{G}_v \in \mathbb{R}^{L\times K}$. $\boldsymbol{G}_v$ describes the preferences of latent user groups. At the same time, latent groups have an additional representation matrix denoted by $\boldsymbol{G}_k \in \mathbb{R}^{L\times K}$ deciding the relations with users. $\boldsymbol{G}_k$ and $\boldsymbol{G}_v$ are similar to the key and value elements in KV-MemNN, respectively. In this way, we can assign a user to multiple groups and accumulate influences of user grouping for him.

3.4 Context-aware input embedding

We split the sessions of a user into two parts and utilize his current session $s_t^u = \{v_1, v_2, \ldots, v_{|s_t^u|}|(u,t)\}$ to construct his context-aware input embeddings, which captures his **short-term** demands, and $s_{1:t-1}^u = \{v_1, v_2, \ldots, v_{|s_{1:t-1}^u|}|(u,t)\}$ as the **long-term** preference. Each session has a related embedding matrix which is computed according to the representations of the items in it.

Here, we feed these matrices to an aggregation function to learn semantic input embeddings $\boldsymbol{x}_s^u$ and $\boldsymbol{x}_l^u \in \mathbb{R}^K$ as follows:

$$\begin{aligned} \boldsymbol{x}_s^u &= \text{pooling}(\boldsymbol{E}_t^u), \\ \boldsymbol{x}_l^u &= \text{pooling}(\boldsymbol{E}_{1:t-1}^u) \end{aligned} \tag{1}$$

We explore three different pooling methods to aggregate item features, i.e., mean, max, and attention pooling functions.

- **Mean pooling:** It averages the values at each dimension of features. Each item has equal importance in contributing to the final result.
- **Max pooling:** It takes the max value for each dimension and captures item set features in an extreme way.
- **Attention pooling:** It is a weighted average pooling, and we calculate the weights according to the relations between the general representation of the target user u and the item v as follows:

$$\begin{aligned} w_{u,v} &= \frac{\exp(\boldsymbol{u}^{\mathrm{T}}\boldsymbol{v})}{\sum_{v_i \in s_t^u} \exp(\boldsymbol{u}^{\mathrm{T}}\boldsymbol{v}_i)}, \\ \boldsymbol{x}_t^u &= \sum_{v \in s_t^u} w_{u,v}\boldsymbol{v} \end{aligned} \tag{2}$$

The input embedding $\boldsymbol{x}_t^u$, where $t \in \{s, l\}$, reflects user u's status at different timestamps, such as his purchase requirements and related groups. Thus, it is reasonable to justify the relations between the groups, i.e., anchor points and user u according to this input embedding $\boldsymbol{x}_t^u$.

3.5 Latent user group influence modeling

We first calculate the similarity between the input embeddings, i.e., $\boldsymbol{x}_s^u$ and $\boldsymbol{x}_l^u$, and key embeddings of latent groups $\boldsymbol{G}_k$, to assess a user's current probability distribution $\boldsymbol{b}_s^u$ and $\boldsymbol{b}_l^u$ to determine the affiliation of users with specific latent groups as

$$\begin{aligned} \boldsymbol{b}_s^u &= \text{softmax}(\boldsymbol{G}_k \boldsymbol{x}_s^u), \\ \boldsymbol{b}_l^u &= \text{softmax}(\boldsymbol{G}_k \boldsymbol{x}_l^u) \end{aligned} \tag{3}$$

where $\boldsymbol{b}_s^u, \boldsymbol{b}_l^u \in \mathbb{R}^L$ and we utilize softmax(·) function to convert the vector $\boldsymbol{G}_k \boldsymbol{x}^u$ to a pseudo probability distribution vector. Then, we aggregate group features, i.e., $\boldsymbol{G}_v$, according to the distributions to construct aggregated group feature vectors as

$$\begin{aligned} \boldsymbol{g}_l^u &= \boldsymbol{G}_v \boldsymbol{b}_l^u, \\ \boldsymbol{g}_s^u &= \boldsymbol{G}_v \boldsymbol{b}_s^u \end{aligned} \tag{4}$$

where $\boldsymbol{g}_l^u \in \mathbb{R}^K$ and $\boldsymbol{g}_s^u \in \mathbb{R}^K$ represent the long- and short-term latent group influences to the user u, respectively.

3.6 Hybrid user representation modeling

This part yields a hybrid user representation from four aspects: two personal preference representations, i.e., users' long- and short-term context-aware input session embeddings, and two group influence representations, i.e., the impacts from users' current groups and historical groups. To combine these four components in a dynamic way, we investigate two approaches to fuse them. For simplicity, we denote $\{\boldsymbol{x}_s^u, \boldsymbol{x}_l^u, \boldsymbol{g}_s^u, \boldsymbol{g}_l^u\}$ as $\boldsymbol{F}$.

