Hindawi
Security and Communication Networks
Volume 2022, Article ID 4264489, 9 pages
https://doi.org/10.1155/2022/4264489

Research Article

Adaptive Alleviation for Popularity Bias in Recommender Systems with Knowledge Graph

Feng Wei,[1] Shuyu Chen ,[1] Jie Jin,[2] Shuai Zhang,[1] Hongwei Zhou,[1] and Yingbo Wu[1]

[1]*School of Big Data and Software Engineering, Chongqing University, Chongqing 400044, China*
[2]*Science and Technology Innovation Department, China Telecom, Chongqing Branch, Chongqing, China*

Correspondence should be addressed to Shuyu Chen; sychen@cqu.edu.cn

Received 15 November 2021; Accepted 16 February 2022; Published 7 April 2022

Academic Editor: Chenquan Gan

Recommender systems are known to suffer from the popularity bias problem: popular items are recommended frequently, and nonpopular ones rarely, if at all. Prior studies focused on tackling this issue by increasing the number of recommended nonpopular (long-tail) items. However, these methods ignore the users' personal popularity preferences and increase the exposure rate of the nonpopular items indiscriminately, which may hurt the user experience because different users have diverse interests in popularity. In this work, we propose a novel debias framework with knowledge graph (AWING), which adaptively alleviates popularity bias from the users' perspective. Concretely, we explore fine-grained preferences (including popularity preference) behind a user-item interaction by using the heterogeneous graph transformer over the knowledge graph embedded with popularity nodes and endow the preferences with explicit semantics. Based on this idea, we can manipulate how much popularity preference affects recommendation results and improves the exposure rate of nonpopular items while considering the popularity preferences of different users. Experiments on public datasets show that the proposed method AWING can effectively alleviate popularity bias and ensure the user experience at the same time. The case study further demonstrates the feasibility of AWING on the explainable recommendation task.

1. Introduction

In the age of the Internet, users can enjoy a variety of services on various electronic platforms. However, as the number of users continues to increase, the problem of information overload becomes more serious, which makes users cannot effectively search for the content they want. Recommendation system is an effective method to solve such problems [1]. Among these, collaborative filtering, as one of the most successful methods of recommendation system, can predict the rating of a certain user for an item and generate a recommendation list by using the preference of a certain user group [2]. So, accurately characterizing users' interests lives at the heart of an effective recommender system [3], which is challenging, however. There are even some studies aiming at hindering the system in its efforts to accurately profile users for their privacy [4]. One barrier to the effectiveness of capturing users' representation is the problem of popularity bias: collaborative filtering recommenders typically emphasize popular items much more than nonpopular ones [5], which makes popular items to be rated higher than their ideal values so that they may be recommended to some users who are actually not interested in those items. Figure 1 illustrates the long-tail phenomenon in the well-known LastFM [6]. The vertical line separates the top 20% of items by popularity, and these items cumulatively have many more ratings than the 80% long-tail items to the right. Similar distributions can be found in other systems as well. After being trained on such long-tailed data, the models inherit this bias and, in many cases, expand it by over-recommending the popular items. As a result, they will be rated by more users and this goes on again and again; the rich gets richer and the poor gets poorer.

Although popular items often get good recommendation, recommending them to users is sometimes not meaningful because these items are likely well-known. In

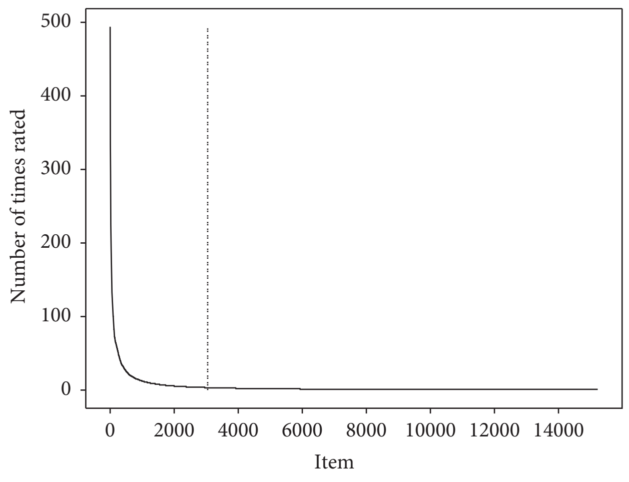

Figure 1: Item popularity in LastFM.

other words, recommending serendipitous items from the nonpopular items is ordinarily considered valuable to users [7], as these are items that users are less likely to know about. Therefore, recommender systems should explore a balance between popular and nonpopular items. Previous studies have mainly focused on increasing the number of recommended nonpopular (long-tail) items. Jones introduces a reweighting method to improve performance on small community detection [8]. Then, some studies [9, 10] introduce a regularization term to correct popularity bias. Besides, a few methods [11] utilize propensity score to decrease the ratio of popular items. However, these methods increase the exposure rate of the nonpopular items indiscriminately, largely ignoring the user's interest in popularity, which may hurt the user experience [12].

