User Popularity Preference Aware Sequential Recommendation

Mingda Qian[1,2], Feifei Dai[1] (✉), Xiaoyan Gu[1], Haihui Fan[1], Dong Liu[1], and Bo Li[1]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{qianmingda,daifeifei,guxiaoyan,fanhaihui,liudong0039,libo}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. In recommender systems, users' preferences for item popularity are diverse and dynamic, which reveals the different items that users prefer. Therefore, identifying user popularity preferences are significant for personalized recommendations. Although many methods have analyzed user popularity preferences, most of them only consider particular types of popularity preferences, leading to inappropriate recommendations for users who have other popularity preferences. To comprehensively study user popularity preferences, we propose a User Popularity preference aware Sequential Recommendation (UPSR) method. By sequentially perceiving user behaviors, UPSR captures the type and the evolution of user popularity preferences. Furthermore, UPSR employs contrastive learning to gather similar users and enhance user interest encoding. Then, we can match items and user popularity preferences more accurately and make more proper recommendations. Extensive experiments validate that UPSR not only outperforms the state-of-the-art methods but also reduces popularity bias.

Keywords: Popularity preference, Popularity bias, Sequential recommendation, Neural networks

1 Introduction

In recommender systems, different users have different types of popularity preferences. Many users are sensitive to popularity. For example, some users favor top-selling products, while others prefer less popular items to show their tastes and personalities. Other users are insensitive to popularity and focus on other features, such as price and appearance. Besides, users could have different popularity preferences in different time periods. For example, users' attitudes toward famous brands may change as age grows. As the diverse and personal user popularity preferences reveal the items that users prefer, comprehensively recognizing user popularity preferences are significant for making accurate and personalized recommendations.

User popularity preferences have been widely studied, and existing methods can be separated according to the types of popularity preferences they cover. 1)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

Many methods [3, 23] emphasize whether items are popular when making recommendations, which equals to only considering the users prefer popular items. However, these methods may be biased toward hot items and recommend too many popular items to users, and such an issue is called popularity bias. Even worse, popularity bias may bring other critical issues, such as the Matthew effect [29] and echo chamber [13]. 2) To alleviate popularity bias, [48, 1] directly reduce the exposure of popular items. However, these methods ignore the users that prefer popular items. 3) Several methods consider more types of popularity preferences to simutanuously alleviate popularity bias and making accurate recommendations. [31, 45, 16] consider the users either prefer popular items or are insensitive to popularity. However, these methods ignore the users who prefer items with medium or low popularities and can not capture dynamic user popularity preferences. As a result, the users prefer less popular items or largely change their popularity preferences will be wrongly recognized and may receive inappropriate recommendations.

To solve the above issues, we aim to comprehensively capture the diverse and dynamic user popularity preferences. To achieve this, we define user popularity set as $\mathbf{Q}^u = \{\mathbf{P}_i | v_i \in S_u\}$, where item popularity $\mathbf{P}_i$ denotes the number of users who have interacted with item v_i, and S_u is the behavior sequence of user u. Then, we reveal the type and the evolution of user popularity preferences as follows: 1) To identify whether users prefer items with high, medium, or low popularity, we calculate the means μ of $\mathbf{Q}$. Then, we can separate these types of users and make more personalized recommendations. 2) To measure the change amplitude of user popularity preference, we use the standard deviations σ of $\mathbf{Q}$. Hence, we can enhance the learning of user popularity preferences. 3) To obtain the evolution of popularity preferences, we divide user popularity sets according to time periods and capture local user popularity preferences. With the evolution tendencies, we can predict users' future preferences more accurately.

To materialize our ideas, we propose a User Popularity preference aware Sequential Recommendation (UPSR) method. As shown in Figure 1, UPSR consists of three parts. 1) We apply sequential recommendation methods [42, 28, 7] as our basic model to extract general features. 2) The sequential popularity perception module enables UPSR to perceive user popularity preferences. To capture the evolution of popularity preferences, this module sequentially processes the subsequences of user behavior sequences. To obtain the types of popularity preferences, we introduce the means and the standard deviations of user popularity sets. Therefore, UPSR can give more personalized recommendations.3) The popularity contrastive learning module gathers similar users according to the captured popularity preferences and the general features. With the gathered users, this module can further enhance the encoding of user interests, and UPSR can give more proper recommendations.