- **MLP hybrid:** We use a multi-layer perception (MLP) to map each feature vector to a scalar, followed by a softmax layer converting the scalar to a weight for each component.

$$\begin{aligned} w_{\boldsymbol{f}} &= \frac{\exp(\text{MLP}(\boldsymbol{f}))}{\sum_{\boldsymbol{f}' \in \boldsymbol{F}} \exp(\text{MLP}(\boldsymbol{f}'))}, \\ \boldsymbol{h}^u &= \sum_{\boldsymbol{f}' \in \boldsymbol{F}} w_{\boldsymbol{f}'} \boldsymbol{f}' \end{aligned} \tag{5}$$

- **Attention hybrid:** We calculate the weights according to the relations between the components and the embedding of the target user u.

$$\begin{aligned} w_{\boldsymbol{f}} &= \frac{\exp(\boldsymbol{u}^{\mathrm{T}} \boldsymbol{f})}{\sum_{\boldsymbol{f}' \in \boldsymbol{F}} \exp(\boldsymbol{u}^{\mathrm{T}} \boldsymbol{f}')}, \\ \boldsymbol{h}^u &= \sum_{\boldsymbol{f}' \in \boldsymbol{F}} w_{\boldsymbol{f}'} \boldsymbol{f}' \end{aligned} \tag{6}$$

3.7 Model learning

The total training procedure is shown in Algorithm 1. After the final user representation is learned, we compute the inner product of user representations and item embedding as their similarities or users' preferences for items:

Algorithm 1 Traning process

Input: embedding dimension K, initial number of group L_0, maximum number of group L, sessions data S, and initial learning rate η

Output: trained model with parameters Θ

do initialization;

shuffle the sessions data S

while not convergence **do**

  for batch in S **do**

    randomly select t' for each session sequence and split sessions into long- and short-term;

    do positive sampling in the last session;

    do negative sampling in unvisited items;

    compute loss according to Formula (8);

    do backpropagation and update parameters Θ;

  end

end

$$\hat{R}_{u,v} = (\boldsymbol{h}^u)^{\mathrm{T}} \boldsymbol{v} \tag{7}$$

We utilize a ranking and pairwise loss function proposed in Ref. [17] to train the model. For positive sampling, we randomly pick an item from user u's current session. And for negative sampling, we choose an item that the user u never bought or visited before. We denote the positive item and negative item as v^+ and v^-, respectively. Then, we calculate the final loss as follows:

$$\arg\min_{\Theta} \sum_{(u, S_t^u, v^+, v^-) \in \mathcal{D}} -\ln \sigma(\hat{R}_{u,v^+} - \hat{R}_{u,v^-}) \tag{8}$$

where $\mathcal{D}$ is the train set containing all samples, and $\sigma(x) = \dfrac{1}{1+\mathrm{e}^{-x}}$ is a sigmoid function.

4 Experiment

4.1 Experimental setup

Datasets. We use the Tmall dataset[33], which contains the purchase behaviors of users on the Tmall online shop, and the Gowalla dataset[34], which collects check-in behaviors of users. Following the settings in Ref. [10], we keep the last seven months of data and items that have been observed by no less than 20 users. We aggregate items purchased in one day by the same user into a session and remove the sessions that only contain a single item. We randomly pick 20% of total users for test and randomly select an item in their last session as

the target item to be predicted. Then, the statistics of datasets are shown in Table 1.

Baselines. We compare our model with the following baselines, including SBRSs, NSRSs, and local NSRSs. (1) BPR[17] is a classic NSRS that learns how to rank from users' feedback data by pairwise optimization. (2) GRU4Rec-bpr and (3) GRU4Rec-ce[35] are outstanding algorithms that use gated recurrent unit (GRU) to model the sequential data. The former uses BPR as the ranking loss function while the latter takes cross-entropy as the loss function. (4) CMN[16] is a kind of local NSRSs that takes users' neighbors as the values in the memory bank. (5) SHAN[7] is a state-of-the-art SBRS, which also utilizes a hierarchical neural network.

Metrics. We use Recall, AUC, and mAP to evaluate models. Recall measures how much the prediction covers the ground truth. AUC evaluates how highly positive examples have been ranked over negative examples. mAP evaluates the location of the real visited items in the predicted list.

Parameters settings. Without specification, we set K to 150 and L to 512 for both datasets. The initial learning rate is set to 0.03 with a 0.8 decay rate in every eight steps. We train the model until the convergence is reached. In the final models, we choose {pooling layer: mean pooling} and {hybrid layer: MLP hybrid} for Tmall dataset, while choosing {pooling layer: attention pooling} and {hybrid layer: attention hybrid} for Gowalla dataset since they perform best in our experiments.

4.2 Comparison of performance

Figure 3 shows the performance of LSUG and other baselines on both Tmall and Gowalla datasets under all metrics. From Fig. 3, we can observe that:

(1) Our LSUG outperforms all the baselines, including a latent factor CF model, i.e., BPR, a local NSRS, i.e., CMN, two sequential models, i.e., GRU-bpr and GRU-ce, and a hierarchical SBRS, i.e., SHAN, with a large margin, especially on Tmall dataset. For example, LSUG improves 16.9% compared with SHAN (15.9% vs. 13.6%) at Recall@20 and 4.16% at

Table 1 Statistics of datasets.