In this paper, we propose an adaptive framework to alleviate popularity bias from the users' perspective, named AWING (Adaptive Alleviation for Popularity Bias with Knowledge Graph). The key idea is that a user typically has multiple preferences (or reasons), driving him to consume different items. Based on this idea, we can capture users' popularity preferences and remove a percentage of popularity preferences (lower a ratio of weight in popularity preferences) based on their profiles with a knowledge graph for these users who are not interested in popular items. The knowledge graph is a practical approach to represent large-scale information from multiple domains [13]. To describe a knowledge graph, we can use nodes in the graph as entities and edges as relations between entities, which follows the resource description framework (RDF) standard [14]. Of course, to capture users' multiple preferences on items, the content feature information is important and helpful; some studies also utilize it to optimize the learning model [15]. However, disentangling these preferences is challenging and has not been well explored. Specifically, we face two key challenges: (1) although knowledge graph can provide rich information to learn these preferences, it lacks knowledge of popularity bias; (2) different users have diverse interests in popular items; how to adaptively eliminate popularity bias according to personal taste is another challenge. To cope with these two challenges, we design AWING with two stages: identifying fine-grained preferences behind a user-item interaction and generating recommendation results that match the user's interest in popular items. AWING mainly includes the following four components: (1) a component which constructs knowledge graph embedded with popularity nodes; (2) a component which models fine-grained user preferences; (3) a component which learns the representations of the users, items, and fine-grained preferences based on a heterogeneous graph transformer model; (4) a component which generates personalized recommendation list with removing popularity preference personally. Among these components, components (1–3) deal with challenge (1), while the component (4) deals with challenge (2).

The contributions of this work are summarized as follows:

(i) To the best of our knowledge, we are the first to introduce knowledge graph embedded with popularity nodes in heterogeneous graph to alleviate popularity bias

(ii) We propose a flexible framework AWING to alleviate popularity bias from the users' perspective, which uses fine-grained preferences to profile user-item relationships over the knowledge graph and then remove a percentage of popularity preference for different users

(iii) We conduct extensive experiments on two public datasets to demonstrate the effectiveness of the proposed model for alleviating popularity bias and the case study shows the feasibility of our model on the explainable recommendation task

2. The Proposed Model

In this section, we first introduce the notions and definitions used throughout this paper, and then we show how popularity nodes are embedded in knowledge graph and how to model fine-grained preferences of users. After that, we present how to develop the heterogeneous graph transformer [16] module of the proposed AWING on the synthetic graph. Finally, we use the trained AWING to estimate whether a user will adopt an item considering the fine-grained preferences.

2.1. Preliminary

Interaction data: given a list of user-item interactions $Y = \{(u, i)\}$, we use implicit feedback as the protocol so that each pair (u, i) implies the user $u \in U$ consumes the item $i \in I$. An additional relation interact_with is introduced to explicitly present the user-item relationship and convert a (u, i) pair to the $(u, \text{interact\_with}, i)$ triplet. As such, the user-item interactions can be seamlessly combined with KG.

Knowledge graph (KG): KG is a directed graph composed of subject – property – object triple facts. Each triplet denotes a relationship r from head entity s to tail entity t, formally defined by (s, r, t), where s and t are entities, and r is a relation. With the mappings between items and KG entities (also includes items), KG can profile items and offer complementary information to the interaction data.

Heterogeneous graph (HG): formally, a heterogeneous graph is defined as a directed graph $G = (V, E, A, R)$ where each node $v \in V$ and each edge $e \in E$ are associated with their type mapping functions $\tau(v)\colon V \longrightarrow A$ and $\phi(e)\colon E \longrightarrow R$. A and R donate the sets of node types and edge types, respectively.

Task description: given the interaction data Y and the KG G, our task is to learn a function that can predict how likely a user would select an item while further alleviating popularity bias.

Definition 1. Popularity: we define the number of times an item rated by all users as its popularity.

Definition 2. Popular item: an item is a popular item if its popularity is in the top 20% of all items.

Definition 3. Niche user (N) [12]: a user is a niche user if she/he is in the bottom 20% regarding the ratio of popular items in her/his profile. For these users, more than half of their profile consists of nonpopular (long-tail) items.

Definition 4. Blockbuster-focused user (B) [12]: a user is a blockbuster-focused user if she/he is in the top 20% regarding the ratio of popular items in her/his profile. These users, on average, have most popular items in their profile.

Definition 5. Diverse user (D) [12]: a user is a diverse user if she/he is neither a niche user nor a blockbuster-focused user.

2.2. The Architecture of AWING. We now present the proposed AWING. As illustrated in Figure 2, it consists of four key components: (1) KG embedded with popularity nodes, which inserts popularity nodes into KG to enrich the relations of KG; (2) fine-grained user preferences modeling, which uses multiple preferences to profile user-item relationships and aligns each preference with the relation in knowledge graph embed with popularity nodes; (3) heterogeneous graph transformer, which fully models heterogeneity to maintain dedicated representations for different types of nodes and edges in the heterogeneous graph; (4) model prediction, which uses the mutual attention to predict how likely the user would adopt the item under each preference.

2.2.1. KG Embedded with Popularity Nodes. We first divide the items into K groups according to their popularity. Next, we create K popularity nodes, termed $pn_i, i \in \{1, \ldots, K\}$, representing these groups and connecting items to their corresponding nodes. In this way, a new relation, $(i, \text{popularity}, pn)$, is introduced to KG, which integrates popularity information into the knowledge graph. There also exits other relations and entities (come from the attributes of items) in KG. As shown in Figure 2, we denote the new graph as KGEPN (KG embedded with popularity nodes).