In summary, we make the following contributions:

- We propose a sequential popularity perception module that comprehensively extracts the type and the evolution of user popularity preference. Therefore, items consistent with user preferences can be recommended.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

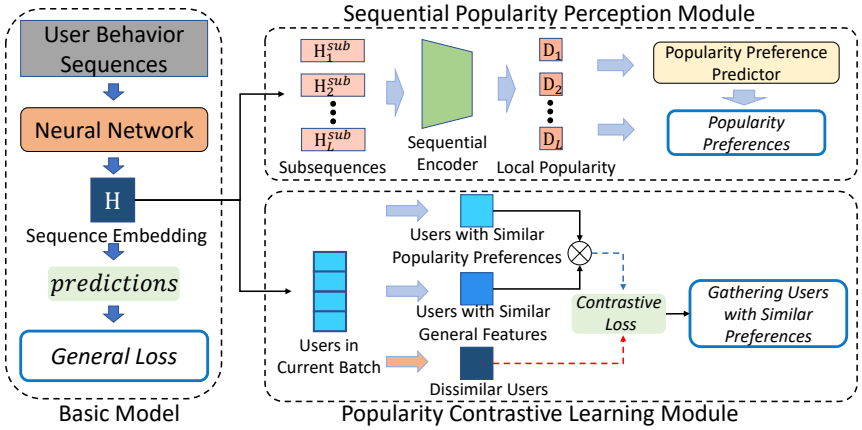

Fig. 1: The overall architecture of UPSR.

– We propose a popularity contrastive learning module that gathers similar users according to popularity preferences and general features. As a result, we can capture user interests more appropriately.
– Extensive experiments are conducted on three popular datasets for evaluation. The results show that UPSR not only outperforms the state-of-the-art methods but also reduces popularity bias.

2 Related Works

This section discusses three lines of works that are closely related to ours: sequential recommendation methods, popularity aware recommendation methods, and contrastive learning methods.

2.1 Sequential Recommendation

General recommendation methods [36, 24, 37, 10, 9] model user interests in a static way and gain users' general interests. However, users have dynamic and evolving interests, and recommender systems should rely on users' future interests rather than general interests. To solve this issue, sequential recommendation methods utilize temporal information to capture dynamic user interests and make proper recommendations. With this advantage, sequential recommendation methods receive great attention and are widely used in industry [18, 40, 34].

The key to sequential recommendation is analyzing user behavior sequences. Early works model the sequences with the Markov Chain [35]. With the development of deep learning [12, 41, 47, 44], Recurrent Neural Networks and its variants are applied [22, 21] to analyze the sequences. Recently, the self-attention mechanism [41] has been introduced to model user behavior sequences [25, 39] and

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

achieves outstanding performances. Besides, TiSASRec [28] and MEANTIME [7] argue that the timestamps of user behavior can not only generate the sequences but also enhance behavior sequence modeling. However, existing methods do not explicitly perceive user popularity preference and may recommend items not align with user popularity preferences. Therefore, we apply UPSR to these methods to capture user popularity preferences and make more accurate recommendations.

2.2 Popularity Aware Recommendation

In recommender systems, popularity has long been recognized as a significant feature. Many existing methods consider utilizing item popularity to improve the performance of recommendations. For example, MostPop is a widely used baseline method that recommends the most popular items. [3, 27] extracts temporal item popularity for recommendations. [23] re-visits and improves MostPop. Graph neural network based methods [5] represent item popularity by the degree of item nodes. However, making recommendations depending on item popularity may expose popular items too frequently, while rarely recommending less popular items. Such an issue is called popularity bias. Popularity bias may cause low-quality recommendations and even echo chamber [13]. Furthermore, popularity bias may bring the Matthew effect [29], which is unfair to small businesses on e-commerce platforms.

To alleviate popularity bias, many existing methods directly reduce the exposure of popular items. For example, [2, 48] take item popularity as prior and reduce the frequency of popular items in recommendation lists. [1] designs regularization according to item popularity. [15] includes random sampling to avoid bias. However, these methods can not perceive user popularity preferences. As a result, the popular items may not be recommended to the users that prefer popular items, and these methods often sacrifice recommendation accuracy to reduce popularity bias.

To solve this issue, several methods consider user popularity preferences when alleviating popularity bias. In these methods, [31] considers the preferences when utilizing item popularity. PDA [45] and CauSeR [16] introduce causal intervention to leverage popularity bias. These methods detect whether a user interacts with an item because of its popularity. However, such a process can hardly accurately model users with niche hobbies or occasionally flow crowds. Meanwhile, these methods ignore that user popularity preferences can be dynamic, leading to inappropriate recommendations. Therefore, we propose UPSR to comprehensively capture user popularity preferences.

2.3 Contrastive Learning

Recently, contrastive learning [14, 6, 33] shows superior ability in self-supervised learning. In recommender systems, existing works construct multiple types of training signals to perform contrastive learning. For example, [46] constructs training signal according to items, attributes, and sequences. CL4SRec [43]

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

makes data augmentation based on user behavior sequences. However, these methods ignore user popularity preferences and may recommend inappropriate items.