Dataset	Number of users	Number of items	Average session length	Number of train sessions	Number of test sessions	User-item matrix density (%)
Tmall	20 202	24 774	2.72	70 895	4040	0.039
Gowalla	15 076	12 419	2.95	128 374	3015	0.15

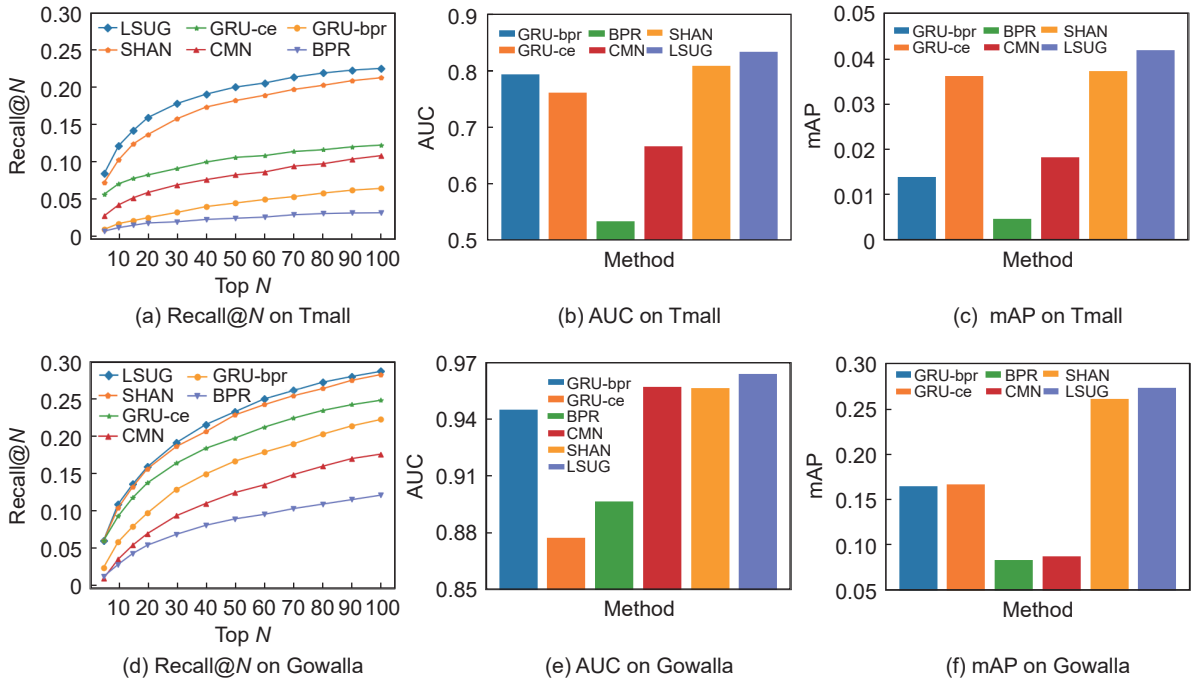


Fig. 3 Performance comparison of 6 methods on different datasets.

Recall@100 (22.5% vs. 21.6%), although SHAN outperforms other baselines. Since both LSUG and SHAN split the sessions into long- and short-term ones as well as both of them are a hierarchical model, the gain of performance might come from group influence components (GICs), which indicates the GICs are beneficial to model user preferences and help to make better recommendations.

(2) Both GRU4Rec-bpr and GRU4Rec-ce perform well, the reason is that they might successfully capture sequential patterns, i.e., the dependency relation between items. Moreover, GRU4Rec-ce is better than GRU4Rec-bpr under Recall@N and AUC metrics. The reason might be that the softmax layer computes the probability of positive items over all negative ones, while the BPR loss only uses the sampled item pairs.

(3) Although CMN and BPR are both NSRSs, CMN outperforms BPR. It may be because CMN collects user neighbors' preferences for the current user. It further proves that local information helps model user preferences.

4.3 Influence of components

4.3.1 Influence of latent user group

To further investigate the effectiveness of user groups, we remove the group features from the model and only combine $\boldsymbol{x}_s^u$ and $\boldsymbol{x}_l^u$. It leads to the results shown in Fig. 4, in which LSUG-d denotes LSUG deleting group features. We could see that, without group features, the performance of the model becomes worse, which proves that the group features are important.

4.3.2 Influence of pooling and hybrid methods.

To show the influence of aggregation methods, we exhibit the performance of all combinations of {pooling layer: mean pooling, max pooling, and attention pooling} × {hybrid layer: attention hybrid and MLP hybrid}. As shown in Table 2, for combining item features, i.e., the pooling layer in our model, mean pooling is better than max pooling under all experimental settings, e.g., mean-MLP vs. max-MLP. The reason might be that mean pooling takes all item features into consideration and passes the information to the downstream network, while max pooling only picks the most extreme features. Attention pooling sometimes obtains worse results than mean pooling. A possible explanation is that sometimes the target item is not similar to a user's general preference representation, i.e., $\boldsymbol{u}$, and thus the model pays false attention.

For hybrid methods, i.e., the hybrid layer in our model, attention and MLP get comparable results, and we could not conclude which one is better. However, most combinations achieve better results than SHAN steadily on both datasets, which shows the power of user group modeling. On the Tmall dataset, attn-MLP outperforms attn-attn by a large margin, while on the Gowalla dataset, the observation is the opposite. We note that the Gowalla dataset records users' check-in data, and a user could visit one place repeatedly, while a user purchases already bought items with much lower frequency on the Tmall dataset. Under such a circumstance, user embeddings on the Gowalla dataset might be more similar to the frequently visited items. These items occur a lot in the test set as well. As a result, on the Gowalla dataset, the attention mechanism considers general user preferences, leading to better performance. On the contrary, MLP only considers current features and thus gets worse results on the Gowalla dataset but yields a better performance on the Tmall dataset.