2.2.2. Fine-Grained User Preferences Graph. We aim to capture the intuition that multiple preferences influence the behaviors of users. Here, we frame the preference as the reason for users' choices of items, reflecting the commonality of all users' behaviors. Taking music recommendation as an example, possible preferences are diverse considerations on music attributes, such as artist, genre, or popularity mentioned above. Such intuition motivates us to model user-item relations at the granularity of preferences. Assuming P as the set of preferences shared by all users and n as the number of types in the set P, we can slice a uniform user-item relation into the n preferences and decompose each $(u, \text{interact\_with}, i)$ triple into $\{(u, p, i) \mid p \in P\}$, as illustrated in Figure 2, termed preference graph (PG for short). Since the preferences are expressed as latent vectors that are vague to deeper understanding, we set the number of preferences as that of relations in KGEPN and transfer the information on the relation in KGEPN to the preferences. Concretely, we utilize the Euclidean norm to align the preferences embeddings p (between users and items) and relations embeddings r (between items and entities) in KGEPN:

$$L_{\text{align}} = \sum_{p \in P} \|\mathbf{p} - \mathbf{r}\|_2^2. \tag{1}$$

2.2.3. Heterogeneous Graph Transformer. After we get PG and KGEPN mentioned above, we combined them into a new heterogeneous graph (HG); we developed heterogeneous graph transformer (HGT for short) on the HG. HGT aims to aggregate information from the neighbors of target node t. Such a process can be decomposed into two parts: message computation and message aggregation. We denote the output of the (l)-th HGT layer as $H(l)$ and the depth of HGT as L.

The message computation part incorporates the message matrix, $W_{\phi(e)}^{\text{MSG}}$, to alleviate the distribution differences of nodes and edges of different types. Based on a source node s and an edge e, HGT calculates the message passed by s on e by

$$\text{Message}(s, e, t) = M_{\tau(s)}\left(H^{(l-1)}[s]\right) \cdot W_{\phi(e)}^{\text{MSG}}, \tag{2}$$

where $M_{\tau(s)}$ is a unique linear projection for node type $\tau(s)$.

In message aggregation part, HGT first calculates the heterogeneous mutual attention between source node s and target node t to control the influences of s on t. The attention mechanism was first proposed by the Google team to classify images [17]. Now, it is widely used in graph neural networks in recommender system [18]. HGT utilizes a unique linear projection ($Q_{\tau(t)}$ or $K_{\tau(s)}$) for each type of node and a distinct edge-based matrix $W_{\phi(e)}^{ATT} \in \mathrm{R}^{d \times d}$ for each edge type

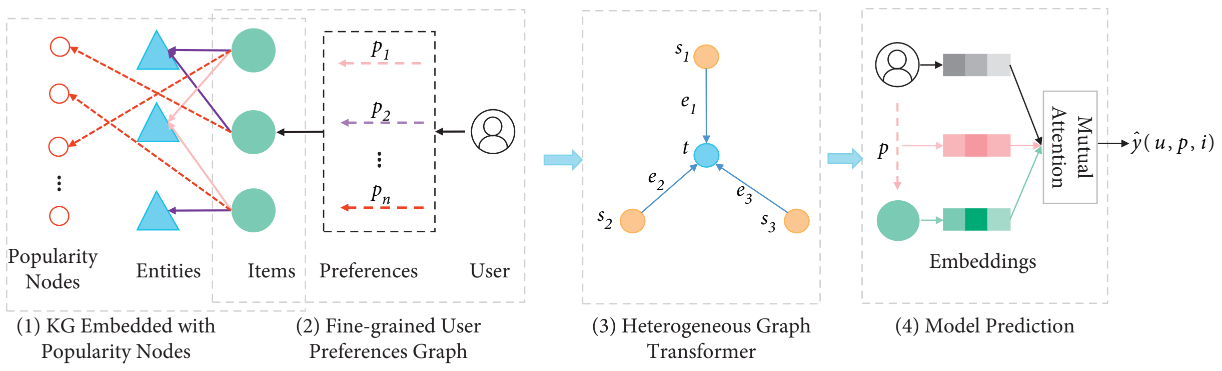

FIGURE 2: The overall architecture of AWING. Different colored arrows indicate different preferences and relations. It is best viewed in color.

$\phi(e)$ to model the distribution differences maximally; d is the size of dimension in the head of attention mechanism. The model can capture different semantic relations even between the same node type pairs, e.g., multiple preferences between the same user-item pair as mentioned above. Specifically, we calculate the heterogeneous mutual attention for each edge (s, e, t) by

$$\text{Attention}(s, e, t) = \underset{\forall s \in N(t)}{\text{softmax}}\left(\frac{K_{\tau(s)}\left(H^{(l-1)}[s]\right) \cdot W_{\phi(e)}^{\text{ATT}} \cdot Q_{\tau(t)}\left(H^{(l-1)}[t]\right)}{\sqrt{d}}\right), \tag{3}$$

where $N(t)$ denotes the one-hop neighbors of node t.