3 PROPOSED METHOD

As shown in Figure 1, UPSR consists of three parts. The basic model extracts sequence embeddings from user behavior sequences. The sequential popularity perception module perceives the type and the evolution of user popularity preferences. The popularity contrastive learning module gathers users with similar popularity preferences and general features. By comprehensively capturing popularity preferences, UPSR makes more personalized recommendations.

3.1 Problem Statement

In this work, the behavior sequence of user u is denoted as $S^u = \langle S_1^u, \cdots, S_{|S^u|}^u \rangle$, in which S_i^u represents the i-th item that u has interacted with. Following [25, 39, 7], we regularize user behavior sequences S^u into fixed-length n. If S^u is shorter than n, $n - |S_u|$ special tokens [PAD] are padded before S^u. If S^u is longer than n, the last n behaviors are retained. Given the regularized user behavior sequences s^u, we aim to predict which item will attract user u next.

3.2 Basic Model

To capture the dynamic user interests, we apply regular sequential recommendation methods as our basic model.

Sequential recommendation methods often contain the steps shown in Figure 1. 1) Regularized user behavior sequence s^u is taken as input. 2) These methods apply neural networks such as RNN, attention mechanism to generate sequence embeddings $\mathbf{H} \in \mathbb{R}^{n \times d}$, where d denotes embedding length and $\mathbf{H}_i$ encodes the user interest when the user interacts with the i-th item in the sequence. 3) The final recommendations are given based on $\mathbf{H}$.

UPSR only requires the basic model to generate sequence embeddings $\mathbf{H}$, and it does not matter what type of neural networks the basic model applied or whether it has other input and output. Therefore, UPSR has promised generality.

3.3 Sequential Popularity Perception Module

Given the sequence embeddings $\mathbf{H}$ from the basic model, the sequential encoder measures local popularity preferences to obtain the evolution of user popularity preferences. Meanwhile, the popularity preference predictor learns the means and the standard deviations of user popularity sets that represent the types of user popularity preferences. Finally, a denoised popularity preference loss is applied to alleviate noise and train the module.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

Sequential Encoder To capture the evolution of user popularity preference, we need to analyze multiple successive local popularity preferences. To obtain local popularity preferences, s^u are divided into $L = \lceil \frac{n}{n_{sub}} \rceil$ subsequences of fixed-length n_{sub}. The subsequence embeddings are generated by averaging the corresponding fragments of the sequence embeddings $\mathbf{H}$:

$$\mathbf{H}_i^{sub} = \frac{\sum_{j \in \phi_i} \mathbf{H}_j}{|\phi_i|} + \mathbf{O}_i, \tag{1}$$

where ϕ_i is the set of behaviors in i-th subsequence, and $\mathbf{O} \in \mathbb{R}^{L \times d}$ is the position embeddings of subsequences. Then, we encode $\mathbf{H}^{sub}$ into local popularity preference embeddings:

$$\mathbf{D}_i = \text{GELU}(\mathbf{H}_i^{sub}\mathbf{W_1} + \mathbf{b}_1)\mathbf{W_2} + \mathbf{b}_2, \tag{2}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d \times 2d}, \mathbf{b}_1 \in \mathbb{R}^{1 \times 2d}, \mathbf{W}_2 \in \mathbb{R}^{2d \times d}, \mathbf{b}_2 \in \mathbb{R}^{1 \times d}$ are parameters of fully connected layers. Following BERT [11], we utilize GELU [20] as our activation function.

As a result, the local popularity preference embeddings $\mathbf{D}$ perceive the dynamic user popularity preferences.

Popularity Preference Predictor Given $\mathbf{D}$, we predict the means and the standard deviations of user popularity sets that indicate the type of user popularity preferences. We average the local popularity preference embeddings to generate the overall popularity preference embeddings as:

$$\mathbf{E} = \sum_{i=1}^{L} \mathbf{D}_i / L. \tag{3}$$

Then, the means and the standard deviations are predicted as:

$$\mathbf{G} = \text{GELU}(\mathbf{E}\mathbf{W_3} + \mathbf{b}_3)\mathbf{W_4} + \mathbf{b}_4, \tag{4}$$

where $\mathbf{W}_3 \in \mathbb{R}^{d \times d}, \mathbf{b}_3 \in \mathbb{R}^{1 \times d}, \mathbf{W}_4 \in \mathbb{R}^{d \times 2}, \mathbf{b}_4 \in \mathbb{R}^{1 \times 2}$ are learnable parameters. The two dimensions of $\mathbf{G} \in \mathbb{R}^{1 \times 2}$ are the predicted mean and standard deviation, respectively.