We randomly sample several users from both

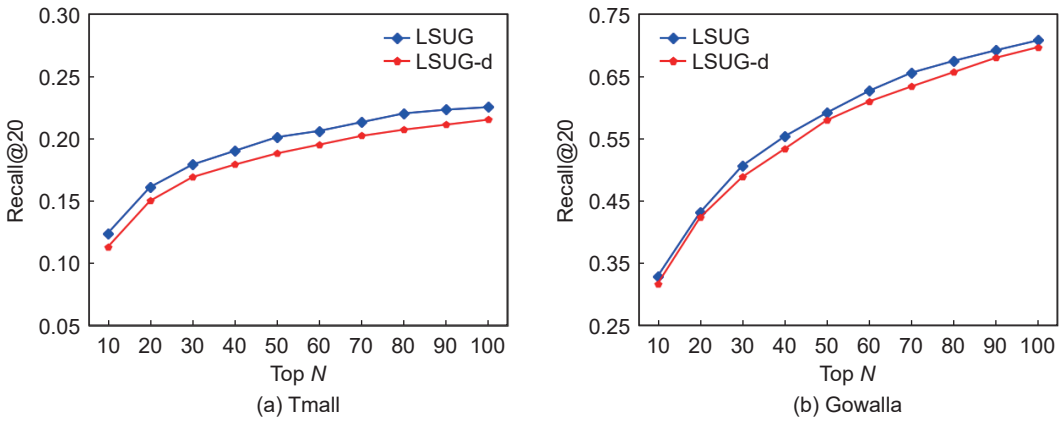


Fig. 4 Recall@20 under different sizes of the recommendation list of LSUG and LSUG-d.

Table 2 Comparison results of different pooling functions and hybrid methods.

Pooling and hybrid method	Tmall		Gowalla	
	Recall@20	mAP	Recall@20	AUC
SHAN	0.136	0.037	0.424	0.956
Max-attn	0.146	0.045	0.412	0.957
Max-MLP	0.136	0.039	0.399	0.944
Mean-attn	0.153	**0.047**	0.429	0.961
Mean-MLP	**0.159**	0.042	0.424	0.957
Attn-attn	0.141	0.041	**0.433**	**0.962**
Attn-MLP	0.155	0.042	0.400	0.954

datasets and visualize their weights of long- and short-term personal preferences (i.e., LP & SP), and long- and short-term group influences (i.e., LG & SG), as shown in Fig. 5. We can observe that the weights are customized for different users.

4.4 Influence of hyper-parameters

We study the influence of the number of groups L in our model. Specifically, the value of L changes from 100 to 1000, and we only record Recall@20 values since the other metrics have similar observations. As shown in Fig. 6, as the number of groups grows, the value of metric increases gradually at first on both datasets. This indicates that a small size is not enough to cover all potential latent user groups. However, the too larger size also decreases the performance because it might cause overfitting problems. Thus, a proper group size, e.g., $L = 500$ on the Tmall dataset and $L = 600$ on the Gowalla dataset, should be tuned to gain remarkable results.

4.5 Adaptive user group modeling

The previous results show that the number of groups L greatly impacts our model's performance. However, empirically tuning the value of L is a traditional but not wise choice. Next, we will incorporate an adaptive learning unit into LSUG to maintain the number of groups and learn the representation of emerging groups automatically. The revised version of LSUG is named adaptive LSUG (A-LSUG). We will also conduct related ablation studies and hyper-parametric analysis to explore the effectiveness of this new unit.

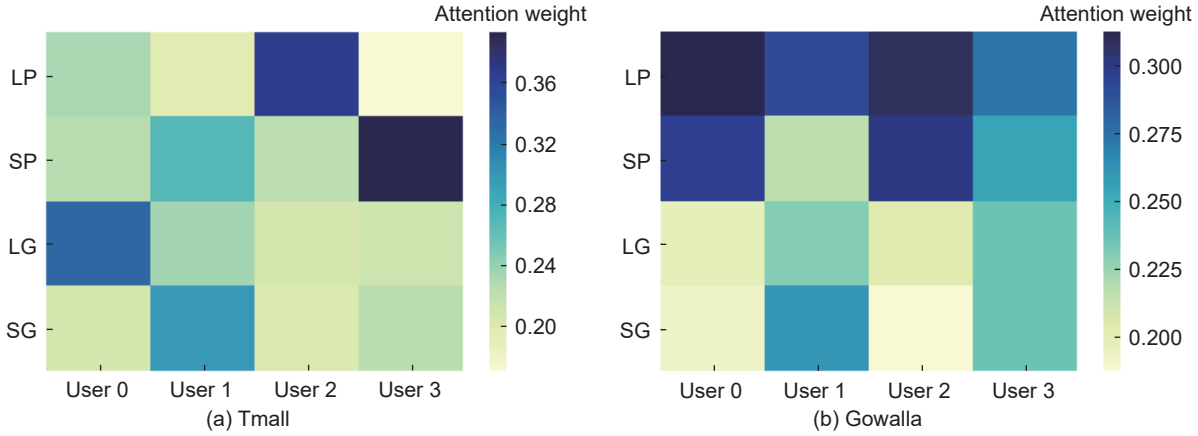


Fig. 5 Weights visualization.