Next, HGT uses the attention vector as the weight to average the corresponding messages from the source nodes and get the updated vector $\widetilde{H}^{(l)}[t]$ as

$$H^{(l)}[t] = \sum_{s \in N(t)} (\text{Attention}(s, e, t) \cdot \text{Message}(s, e, t)). \tag{4}$$

We opt for the pairwise BPR loss to train HGT. Specifically, we encourage the score between node s and node t to be higher than the score between node s and random node t':

$$\begin{aligned} L_{\text{BPR}} &= \sum_{(s,e,t) \in HG} \sum_{(s,e,t') \in HG'} -\ln\sigma\left(\hat{y}(s, e, t) - \hat{y}(s, e, t')\right), \\ \hat{y}(s, e, t) &= \frac{K_{\tau(s)}\left(H^{(L)}[s]\right) \cdot W_{\phi(e)}^{\text{ATT}} \cdot Q_{\tau(t)}\left(H^{(L)}[t]\right)}{\sqrt{d}}. \end{aligned} \tag{5}$$

By combining the aligning loss and BPR loss, we minimize the following objective function to learn the model parameter:

$$L_{\text{HGT}} = L_{\text{BPR}} + \alpha L_{\text{align}} + \lambda \|\Theta\|_2^2, \tag{6}$$

where Θ is the set of model parameters and α and λ are two hyperparameters to control the aligning loss and L_2 regularization term, respectively.

2.2.4. Model Prediction. Benefiting from the individual edge-based matrix for each edge type, we can quantify the user's preference at a finer granularity. Specifically, as is exhibited in Figure 2, given final representations of user u and item i, for each preference, we calculate the corresponding score between u and i by

$$\hat{y}(u, p, i) = \frac{K_{\tau(i)}\left(H^{(L)}[i]\right) \cdot W_{\phi(p)}^{\text{ATT}} \cdot Q_{\tau(u)}\left(H^{(L)}[u]\right)}{\sqrt{d}}. \tag{7}$$

Then, we sum these different scores up as the probability of u adopting i:

$$\hat{y}_{ui} = \sum_{p \in P} \hat{y}(u, p, i). \tag{8}$$

In addition, we can combine multiple preferences as needed. To alleviate popularity bias, we remove a percentage of popularity preference for every user. Concretely, we design a weight w to manipulate how much popularity preference affects recommendation results:

$$\hat{y}_{ui/\text{adaptive}} = \sum_{p \in P} \hat{y}(u, p, i) - w \cdot \hat{y}(u, \text{popularity}, i), \tag{9}$$

where $w = 1 - RP(u)$ and $RP(u)$ is the ratio of popular items in the profile of user u. If a user is very interested in popular items, we will hardly remove her popularity preference, which can guarantee user experience.

3. Experiments

We provide empirical results to demonstrate the effectiveness of our proposed AWING. The experiments are designed to answer the following research questions (RQ):

(i) RQ1: how does AWING perform, compared to the state-of-the-art recommender models? Mainly, can AWING effectively alleviate popularity bias?

(ii) RQ2: how does the key hyperparameter K affect the recommendation performance?

(iii) RQ3: can AWING provide insights on user preferences and give an intuitive impression of explainability?

3.1. Experimental Settings

3.1.1. Datasets. To evaluate the effectiveness of AWING, we utilize two benchmark datasets: LastFM and DBbook-2014, which are publicly accessible and vary in terms of domain, size, and sparsity.

(i) LastFM: the dataset is the collection of listening records. The songs, which interacted with the current user only once, are treated as negative feedback. Because these songs may be misclicked by the user and are not helpful for improving the recommendation performance, to ensure the quality of the dataset, we use the 5-core setting, i.e., retaining users and items with at least five interactions.

(ii) DBbook-2014 [19]: the dataset consists of users and their binary feedback (1 for likes and 0 otherwise). Similarly, we use the 5-core setting to ensure that each user and item have at least five interactions.

Besides the user-item interactions, we need to construct item knowledge for each dataset. For LastFM and DBbook-2014, we follow the way in [20] to map items into freebase entities. We summarize the statistics of the two datasets in Table 1. We randomly select 80% of items associated with each user to constitute the training set and use all the remaining as the test set. The experiments are conducted with five-fold cross-validation for ten times, and the average results are reported.

3.1.2. Evaluation Metrics. Apart from a relevance-based metric (Recall@N) and a ranking-based metric (NDCG@N), we choose three metrics to measure the popularity bias.

(i) DGAP [12]: the group average popularity (GAP (g)) metric measures the average popularity of items in the profiles of users in a particular group g or their recommendation lists. Furthermore, the change in GAP (Δ GAP) is the amount of unwanted popularity in the recommendations imposed by the algorithms to each group:

$$\Delta\mathrm{GAP}(g) = \frac{GAP(g)_r - GAP(g)_p}{GAP(g)_p},$$
$$\mathrm{GAP}(g)_p = \frac{\sum_{u\in g}\sum_{i\in P_u}\phi(i)/|P_u|}{|g|}, \quad (10)$$
$$\mathrm{GAP}(g)_r = \frac{\sum_{u\in g}\sum_{i\in R_u}\phi(i)/|R_u|}{|g|},$$

where g is the group of users (in our case, it is either N, D, or B), $\phi(\cdot)$ is the popularity of a specific item, P_u is the list of items in the profile of user u, and R_u is the list of items in the recommendation result of user u.