Denoised Popularity Preference Loss To train the above predictor, we introduce a denoised popularity preference loss. The reason for denoising is that the data of recommender systems are often sparse, and many items have low popularity. As a result, one noisy user behavior could have a non-negligible impact on these items and further influence the reliability of the labels.

For user u, we generate a label $\hat{\mathbf{G}}^u = [\mu^u, \sigma^u]$ that consists of the mean and the standard deviation of the user popularity set $\mathbf{Q}^u$ from the training set. With

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

the labels, the denoised loss function is defined as:

$$\mathcal{L}_{popularity} = \frac{1}{2}\sum_{i=1}^{2}(\text{ReLU}(\mathbf{F}_i - \alpha_p))^2, \tag{5}$$

$$\mathbf{F} = \text{abs}(\mathbf{G} - \log(\hat{\mathbf{G}})), \tag{6}$$

where abs is the absolute value function and $\mathbf{F}$ are the prediction errors. As the values of the labels $\hat{\mathbf{G}}$ are often up to hundreds, we apply the logarithm function to smooth the labels. The ReLU [32] activation discards the gradients when the prediction error is smaller than the threshold α_p. This operation eliminates noise since noisy behavior hardly appears frequently and majorly pollutes small gradients.

With the denoised loss, our predictor can predict the means and standard deviations of user popularity sets, which further informs user embeddings about user popularity preferences.

3.4 Popularity Contrastive Learning Module

This module introduces contrastive learning [17, 6] to gather similar users and encode user interests accurately. Two users are identified as similar only if they have close popularity preferences and similar general features. As irrelevant items could have the same popularity, dissimilar users could have similar popularity preferences. Therefore, the identification should not solely depend on popularity preferences.

Following this idea, user similarities are calculated based on the overall popularity preference embeddings $\mathbf{E}$ and the overall sequence embeddings $\widetilde{\mathbf{H}} = \sum_{i=1}^{n}\mathbf{H}_i/n$. Given two users t and u, their similarity is calculated as:

$$\text{sim}(t, u) = \cos(\widetilde{\mathbf{H}}^t + \mathbf{E}^t, \widetilde{\mathbf{H}}^u + \mathbf{E}^u). \tag{7}$$

For each user, its positive samples are the users meet two conditions in the current batch, and the other users are negative samples. The first condition is that positive users should have similar popularity preference types. Given two users t and u, we calculate the distance of their labels:

$$\text{mean}(\text{abs}(\log(\hat{\mathbf{G}}^t) - \log(\hat{\mathbf{G}}^u))) < \alpha_p. \tag{8}$$

The second condition is that positive users should have similar general features. This module records the sequence embeddings in the past epochs and judges positive users based on the records. In epoch l, given the past embeddings $\mathbf{B}^{l-1} \in \mathbb{R}^d$ of user u, the condition is defined as:

$$\cos(\mathbf{B}^{l-1,t}, \mathbf{B}^{l-1,u}) < \alpha_s, \tag{9}$$

where α_s is another threshold. The past embeddings $\mathbf{B}^l$ are generated by a bootstrapping mechanism $\mathbf{B}^l = (1-\beta)\widetilde{\mathbf{H}} + \beta\mathbf{B}^{l-1}$, where β controls the pace of refreshing.

This module selects samples from current batch and does not introduce additional samples. Hence, UPSR stays light.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

Contrastive Learning Loss The contrastive learning is trained with cross-entropy loss, which is defined as:

$$\mathcal{L}_{contrastive} = -\sum_{i \in \phi_{pos}} \log \frac{\exp(\text{sim}(u, i)/\tau)}{\sum_{j \in \phi_{pos} \cup \phi_{neg}} \exp(\text{sim}(u, j)/\tau)}, \tag{10}$$

where ϕ_{pos} and ϕ_{neg} are the sets of positive and negative samples of user u, and τ is a temperature hyperparameter that scales the values of similarities.

With this loss, UPSR gathers similar users in the embedding space and adjusts user interest appropriately.

3.5 Network Training

UPSR trains the basic model with its original loss function $\mathcal{L}_{basic}$, and the final loss function of UPSR is defined as:

$$\mathcal{L} = \mathcal{L}_{basic} + \gamma_1 \mathcal{L}_{popularity} + \gamma_2 \mathcal{L}_{contrastive}, \tag{11}$$

where γ_1 and γ_2 are trade-off hyperparameters. To avoid overfitting, we apply the widely used dropout technique [38] and layer normalization technique [4] on input embeddings.