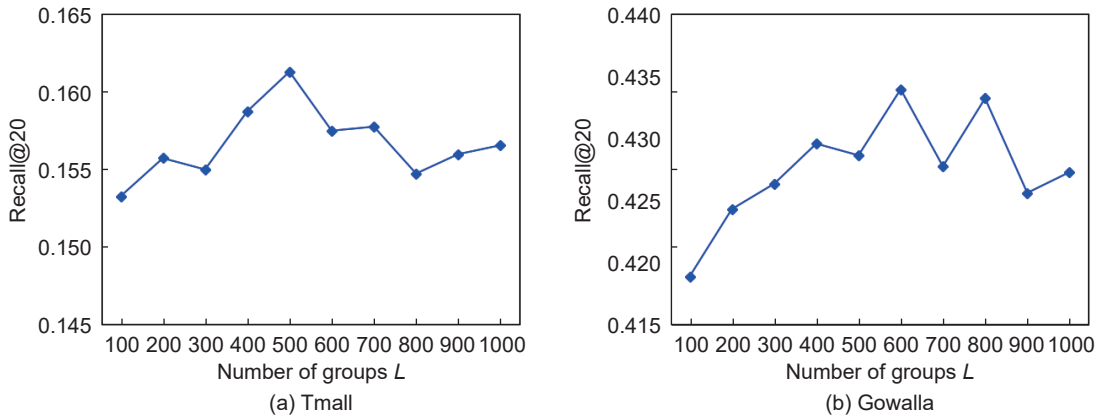


Fig. 6 Results of experiments exploring the impact of the number of groups L on LSUG.

4.5.1 Adaptive learning unit

The adaptive learning unit should address two issues: (1) how to determine the number of groups and (2) how to determine the representation of new groups. The solution is that the model first has initial user groups with a small number. Then, it assesses the necessity for creating a new group when determining the group of a target user. If the assessment result is positive, then the target user is viewed as the pivot of the new group. Also, the new group has the same representation as the target user.

Specifically, we first aggregate the correlation score between the current user's session representation and all existing latent user group features. Then, the necessity for creating a new group depends on the following possibility:

$$\begin{aligned} p_l^u &= \text{sigmoid}(\sum_k \boldsymbol{G_k} \boldsymbol{x}_l^u), \\ p_s^u &= \text{sigmoid}(\sum_k \boldsymbol{G_k} \boldsymbol{x}_s^u) \end{aligned} \tag{9}$$

where p_l^u and p_s^u are scalars, and we utilize the sigmoid$(\cdot)$ function to convert the accumulated scores to a probability distribution. Intuitively, a user's session representation should have a relatively large accumulated similarity with all group representations, i.e., the value of p^u is larger than a threshold. If not, the user may be far from the groups. In such a case, the current user should be viewed as a pivot member to form a new group. We thus introduce a threshold α to control this process, and the key and value of groups are updated as follows, respectively:

$$\boldsymbol{G}_k = \begin{cases} \boldsymbol{G}_k, & p^u \geqslant \alpha; \\ \boldsymbol{G}_k \bigoplus \boldsymbol{x}^u, & p^u < \alpha \end{cases} \tag{10}$$

$$\boldsymbol{G}_v = \begin{cases} \boldsymbol{G}_v, & p^u \geqslant \alpha; \\ \boldsymbol{G}_v \bigoplus \boldsymbol{x}^u, & p^u < \alpha \end{cases} \tag{11}$$

where $\bigoplus$ is a concatenation operator appending a vector to a matrix.

4.5.2 Effectiveness of A-LSUG

To analyze the effectiveness of the adaptive learning unit, we compare A-LSUG with LSUG and observe their performance on the Tmall and Gowalla datasets. In these experiments, the user group size in LSUG is set to 512, which is a fixed value in the training process of the model. While user group size of A-LSUG is initially set to 64, which changes adaptively during the training process. Then, the results are shown in Table 3.

Compared with LSUG, the adaptive learning unit (i.e., A-LSUG) improves the predictive accuracy under Recall@20 and mAP in both datasets, indicating the effectiveness of the adaptive learning unit. However, the results of A-LSUG are worse than LSUG under AUC. One possible reason is that the BPR loss function is optimized to maximize the ranking of positive samples, while the AUC considers the ranking of all candidate items. Therefore, the improved adaptive learning unit models user preferences in a more flexible way, which may have a positive influence on the model's performance under recall metric but may lead to a decrease in performance under AUC metric.

4.5.3 Ablation study

Like the experiment settings in LSUG, we conduct an ablation study and remove the group features, i.e., $\boldsymbol{g}_s^u$ and $\boldsymbol{g}_l^u$, from the model. Then, the model only has $\boldsymbol{x}_s^u$ and $\boldsymbol{x}_l^u$. The revised version of A-LSUG is named A-LUSG-d.

The results are shown in Fig. 7. We could see that removing group features dramatically reduces the predictive accuracy in Tmall, indicating the importance of adaptive user group modeling. The performance of A-LSUG at Recall@N is higher than that of A-LSUG-d before Top-50 in Gowalla. However, its performance improvement trend is relatively slow when the recommendation list size exceeds 50. One possible reason is that the adaptive learning unit makes the model better at recommending highly related items. Specifically, A-LSUG may have already ranked most of the items that meet users' preferences at the top of the recommendation list. A larger size of recommendation list would bring more unrelated items.

Table 3 Performance comparison between LSUG and A-LSUG.