(ii) APT@N [9]: the average percentage of tail items (APT) quantifies the ratio of nonpopular items in the recommendation lists:

$$\mathrm{APT@}N = \frac{1}{|U|}\sum_{u\in U}\frac{|R_u \cap I^{up}|}{N}. \quad (11)$$

(iii) AD@N [21]: aggregate diversity (AD) counts the total number of different items that have been recommended to at least one user:

$$A\ D@N = \frac{|\cup_{u\in U} R_u@N|}{|I|}. \quad (12)$$

3.1.3. Comparison Method. We compare our proposed AWING with the following baselines:

(i) BPRMF [22] is a classical CF method that only uses the user-item ratings for the recommendation, assuming that users tend to assign higher ranks to observed items.

(ii) BPRMF [23] is a GCN-based general recommendation model that leverages the user-item proximity to learn node representations and generate recommendations, which is reported as the state-of-the-art method.

(iii) KTUP [24] employs TransH on user-item interactions and KG triplets simultaneously to learn user preference and perform KG completion.

(iv) IPS–CN [25] adds normalization, which also achieved lower variance than plain IPS, at the expense of introducing a small amount of bias.

(v) ESAM [10] regards popular and nonpopular items as the source and target domains, respectively, and introduces three regularization terms for transferring the knowledge from these well-trained popular items to the long-tail items.

Table 1: Statistics of the datasets.

		LastFM	DBbook-2014
User-item interactions	#Usesrs	7,457	5,576
	#Items	15,226	2,680
	#Interactions	303,917	65,961
	#Sparsity	0.27%	0.44%
Knowledge 1 graph	#Entities	35,952	13,882
	#Relations	9	13
	#Triplets	464,567	334,511

3.1.4. Implementation Details. We implement our AWING model in PyTorch. We use AdamW [26] to train the model, where the initial learning rate is 0.001. In addition, we use 256 as the hidden dimension throughout the neural networks, and the batch size is fixed as 1024. As for other hyperparameters, we conduct a grid to confirm the optimal settings. More specifically, the coefficients of additional constraints (i.e., aligning loss and L2 regularization) α and λ are searched in $\{10^{-5}, 10^{-4}, \ldots, 10^{-1}\}$ and the number L of HGT layers is tuned in $\{1, 2, 3\}$. Finally, we set $\alpha = 10^{-2}$, $\lambda = 10^{-3}$, and $L = 2$ in our experiments.

3.2. Overall Performance Comparison (RQ1). Table 2 shows the best recommendation performance of all models on two datasets. In particular, AWING-APS is the variant of AWING which removes a percentage of popularity preference when recommended. In addition, the bold numbers indicate the best in each row and the *underlined* values indicate the second best. We can draw the following conclusions from the table.

Firstly, the BPRMF model, the most basic model, has the worst performance on two datasets and has serious popularity bias. Although LightGCN performs better than BPRMF, its APT value is also meager, showing that the popularity bias is ubiquitous in recommender systems.

Next, the performance of the KG-aware baseline, KTUP, is better than LightGCN and BPRMF on all metrics, which demonstrates that the introduction of KG is beneficial not only for the recommendation but also for alleviating popularity bias. Nevertheless, the improvement of APT and GAP is not enough.

Besides, compared with the three models above, IPS-CN and ESAM improve a lot on APT, but they do not keep the ratio of popular and nonpopular items according to their profiles, which may hurt the user experience.

Finally, our proposed method AWING which outperforms all the compared baselines on both datasets in terms of recall and NDCG, which indicates that identifying fine-grained preferences is helpful for the recommendation. Moreover, our proposed method AWING-APS performs best among all models in the light of Δ GAP, AD, and APT, which verifies the significance of removing popularity preference to alleviate popularity bias. Especially, AWING-APS has a shallow value of Δ GAP. In other words, AWING-APS keeps a similar ratio of popular and nonpopular items, which guarantees the user experience well. It should be noted that although AWING-APS sacrifices a small amount of recall and NDCG, this is negligible compared to IPS-CN and ESAM.

We can also find that, in the two datasets, the former dataset is sparser. The proposed model performs worse than the latter in the metrics of ranking task (Recall@N and NDCG@N) for the recommendation. However, for most metrics to measure the popularity bias, e.g., AD and APT, the improvement on the LastFM is greater than the DBbook-2014.

3.3. Parameter Sensitivity Analysis (RQ2). In this section, we investigate the impact of the number of popularity nodes K for popularity bias. In this experiment, we tune K in the range of [2, 13] with a step of 1 to report the corresponding performance. From Figure 3, we observe that AWING-APS achieves the best performance when $K = 6$ and 7 on two datasets, respectively. This is because a too small K does not have enough capacity to distinguish the different degrees of popularity, while a too-large K causes sparse data in each group and adversely suffers from overfitting.

3.4. Case Study (RQ3). An important benefit of attention recommender system is the explainability of the results [27]. In the same way, benefiting from the HGT, we can infer the fine-grained user preferences on the target item. Towards this end, we present an example of LastFM to give an intuitive impression of our explainability. We randomly selected one user, u306, and two relevant music m749 and m1364 (from the test, unseen in the training phase). Figure 4 shows the visualization of the example. AWING searches for the most influential preference based on the attention scores (cf. (7)). Thus, it explains this behavior as user u306 selects music m749 since it matches her interest in the featured artist. Similarly, we can infer that u306 chooses m1364 just because of its popularity.