4 Experiment

4.1 Datasets

We conduct extensive experiments on three widely used datasets. 1) MovieLens [19] 1M (ML-1M)[3] is a movie rating dataset. 2) Amazon[4] is a series of datasets [30] that contain numerous reviews from Amazon.com, and the datasets are split by product categories. In this work, we adopt the Beauty and the Game category. For dataset preprocessing and splitting, we follow the common procedure in [39, 7]. For each user's behavior sequence, we hold out the last behavior for test and the second last behavior for validation. The rest behaviors are for training. In Table 1, we present the statistics of the preprocessed datasets.

4.2 Baselines

We introduce two types of methods as our baselines. 1) We select two sequential recommendation methods. **CL4SRec** [43] introduces contrastive learning, and **MEANTIME** [7] is based on the transformer architecture. Besides, MEANTIME takes the exact values of timestamps as additional input. 2) We adopt **PDA** [45] and **CauSeR** [16] that analyze user popularity preferences to alleviate popularity bias and improve recommendation accuracy.

We take CL4SRec and MEANTIME as the basic models of UPSR, PDA, and CauSeR, and we use suffixes _C and _M to indicate which basic model is applied.

[3] https://grouplens.org/datasets/movielens/1m/
[4] http://jmcauley.ucsd.edu/data/amazon/

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

Table 1: Dataset statistics. The column Avg. presents the average behaviors per user and the column Pop. presents the average popularity of items.

Dataset	Users	Items	Behaviors	Avg.	Pop.
ML-1M	6,040	3,416	1.00M	165.50	292.63
Beauty	40,226	54,542	0.35M	8.80	6.49
Games	29,341	23,464	0.28M	9.58	11.97

4.3 Implementation Details and Evaluation Metrics

For a fair comparison, the hyperparameters in baselines and UPSR are tuned through a grid search on the validation set. We tune the hyperparameters in UPSR by a grid search, and the final hyperparameters are set as follows: the feature length $d=128$, the dropout rate is 0.1, the trade-off parameters $\gamma_1=1$ and $\gamma_2=0.01$, the thresholds $\alpha_p=0.05$ and $\alpha_s=0.1$, the number of subsequences $L=4$, the temperature $\tau=1$, and the pace $\beta=0.5$. As the datasets have different densities, the sequence length n is 200 for the ML-1M and 50 for the Amazon datasets. Following the discussion in [8], we obtain the number of training epochs from the validation sets to get more convincing results. Following [26], we treat all the items that a user has not interacted with as negative items. Then, the models are asked to rank the negative items along with the positive items in the test set.

For evaluation, we apply three metrics. To evaluate the recommendation accuracy, we introduce the widely used Hit@K and NDCG@K. K indicates that the method recommends a list of K items for each user, and we select K=10. To evaluate the effect of alleviating popularity bias, we introduce the Average Recommendation Popularity (ARP). ARP $= \sum_{i=1}^{\mathrm{K}} \mathbf{P}_i/\mathrm{K}$, where $\mathbf{P}_i$ denotes the popularity of the i-th item in the recommdation list.

The ARP metric measures the item popularity of the recommendation list and represents the popularity bias that a method brings. However, a recommender system that achieves a very low ARP is not always good. For example, if a recommender system only recommends unpopular items, it will achieve low ARP but shapely decrease recommendation accuracy. Meanwhile, such a recommender system ignores the users prefer items with medium or high popularities. Therefore, UPSR aims to achieve high performance on Hit and NDCG metrics, while achieving a relatively low ARP.

4.4 Performance Comparison

The overall performance of UPSR and the baselines are presented in Table 2, 3, and 4. The experiments are repeated six times. From the results, we can observe that UPSR not only gives more accurate recommendations but also alleviate popularity bias. The results of Hit@10 and NDCG@10 metrics indicate that UPSR consistently outperforms other methods on three datasets. Meanwhile, the results of ARP show that UPSR can also alleviate popularity bias effectively when

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

Table 2: Overall performance on the ML-1M datasets.

Dataset	Method	Hit@10	NDCG@10	ARP
ML-1M	CL4SRec	0.1943±0.0022	0.1048±0.0017	666.4±12.5
	PDA_C	0.1936±0.0025	0.1041±0.0021	646.5±10.8
	CauSeR_C	0.1957±0.0026	0.1061±0.0019	642.9±13.5
	UPSR_C	0.2035±0.0027	0.1135±0.0024	642.4±9.3
	MEANTIME	0.3304±0.0028	0.1919±0.0022	610.8±11.7
	PDA_M	0.3287±0.0027	0.1909±0.0025	597.9±9.4
	CauSeR_M	0.3337±0.0028	0.1939±0.0022	602.9±8.4
	UPSR_M	**0.3465**±0.0031	**0.2002**±0.0023	**597.1**±7.9

Table 3: Overall performance on the Beauty datasets.