Method	Recall@20		AUC		mAP	
	Tmall	Gowalla	Tmall	Gowalla	Tmall	Gowalla
LSUG	0.159	0.454	0.795	0.964	0.042	0.273
A-LSUG	0.184	0.485	0.766	0.937	0.082	0.279

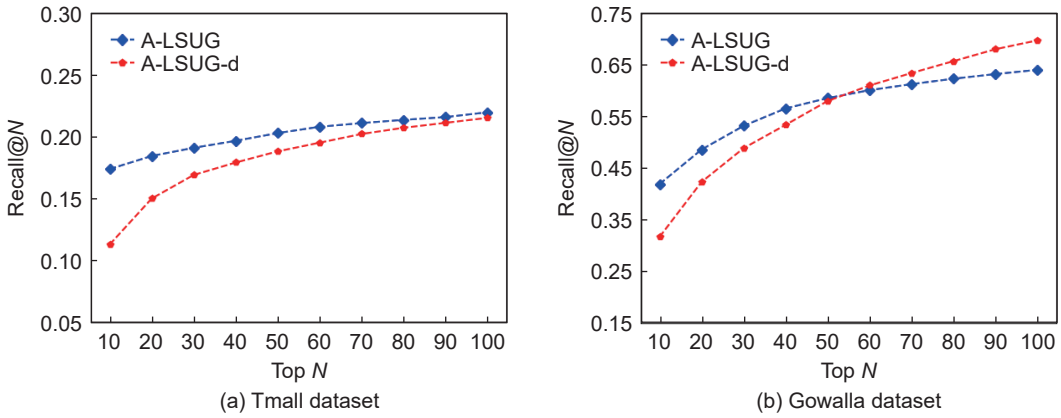


Fig. 7 Recall@*N* under different sizes of the recommendation list of A-LSUG and A-LSUG-d.

Thus, the model's performance at Recall@*N* will increase slowly when the size of the recommendation list becomes larger. Besides, the model training process uses the Top-20 recommendation list for validation. Therefore, when the recommendation list is too large, the model's performance will be limited.

4.5.4 Influence of maximum group number

To constrain the number of groups infinitely increasing, we introduce a hyper-parameter named maximum number of user groups L. When the number of user groups reaches L, the model stops the adaptive learning. This part explores the influence of L on model performance. The results are summarized in Fig. 8. We find that increasing the maximum value of user groups enhances performance. However, the settings $L \leqslant 512$ on the Tmall dataset and $L \leqslant 256$ on the Gowalla dataset impair the accuracy. One possible reason is that increasing the value of L not only encourages irrelevance among groups but also makes the interests of some user groups too fine-grained to hinder the modeling of users' interests.

5 Conclusion

In this paper, we proposed a next-item recommendation model based on learning long- and short-term user groups. Specifically, we split user behaviors into long- and short-term sessions. For all sessions, we abstract their representations according to their items. After that, the designed GICs detect users' latent long- and short-term groups and incorporate the influences from different latent groups to form the final user representations. Moreover, to avoid setting the number of groups manually, we further incorporated an adaptive learning unit to assess the necessity for creating a new group and learn the representation of emerging groups automatically. The extensive experiments on two real-world datasets demonstrate that our model outperforms several state-of-the-art models regarding multiple metrics.

There are some points for future work. For example, in our settings, the number of user groups can only increase automatically. However, the user groups can

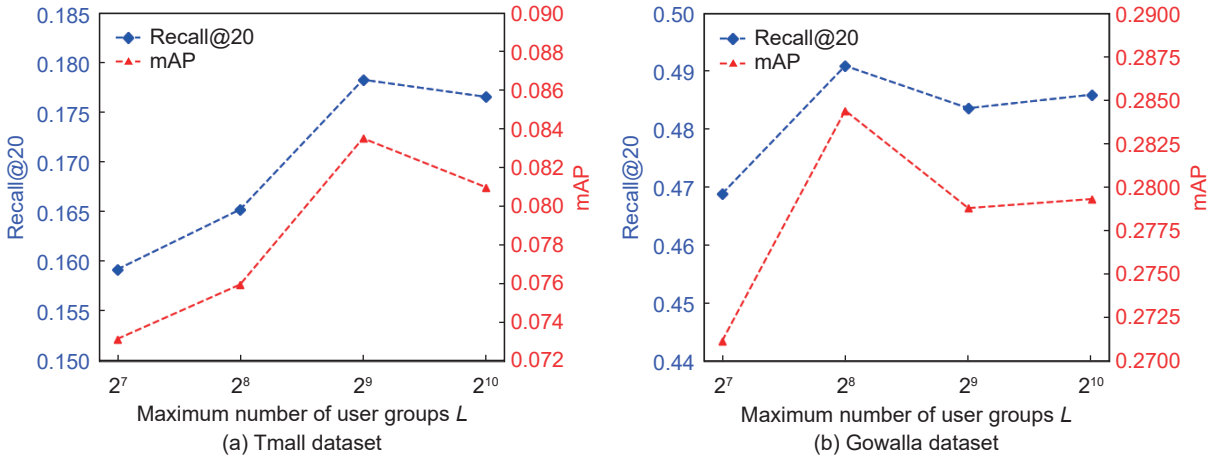


Fig. 8 Influence of maximum number of user groups *L* on A-LSUG.

also fade away with time. Thus, we will explore a mechanism to decrease the number of user groups. Besides, we can utilize contrast learning to encourage the difference between user groups explicitly, making the representation of user groups more representative.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (No. 62202282) and Shanghai Youth Science and Technology Talents Sailing Program (No. 22YF1413700).

References

[1] N. Zhu, J. Huang, J. Cao, and S. Feng, Learning User Embeddings Based on Long Short-Term User Group Modeling for Next-Item Recommendation, in *Proc. Conference on Computer Supported Cooperative Work and Social Computing* (*ChineseCSCW 2022*), Datong, China, 2022, pp. 18–32.