4. Related Work

In recent years, with the development of recommender systems, more and more attention has been paid to the fairness of recommender systems. Popularity bias is one of the critical factors affecting its fairness. The problem of popularity bias and the challenges it creates for the recommender systems has been well studied by other researchers [5, 28]. Authors in the mentioned works have mainly explored the overall accuracy of the recommendations in the presence of long-tail distribution in rating data.

Table 2: Performance comparison of different recommendation models.

Dataset	Metric	BPRMF	LightGCN	KTUP	IPS-CN	ESAM	AWING	AWING-APS
LastFM (@10,%)	Recall	5.17	5.82	6.75	6.49	5.64	**7.31**	6.86
	NDCG	5.39	5.48	6.04	5.32	5.59	**6.94**	5.89
	Δ GAP (N)	92.67	80.49	60.83	38.13	*25.84*	59.08	**5.21**
	Δ GAP (D)	95.71	87.25	57.39	53.04	*31.82*	64.26	**7.56**
	Δ GAP (B)	87.52	82.98	68.52	−60.49	*−55.25*	63.94	**−4.81**
	AD	5.62	12.07	17.61	16.47	*18.15*	16.24	**24.28**
	APT	1.56	3.76	14.79	23.59	*25.33*	14.66	**36.06**
DBbook (@10,%)	Recall	17.23	20.45	*23.15*	19.45	18.92	**24.62**	23.02
	NDCG	18.86	22.67	*25.92*	22.62	21.53	**27.62**	25.27
	Δ GAP (N)	88.77	94.14	53.32	*25.96*	27.71	55.62	**2.18**
	Δ GAP (D)	85.48	83.46	61.28	35.09	*32.89*	62.52	**8.03**
	Δ GAP (B)	75.49	79.71	43.76	*−34.28*	−43.27	42.24	**−5.33**
	AD	9.58	11.26	14.36	17.33	*17.61*	15.38	**19.78**
	APT	2.91	5.51	7.95	12.06	*13.17*	12.27	**17.68**

The bold numbers indicate the best in each row and the underlined values indicate the second best.

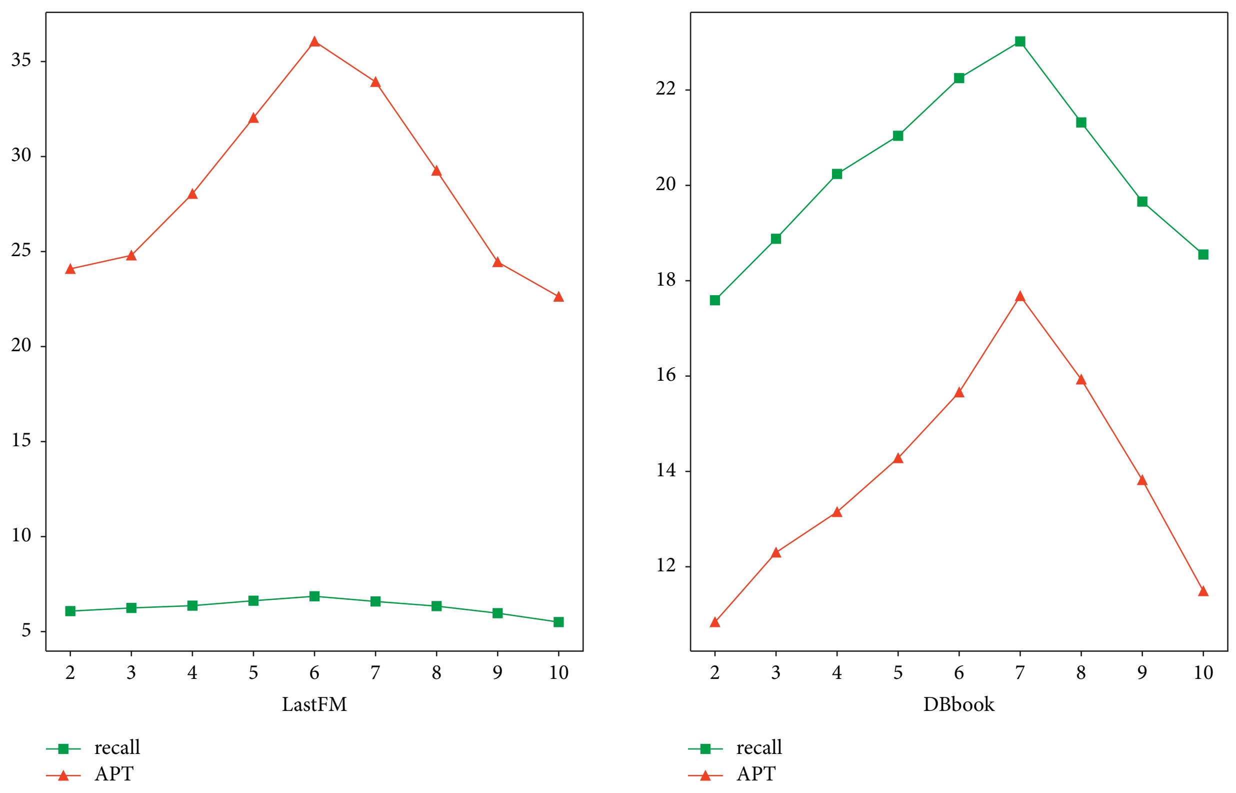

Figure 3: Performance impact of *K*.

