Mitigating Popularity Bias for Users and Items with Fairness-centric Adaptive Recommendation

ZHONGZHOU LIU, School of Computing and Information Systems, Singapore Management University, Singapore
YUAN FANG*, School of Computing and Information Systems, Singapore Management University, Singapore
MIN WU*, Institute for Infocomm Research, A*STAR, Singapore

Recommendation systems are popular in many domains. Researchers usually focus on the effectiveness of recommendation (e.g., precision) but neglect the popularity bias that may affect the fairness of the recommendation, which is also an important consideration that could influence the benefits of users and item providers. A few studies have been proposed to deal with the popularity bias, but they often face two limitations. Firstly, most studies only consider fairness for one side—either users or items, without achieving fairness jointly for both. Secondly, existing methods are not sufficiently tailored to each individual user or item to cope with the varying extent and nature of popularity bias. To alleviate these limitations, in this paper, we propose FAiR, a **f**airness-centric model that **a**dapt**i**vely mitigates the popularity bias in both users and items for **r**ecommendation. Concretely, we design explicit fairness discriminators to mitigate the popularity bias for each user and item locally, and an implicit discriminator to preserve fairness globally. Moreover, we dynamically adapt the model to different input users and items to handle the differences in their popularity bias. Finally, we conduct extensive experiments to demonstrate that our model significantly outperforms state-of-the-art baselines in fairness metrics, while remaining competitive in effectiveness.

CCS Concepts: • **Information systems** → **Recommender systems**; • **Applied computing** → *Sociology*; • **Social and professional topics** → *Race and ethnicity*.