[2] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, and D. Lian, A survey on session-based recommender systems, *ACM Comput. Surv.*, vol. 54, no. 7, p. 154, 2022.

[3] X. Zhang, H. Lin, B. Xu, C. Li, Y. Lin, H. Liu, and F. Ma, Dynamic intent-aware iterative denoising network for session-based recommendation, *Inf. Process. Manag.*, vol. 59, no. 3, p. 102936, 2022.

[4] S. Wang, L. Hu, Y. Wang, Q. Z. Sheng, M. Orgun, and L. Cao, Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks, in *Proc. Twenty-Eighth Int. Joint Conf. Artificial Intelligence*, Macao, China, 2019, pp. 3771–3777.

[5] N. Zhu, J. Cao, Y. Liu, Y. Yang, H. Ying, and H. Xiong, Sequential modeling of hierarchical user intention and preference for next-item recommendation, in *Proc. 13th Int. Conf. Web Search and Data Mining*, Houston, TX, USA, 2020, pp. 807–815.

[6] S. Zhang, Y. Tay, L. Yao, A. Sun, and J. An, Next item recommendation with self-attentive metric learning, presented at the 33rd AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 2019.

[7] H. Ying, F. Zhuang, F. Zhang, Y. Liu, G. Xu, X. Xie, H. Xiong, and J. Wu, Sequential recommender system based on hierarchical attention networks, in *Proc. Twenty-Seventh Int. Joint Conf. Artificial Intelligence*, Stockholm, Sweden, 2018, pp. 3926–3932.

[8] J. Song, J. Xu, R. Zhou, L. Chen, J. Li, and C. Liu, CBML: A cluster-based meta-learning model for session-based recommendation, in *Proc. 30th ACM Int. Conf. Information & Knowledge Management*, Virtual Event, Australia, 2021, pp. 1713–1722.

[9] M. Choi, J. Kim, J. Lee, H. Shim, and J. Lee, Session-aware linear item-item models for session-based recommendation, in *Proc. Web Conference 2021*, Ljubljana, Slovenia, 2021, pp. 2186–2197.

[10] N. Zhu, J. Cao, X. Lu, and H. Xiong, Learning a hierarchical intent model for next-item recommendation, *ACM Trans. Inf. Syst.*, vol. 40, no. 2, pp. 1–28, 2021.

[11] S. Yakhchi, A. Behehsti, S. -M. Ghafari, I. Razzak, M. Orgun, and M. Elahi, A convolutional attention network for unifying general and sequential recommenders, *Inf. Process. Manag.*, vol. 59, no. 1, p. 102755, 2022.

[12] Y. Zheng, C. Gao, J. Chang, Y. Niu, Y. Song, D. Jin, and Y. Li, Disentangling long and short-term interests for recommendation, in *Proc. ACM Web Conference 2022*, Virtual Event, Lyon, France, 2022, pp. 2256–2267.

[13] J. Li, P. Sun, Z. Wang, W. Ma, Y. Li, M. Zhang, Z. Feng, and D. Xue, Intent-aware ranking ensemble for personalized recommendation, in *Proc. 46th Int. ACM SIGIR Conf. Research and Development in Information Retrieval* (*SIGIR*), Taipei, China, 2023, pp. 1713–1724.

[14] J. Guo, Y. Yang, X. Song, Y. Zhang, Y. Wang, J. Bai, and Y. Zhang, Learning multi-granularity consecutive user intent unit for session-based recommendation, in *Proc. Fifteenth ACM Int. Conf. Web Search and Data Mining*, Virtual Event, Tempe, AZ, USA, 2022, pp. 343–352.

[15] E. Christakopoulou and G. Karypis, Local latent space models for top-N recommendation, in *Proc. 24th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, London, UK, 2018, pp. 1235–1243.

[16] T. Ebesu, B. Shen, and Y. Fang, Collaborative memory network for recommendation systems, in *Proc. 41st Int. ACM SIGIR Conf. Research & Development in Information Retrieval*, Ann Arbor, MI, USA, 2018, pp. 515–524.

[17] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, arXiv preprint arXiv: 1205.2618, 2012.

[18] R. He and J. McAuley, Fusing similarity models with Markov chains for sparse sequential recommendation, in *Proc. 2016 IEEE 16th Int. Conf. Data Mining* (*ICDM*), Barcelona, Spain, 2016, pp. 191–200.

[19] R. Zhang and Y. Mao, Movie recommendation via Markovian factorization of matrix processes, *IEEE Access*, vol. 7, pp. 13189–13199, 2019.

[20] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, Personalizing session-based recommendations with hierarchical recurrent neural networks, in *Proc. Eleventh ACM Conf. Recommender Systems*, Como, Italy, 2017, pp. 130–137.

[21] R. Cai, J. Wu, A. San, C. Wang, and H. Wang, Category-aware collaborative sequential recommendation, in *Proc. 44th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Virtual Event, Canada, 2021, pp. 388–397.

[22] X. Zhang, H. Lin, L. Yang, B. Xu, Y. Diao, and L. Ren, Dual part-pooling attentive networks for session-based recommendation, *Neurocomputing*, vol. 440, pp. 89–100, 2021.

[23] M. Zhang, C. Guo, J. Jin, M. Pan, and J. Fang, Modeling hierarchical intents and selective current interest for session-based recommendation, in *Proc. Pacific-Asia Conf. Knowledge Discovery and Data Mining*, Delhi, India, 2021, pp. 411–422.