Moreover, some other researchers have proposed algorithms that can control this bias and give more chances for non-popular items to be recommended [9, 29, 30]. Gruson et al. [25] and Joachims et al. [11] use IPS (Inverse Propensity Score) to eliminate popularity bias by reweighting each instance according to item popularity. Besides, Abdollahpouri et al. [9] provide a regularization term to control popularity bias. Recently, Chen et al. [10] address this problem from domain adaptation, regarding popular and nonpopular items as the source and target domains. In this work, however, we focus on alleviating popularity bias from the perspective of users. That is, we want to take into account the personal interest in popular and nonpopular items.

The recommendation system based on knowledge graph has attracted extensive attention of researchers now. This method can not only improve the accuracy of the recommendation system but also provide explanations for the recommendation results. Zhang et al. [31] proposed CKE with three-information sources: structural information, textual information, and visual information, which shows the structure information of the knowledge graph can enhance the semantic information of the item embedding.

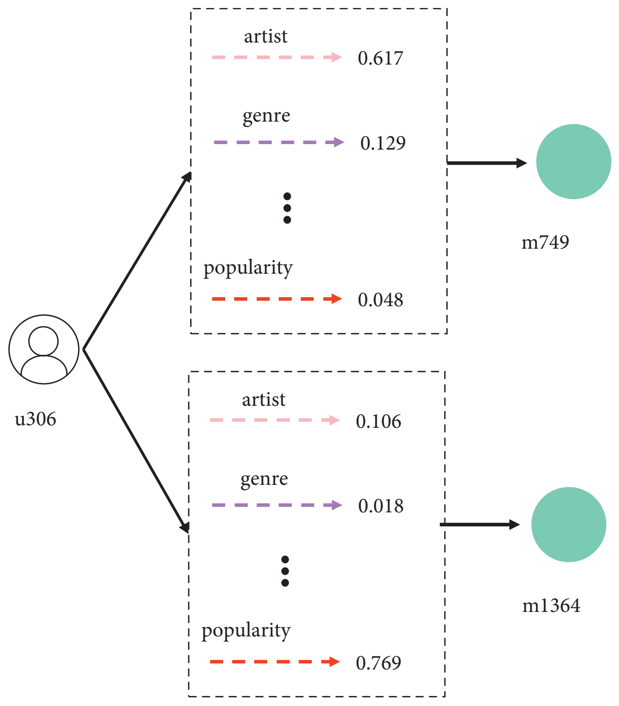

Figure 4: Real example from LastFM.

Wang et al. [32] used KGAT to obtain the item representations and propagate them on the user-item interaction graph with the graph attention mechanism. Huang et al. [33] constructed multiple metapaths from users to entities on the interaction and knowledge graph to obtain user representation. Tu et al. [34] proposed KCAN, which uses the knowledge graph to help automatically distill the knowledge graph into the target-specific subgraph and get the refined node representations. They all did not consider introducing knowledge graph embedded with popularity nodes.

5. Conclusion

This paper aims to utilize KG embedded with popularity nodes to alleviate popularity bias by identifying the fine-grained preferences of users. Our method first constructs a heterogeneous graph that combines KG, preference graph, and popularity nodes. Secondly, we use heterogeneous graph transformer over the heterogeneous graph while aligning fine-grained preferences with the relations in KG to learn the user/item/preference embeddings and the parameters of the mutual attention. Finally, according to the user's interest in popular items, we adaptively remove popularity preference to eliminate popularity bias. In the future, a valuable direction is to consider the dynamic change of popularity, instead of keeping the popularity constant.

Data Availability

The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of the manuscript.

Acknowledgments

This work was supported in part by the National Key Research and Development Project under grant 2019YFB1706101, National Natural Science Foundation under grant 72161005, and research program of Chongqing Technology Innovation and Application Development under grant CSTC2019jscx-zdztzxX0031.

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, IEEE, vol. 17, no. 6, pp. 734–749, 2005.
[2] X. Su and M. k. Taghi, "A survey of collaborative filtering techniques," *Advances in Artificial Intelligence*, vol. 2009, Article ID 421425, 19 pages, 2009.
[3] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, pp. 135–142, ACM, Barcelona, Spain, September 2010.
[4] W. Wang, S. Wang, and J. Huang, "Privacy Preservation for Friend-Recommendation Applications," *Security and Communication Networks*, vol. 2018, Article ID 1265352, 11 pages, 2018.
[5] Y. J. Park and A. Tuzhilin, "The long tail of recommender systems and how to leverage it," in *Proceedings of the Acm Conference on Recommender Systems, Recsys*, October 2008.
[6] HetRec2011, https://grouplens.org/datasets/hetrec-2011, 2011.
[7] G. Shani and A. Gunawardana, *Evaluating Recommendation Systems. Recommender Systems Handbook*Springer, Boston, MA, USA, 2011.
[8] I. Jones, R. Wang, J. Han, and H. Liu, "Community Cores: Removing Size Bias from Community Detection," in *Proceedings of the Tenth International AAAI Conference on Web and Social Media*, AAAI press, Cologne, Germany, May 2016.
[9] H. Abdollahpouri, R. Burke, and B. Mobasher, "Controlling popularity bias in learning-to-rank recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pp. 42–46, ACM, Como Italy, August 2017.
[10] Z. Chen, R. Xiao, C. Li, G. Ye, G. Sun, and H. E. Deng, "Discriminative domain adaptation with non-displayed items to improve long-tail performance," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 579–588, SIGIR, NY, United States, July 2020.
[11] T. Joachims, A. Swaminathan, and T. Schnabel, "Unbiased learning-to-rank with biased feedback," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 781–789, WSDM, NY, United States, February 2017.
[12] H. Abdollahpouri, M. Mansoury, and R. Burke, "The Unfairness of Popularity Bias in Recommendation," 2019, https://arxiv.org/abs/1907.13286>.