Dataset	Method	Hit@10	NDCG@10	ARP
Beauty	CL4SRec	0.0135±0.0003	0.0070±0.0002	76.6±8.2
	PDA_C	0.0136±0.0003	0.0070±0.0003	63.8±2.5
	CauSeR_C	0.0140±0.0005	0.0073±0.0004	63.1±3.3
	UPSR_C	0.0165±0.0004	0.0087±0.0004	63.2±3.4
	MEANTIME	0.0443±0.0003	0.0245±0.0005	57.7±5.4
	PDA_M	0.0439±0.0003	0.0243±0.0005	49.1±3.8
	CauSeR_M	0.0452±0.0006	0.025±0.0002	**48.2**±3.4
	UPSR_M	**0.0486**±0.0005	**0.0266**±0.0003	48.4±3.7

making accurate recommendations. These results represent that UPSR learns user popularity preferences more comprehensively, leading to more accurate and proper recommendations.

The results also validate that UPSR has considerable generality. On the three metrics, UPSR achieves considerable improvement compared with the basic models. Therefore, UPSR is compatible with the basic models based on different neural networks or taking additional input.

4.5 Performance on Particular Users

In this section, we validate whether our methods can comprehensively analyze the users prefer items with medium or low popularities or the users largely change their popularity preferences.

Firstly, we sort the users in ascending order of the mean μ^u of the user popularity set $\mathbf{Q}^u$, where μ^u represents the item popularity the user majorly prefers. Then, the users are averagely separated into five groups. We compare the performance on different user groups in Figure 2. The first two user groups contain users who prefer items with medium or low popularities. On these users, PDA and CauSeR achieve similar Hit and NDCG to the base model MEANTIME. Such results validate that PDA and CauSeR can hardly handle these users. On the contrary, UPSR outperforms the baselines on all five user groups, representing that UPSR can comprehensively understand the users who prefer items with different popularities.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

Table 4: Overall performance on the Game datasets.

Dataset	Method	Hit@10	NDCG@10	ARP
Game	CL4SRec	0.0509±0.0012	0.0271±0.0018	9.2±0.18
	PDA_C	0.0511±0.0023	0.0286±0.0018	8.5±0.23
	CauSeR_C	0.0537±0.0016	0.0282±0.0023	8.6±0.26
	UPSR_C	0.0570±0.0027	0.0292±0.0023	8.2±0.15
	MEANTIME	0.1271±0.0017	0.0679±0.0019	9.0±0.16
	PDA_M	0.1299±0.0015	0.685±0.0017	8.6±0.13
	CauSeR_M	0.1275±0.0018	0.686±0.0019	8.7±0.12
	UPSR_M	**0.1331**±0.0018	**0.0711**±0.0016	**8.1**±0.15

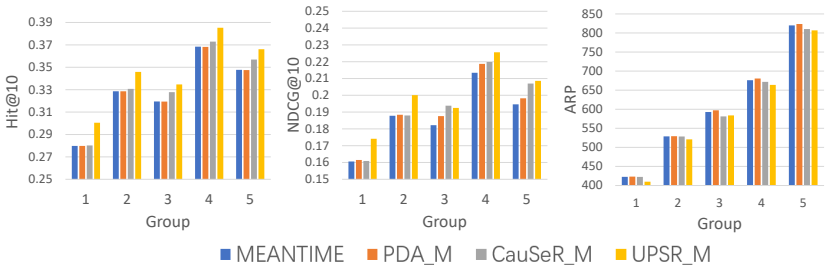

Fig. 2: The performances on users prefer item with different popularities on the ML-1M dataset.

Secondly, we sort users in ascending order of σ^u/μ^u, where σ^u is the standard deviation of $\mathbf{Q}_u$. The standard deviations represent the change amplitude of user popularity preferences, and we apply the means to normalize the results. The users are also averagely separated into five groups, and the results are presented in Figure 3. As PDA and CauSeR can hardly capture the users with dynamic popularity preferences, they achieve high Hit and NDCG on users with small amplitudes (i.e., the first two user groups). Meanwhile, the results illustrate that UPSR accurately learns user dynamic popularity preferences regardless of the amplitudes.

The ARP results in the above two experiments have a similar tendency with the other two metrics, although the differences between methods are smaller.

4.6 Ablation Study

To analyze the impact of UPSR's components, we present four variants of UPSR_M in Table 5.

Mean and **Std** are the variants that UPSR only uses the means or the standard deviations of users' popularity sets as labels. These two variants outperform the basic models, indicating that the means and the standard deviations are effective for identifying the types of user popularity preferences.