[24] Z. Pan, F. Cai, Y. Ling, and M. D. Rijke, An intent-guided collaborative machine for session-based recommendation, in *Proc. 43rd Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Virtual Event, China, 2020, pp. 1833–1836.

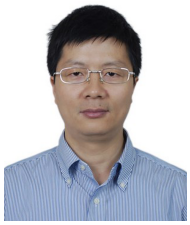
[25] S. Ge, C. Wu, F. Wu, T. Qi, and Y. Huang, Graph

enhanced representation learning for news recommendation, in *Proc. Web Conference 2020*, Taipei, China, 2020, pp. 2863–2869.
[26] J. Zheng, Q. Ma, H. Gu, and Z. Zheng, Multi-view denoising graph auto-encoders on heterogeneous information networks for cold-start recommendation, in *Proc. 27th ACM SIGKDD Conf. Knowledge Discovery & Data Mining*, Virtual Event, Singapore, 2021, pp. 2338–2348.
[27] T. Huang, Y. Dong, M. Ding, Z. Yang, W. Feng, X. Wang, and J. Tang, MixGCF: An improved training method for graph neural network-based recommender systems, in *Proc. 27th ACM SIGKDD Conf. Knowledge Discovery & Data Mining*, Virtual Event, Singapore, 2021, pp. 665–674.
[28] C. Chen, W. Ma, M. Zhang, Z. Wang, X. He, C. Wang, Y. Liu, and S. Ma, Graph heterogeneous multi-relational recommendation, in *Proc. 35th AAAI Conf. Artif. Intell.*, Vancouver, Canada, 2021, pp. 3958–3966.
[29] L. Xia, Y. Xu, C. Huang, P. Dai, and L. Bo, Graph meta network for multi-behavior recommendation, in *Proc. 44th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Virtual Event, Canada, 2021, pp. 757–766.
[30] W. Chen, P. Ren, F. Cai, F. Sun, and M. D. Rijke, Multi-interest diversification for end-to-end sequential recommendation, *ACM Trans. Inf. Syst.*, vol. 40, no. 1, pp. 1–30, 2021.
[31] J. Yuan, Z. Song, M. Sun, X. Wang, and W. X. Zhao, Dual sparse attention network for session-based recommendation, in *Proc. 35th AAAI Conf. Artif. Intell.*, Vancouver, Canada, 2021, pp. 4635–4643.
[32] A. Miller, A. Fisch, J. Dodge, A. -H. Karimi, A. Bordes, and J. Weston, Key-value memory networks for directly reading documents, in *Proc. 2016 Conf. Empirical Methods in Natural Language Processing*, Austin, TX, USA, 2016, pp. 1400–1409.
[33] L. Hu, L. Cao, S. Wang, G. Xu, J. Cao, and Z. Gu, Diversifying personalized recommendation with user-session context, in *Proc. Twenty-Sixth Int. Joint Conf. Artificial Intelligence*, Melbourne, Australia, 2017, pp. 1858–1864.
[34] E. Cho, S. A. Myers, and J. Leskovec, Friendship and mobility: User movement in location-based social networks, in *Proc. 17th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, San Diego, CA, USA, 2011, pp. 1082–1090.
[35] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, Session-based recommendations with recurrent neural networks, presented at 4th Int. Conf. Learning Representations (ICLR), San Juan, Puerto Rico, 2016.

Nengjun Zhu is currently a lecturer at the School of Computer Engineering and Science, Shanghai University. He is a master supervisor, CCF TCCC executive member, and PRIJCAI 2022 local chair. He received the bachelor degree from Sichuan University and the PhD degree from Shanghai Jiao Tong University (SJTU) in 2015 and 2021, respectively. He has been a visiting scholar at Rutgers, the State University of New Jersey, and a researcher at Baidu Research. His research interests include recommender systems, data mining, and decision support systems.

Lingdan Sun received the bachelor degree from North University of China in 2022. She is currently pursuing the master degree at the School of Computer Engineering and Science, Shanghai University, Shanghai, China. Her research interests include recommender systems and data mining.

Xinjiang Lu is currently a senior researcher at the Business Intelligence Lab, Baidu Research. He received the PhD degree from Northwestern Polytechnical University, Xi'an, China in 2018. He has served regularly on the program committees of numerous conferences, including AAAI, IJCAI, KDD, WWW, etc. His recent research interests include spatiotemporal mining, recommender systems, cross-modal text generation, and predictive analytics on a variety of urban computing applications.



Jian Cao is currently a professor at the Department of Computer Science and Engineering, Shanghai Jiao Tong University (SJTU), China and the deputy head of the department. He is the director of the SJTU & Morgan Stanley Joint Research Center on Financial Service Innovation. He is also the leader of the Lab for Collaborative Intelligent Technology. He received the PhD degree from Nanjing University of Science and Technology (China) in 2000. He was a postdoctoral researcher at Shanghai Jiao Tong University from Jan. 2000 to Dec. 2001 and then joined SJTU. He was a visiting scholar at Stanford University in Jan.−July 2004 and Sep.−Oct. 2008. His research interests include network computing, service computing, and data analytics. He leads the research group of collaborative information system. He has authored or co-authored over 180 journal and conference papers in the above areas.

Runtong Li received the master degree from Tsinghua University in 2022. His research focuses on machine learning, with a particular interest in recommender system design.