[13] L. Ehrlinger and W. Wolfram, "Towards a definition of knowledge graphs," *SEMANTiCS (Posters, Demos, SuCCESS)*, vol. 48, pp. 1–4, 2016.

[14] J. M. Gomez-Perez, J. Z. Pan, G. Vetere, and H. Wu, "Enterprise knowledge graph: an introduction, Exploiting linked data and knowledge graphs in large organisations," *Exploiting Linked Data and Knowledge Graphs in Large Organisations*, Springer, New York, NY, USA, pp. 1–14, 2017.

[15] O. Tal and Y. Liu, "A Joint Deep Recommendation Framework for Location-Based Social Networks," *Complexity*, vol. 2019, Article ID 2926749, 2 pages, 2019.

[16] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *Proceedings of The Web Conference 2020*, pp. 2704–2710, WWW, Taipei, Taiwan, April 2020.

[17] V. Mnih, N. Heess, and A. Graves, "Recurrent models of visual attention," in *Proceedings of the Advances in Neural Information Processing Systems*, NISPS, San Francisco, CA, USA, December 2014.

[18] P. Veličković, G. Cucurull, and A. Casanova, "Graph Attention Networks," 2017, https://arxiv.org/abs/1710.10903>.

[19] 2014 http://challenges.2014.eswc-conferences.org.

[20] X. Zhao, G. He, H. Dou, J. Huang, S. Ouyang, and J. Wen, "KB4Rec: A Dataset for Linking Knowledge Bases with Recommender Systems," 2018, https://arxiv.org/pdf/1807.11141.pdf>.

[21] G. Adomavicius and Y. YoungOk Kwon, "Improving aggregate recommendation diversity using ranking-based techniques," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 896–911, 2012.

[22] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: bayesian personalized ranking from implicit feedback," in *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pp. 452–461, AUAI Press, Quebec, Canada, June 2009.

[23] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and Powering Graph Convolution Network for recommendation," 2020, https://arxiv.org/abs/2002.02126>.

[24] Y. Cao, X. Wang, X. He, and Z. Hu, "Tat-Seng Chua. Unifying knowledge graph learning and recommendation: towards a better understanding of user preferences," in *Proceedings of the Web Conference*, pp. 151–161, WWW, San Francisco, CA, USA, May 2019.

[25] A. Gruson, P. Chandar, C. Charbuillet et al., "Offline evaluation to make decisions about playlist recommendation algorithms," in *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, pp. 420–428, WSDM, Melbourne, Australia, February, 2019.

[26] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *proceedings of the 7th International Conference on Learning Representations*, ICLR, New Orleans, LA, USA, May 2019.

[27] J. Bastings and F. Katja, "The elephant in the interpretability room: why use attention as explanation when we have saliency methods?," 2020, https://arxiv.org/abs/2010.05607>.

[28] E. Brynjolfsson, Y. J. Hu, and M. D. Smith, "From Niches to Riches: Anatomy of the Long Tail," *Social Science Electronic Publishing*, vol. 47, 2006.

[29] H. Abdollahpouri, R. Burke, and B. Mobasher, "Managing popularity bias in recommender systems with personalized re-ranking," in *Proceedings of the Florida AI Research Symposium*, pp. 413–418, FLAIRS, Beach Resort in Sarasota, FL, US, May, 2019.

[30] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma, "Correcting popularity bias by enhancing recommendation neutrality," in *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys, Silicon Valley, CA, USA, October 2014.

[31] F. Zhang, N. J. Yuan, and D. Lian, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 353–362, SIGKDD, San Francisco, CA, USA, 1August 2016.

[32] X. Wang, X. He, and Y. Cao, "Kgat: knowledge graph attention network for recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, pp. 950–958, SIGKDD, Anchorage, AK, USA, July 2019.

[33] R. Huang, C. Han, and L. Cui, "Entity-aware collaborative relation network with knowledge graph for recommendation," in *Proceedings of the 30th ACM International Conference on Information Knowledge Management*, pp. 3098–3102, ACM, Queensland, Australia, October 2021.

[34] K. Tu, P. Cui, D. Wang et al., "Conditional graph attention networks for distilling and refining knowledge graphs in recommendation," in *Proceedings of the 30th ACM International Conference on Information Knowledge Management*, pp. 1834–1843, ACM, Queensland, Australia, October 2021.