Remove_P and **Remove_C** are the variants that remove the sequential popularity perception module and the popularity contrastive learning module,

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

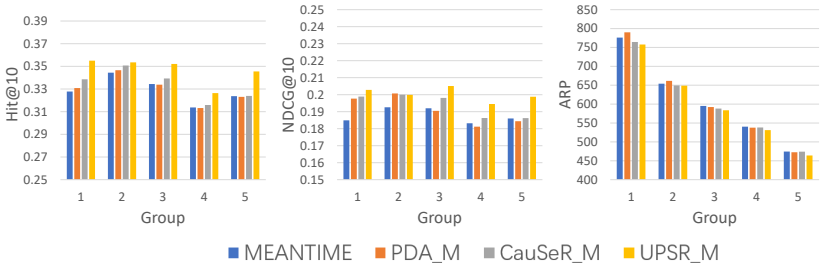

Fig. 3: The performances on users have dynamic popularity preferences on the ML-1M dataset.

Table 5: Hit@10, NDCG@10, and ARP results of the ablation study.

Dataset	ML-1M			Beauty			Game		
Metric	Hit	NDCG	ARP	Hit	NDCG	ARP	Hit	NDCG	ARP
MEANTIME	0.3304	0.1919	610.8	0.0443	0.0245	57.7	0.1271	0.0679	9.0
Mean	0.3407	0.1953	598.0	0.0482	0.0262	49.6	0.1301	0.0698	8.4
Std	0.3384	0.1934	610.4	0.0445	0.0251	56.4	0.1271	0.0707	8.8
Remove_P	0.3358	0.1945	605.5	0.0468	0.0249	51.6	0.1299	0.0696	8.5
Remove_C	0.3414	0.1989	606.4	0.0455	0.0256	51.1	0.1324	0.0705	8.9
UPSR_M	**0.3465**	**0.2002**	**597.1**	**0.0486**	**0.0266**	**48.4**	**0.1331**	**0.0711**	**8.1**

respectively. These variants outperform the basic models, denoting that the modules can give more accurate and proper recommendations independently.

5 Conclusion

To comprehensively capture user popularity preferences, we propose to analyze the means and the standard deviations of user popularity sets. Following the idea, we propose a User Popularity preference aware Sequential Recommendation (UPSR) method. On the one hand, UPSR learns the types and the evolutions of user popularity preferences. On the other hand, UPSR employs contrastive learning to gather similar users and encodes user interests more appropriately. Therefore, UPSR gives more accurate and personalized recommendations. Extensive experiment results show that UPSR not only outperforms the state-of-the-art methods but also reduces popularity bias.

6 Acknowledgments

This work is supported by program XDC02050200.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

References

1. Abdollahpouri, H., Burke, R., Mobasher, B.: Controlling popularity bias in learning-to-rank recommendation. In: Proceedings of the eleventh ACM conference on recommender systems (2017)
2. Abdollahpouri, H., Burke, R., Mobasher, B.: Managing popularity bias in recommender systems with personalized re-ranking. In: The thirty-second international flairs conference (2019)
3. Anelli, V.W., Di Noia, T., Di Sciascio, E., Ragone, A., Trotta, J.: Local popularity and time in top-n recommendation. In: Advances in Information Retrieval: 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14–18, 2019, Proceedings, Part I 41. pp. 861–868. Springer (2019)
4. Ba, L.J., Kiros, J.R., Hinton, G.E.: Layer normalization. CoRR (2016)
5. van den Berg, R., Kipf, T.N., Welling, M.: Graph convolutional matrix completion. In: KDD (2018)
6. Chen, X., He, K.: Exploring simple siamese representation learning. In: CVPR (2021)
7. Cho, S.M., Park, E., Yoo, S.: Meantime: Mixture of attention mechanisms with multi-temporal embeddings for sequential recommendation. In: RecSys (2020)
8. Dacrema, M.F., Cremonesi, P., Jannach, D.: Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In: RecSys (2019)
9. Dai, F., Gu, X., Li, B., Zhang, J., Qian, M., Wang, W.: Meta-graph based attention-aware recommendation over heterogeneous information networks. In: International Conference on Computational Science. pp. 580–594. Springer (2019)
10. Dai, F., Gu, X., Wang, Z., Qian, M., Li, B., Wang, W.: Heterogeneous side information-based iterative guidance model for recommendation. In: Proceedings of the 2021 International Conference on Multimedia Retrieval. pp. 55–63 (2021)
11. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT (2019)
12. Dong, S., Wang, P., Abbas, K.: A survey on deep learning and its applications. Computer Science Review **40**, 100379 (2021)
13. Ge, Y., Zhao, S., Zhou, H., Pei, C., Sun, F., Ou, W., Zhang, Y.: Understanding echo chambers in e-commerce recommender systems. In: SIGIR (2020)
14. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. arXiv preprint arXiv:1803.07728 (2018)
15. Gruson, A., Chandar, P., Charbuillet, C., McInerney, J., Hansen, S., Tardieu, D., Carterette, B.: Offline evaluation to make decisions about playlistrecommendation algorithms. In: WSDM (2019)
16. Gupta, P., Sharma, A., Malhotra, P., Vig, L., Shroff, G.: Causer: Causal session-based recommendations for handling popularity bias. In: CIKM (2021)
17. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR (2006)
18. Hansen, C., Hansen, C., Maystre, L., Mehrotra, R., Brost, B., Tomasi, F., Lalmas, M.: Contextual and sequential user embeddings for large-scale music recommendation. In: Fourteenth ACM Conference on Recommender Systems (2020)
19. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. ACM TIIS (2015)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

20. Hendrycks, D., Gimpel, K.: Bridging nonlinearities and stochastic regularizers with gaussian error linear units. CoRR **abs/1606.08415** (2016)
21. Hidasi, B., Karatzoglou, A.: Recurrent neural networks with top-k gains for session-based recommendations. In: CIKM (2018)
22. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: ICLR (2016)
23. Ji, Y., Sun, A., Zhang, J., Li, C.: A re-visit of the popularity baseline in recommender systems. In: SIGIR (2020)
24. Kabbur, S., Ning, X., Karypis, G.: FISM: factored item similarity models for top-n recommender systems. In: SIGKDD (2013)
25. Kang, W., McAuley, J.J.: Self-attentive sequential recommendation. In: ICDM (2018)
26. Krichene, W., Rendle, S.: On sampled metrics for item recommendation. In: SIGKDD (2020)
27. Lex, E., Kowald, D., Schedl, M.: Modeling popularity and temporal drift of music genre preferences. Transactions of the International Society for Music Information Retrieval **3**(1) (2020)
28. Li, J., Wang, Y., McAuley, J.: Time interval aware self-attention for sequential recommendation. In: WSDM (2020)
29. Liu, Y., Ge, K., Zhang, X., Lin, L.: Real-time attention based look-alike model for recommender system. In: SIGKDD (2019)
30. McAuley, J.J., Targett, C., Shi, Q., van den Hengel, A.: Image-based recommendations on styles and substitutes. In: SIGIR (2015)
31. Nagatani, K., Sato, M.: Accurate and diverse recommendation based on users' tendencies toward temporal item popularity. In: RecSys (2017)
32. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML (2010)
33. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: European conference on computer vision. pp. 69–84. Springer (2016)
34. Qian, M., Gu, X., Chu, L., Dai, F., Fan, H., Li, B.: Flexible order aware sequential recommendation. In: Proceedings of the 2022 International Conference on Multimedia Retrieval. pp. 109–117 (2022)
35. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized markov chains for next-basket recommendation. In: WWW (2010)
36. Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.): Recommender Systems Handbook. Springer (2011)
37. Sedhain, S., Menon, A.K., Sanner, S., Xie, L.: Autorec: Autoencoders meet collaborative filtering. In: WWW (2015)
38. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. JMLR (2014)
39. Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., Jiang, P.: Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In: CIKM (2019)
40. Tan, Q., Zhang, J., Yao, J., Liu, N., Zhou, J., Yang, H., Hu, X.: Sparse-interest network for sequential recommendation. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining (2021)
41. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
42. Wang, S., Hu, L., Wang, Y., Cao, L., Sheng, Q.Z., Orgun, M.A.: Sequential recommender systems: Challenges, progress and prospects. In: IJCAI (2019)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8

43. Xie, X., Sun, F., Liu, Z., Wu, S., Gao, J., Zhang, J., Ding, B., Cui, B.: Contrastive learning for sequential recommendation. In: ICDE
44. Zhang, M., Liu, X., Liu, W., Zhou, A., Ma, H., Mei, T.: Multi-granularity reasoning for social relation recognition from images. In: 2019 IEEE International Conference on Multimedia and Expo (ICME). pp. 1618–1623. IEEE (2019)
45. Zhang, Y., Feng, F., He, X., Wei, T., Song, C., Ling, G., Zhang, Y.: Causal intervention for leveraging popularity bias in recommendation. In: SIGIR (2021)
46. Zhou, K., Wang, H., Zhao, W.X., Zhu, Y., Wang, S., Zhang, F., Wang, Z., Wen, J.R.: S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 1893–1902 (2020)
47. Zhu, X., Li, J., Liu, Y., Liao, J., Wang, W.: Operation-level progressive differentiable architecture search. In: 2021 IEEE International Conference on Data Mining (ICDM). pp. 1559–1564. IEEE (2021)
48. Zhu, Z., He, Y., Zhao, X., Zhang, Y., Wang, J., Caverlee, J.: Popularity-opportunity bias in collaborative filtering. In: WSDM (2021)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-35995-8_8