Additional Key Words and Phrases: Fairness, Popularity bias

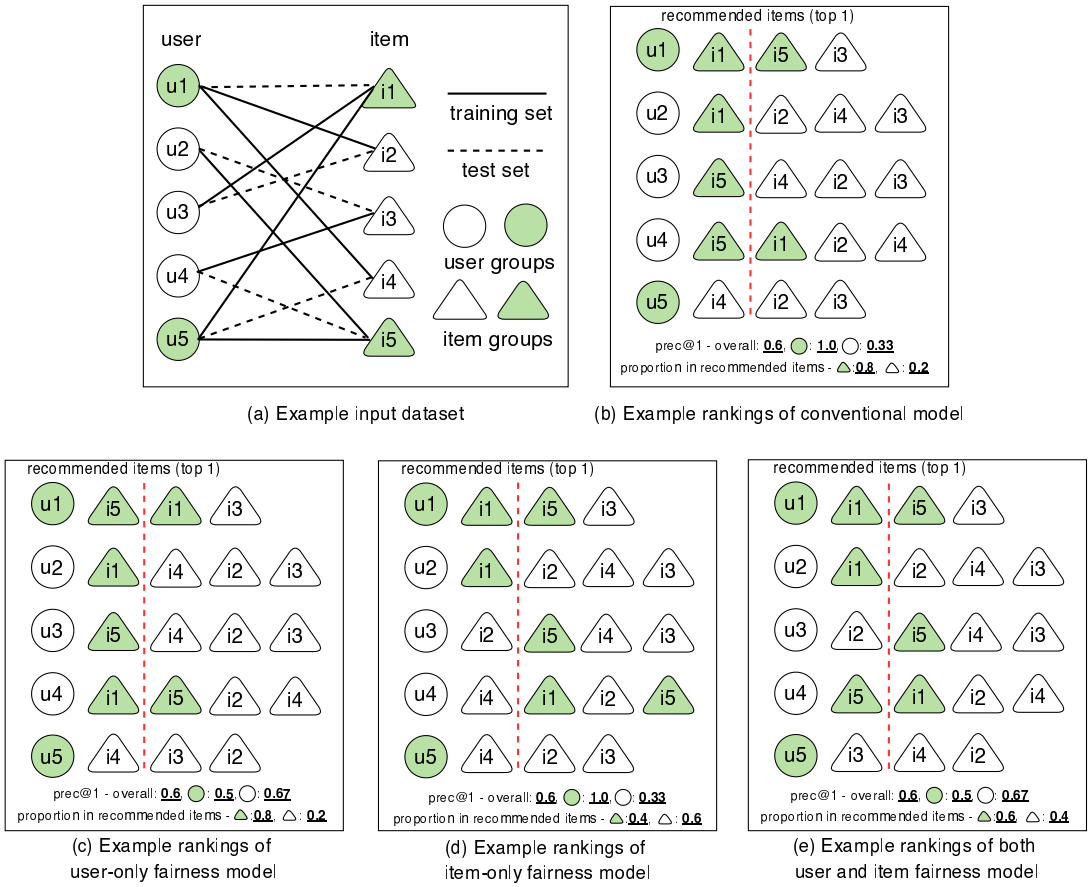
1 INTRODUCTION

As recommendation systems [52] have achieved widespread success in many real-world applications, more and more researches have revealed that conventional recommendation is often plagued by biases which may harm the ethics of recommendation outcomes and lead to negative social impacts [9]. One of the most well-known biases is called the *popularity bias*: Users and items are distributed unevenly in the training data (often following some long-tailed distribution), which means popular users or items would obtain higher weights during model training, making the model biased toward those popular ones [2].

The popularity bias would hurt both users and item providers alike, and we illustrate the impact with a toy example in Fig. 1. In this toy example, consider two user groups of different popularity, based on the number of interactions each user has in the training data. For instance, in Fig. 1(a), users with two interactions in the training data (i.e., solid edges) belongs to the popular group (denoted by solid circles), while users with only one interactions in the training data belongs to less popular group (denoted by hollow circles). Similarly, there are also two groups of items with different popularity in the training data (denoted by solid and hollow triangles, respectively). Suppose for each user, we rank his/her non-interacted items in the training set, and recommend the top-1 item to the user. We regard the dotted edges in Fig. 1(a) as the ground truth in the testing set. Due to the uneven distribution of data, Fig. 1(b) reveals the potential outcomes of a conventional recommendation model (i.e., one that does not consider fairness such as NCF [20]). On

*Corresponding authors

Authors' addresses: Zhongzhou Liu, zzliu.2020@phdcs.smu.edu.sg, School of Computing and Information Systems, Singapore Management University, 80 Stamford Rd, Singapore, 178902; Yuan Fang, yfang@smu.edu.sg, School of Computing and Information Systems, Singapore Management University, 80 Stamford Rd, Singapore, 178902; Min Wu, wumin@i2r.a-star.edu.sg, Institute for Infocomm Research, A*STAR, 1 Fusionopolis Way, Singapore, 138632.



(a) Example input dataset (b) Example rankings of conventional model

(c) Example rankings of user-only fairness model (d) Example rankings of item-only fairness model (e) Example rankings of both user and item fairness model

Fig. 1. A toy example illustrating the effect of popularity bias in different models. The bipartite graph in (a) represents the user-item interaction data, where the solid/dotted edges represent the training/test sets, respectively. Solid circles represent the group of popular users with more interactions in the training set, and hollow circles represent the group of less popular users with fewer interactions in the training set. Items are similarly grouped, where the solid/hollow triangles represent the popular/less popular groups of items, respectively.

the user side, the mean precision of the popular user group (solid circle) is 1.0 while that of the less popular group is only 0.33 (hollow circle). Although the overall mean precision of all users stands at 0.6 which seems reasonable, the model is much more effective and thus useful for the popular users than for the less popular users. On the item side, the popular items often receive disproportionally more "exposure" (i.e., being recommended more frequently) than the less popular items. For instance, the popular item group (solid triangle) only accounts for 40% of all items in the toy example and $\frac{4}{7}$ of all the interactions in the training set, but they get an overrated 80% exposure in the top-1 recommendation. To sum up, users in a less popular group are unfairly disadvantaged by virtue of the access to lower quality recommendations. At the same time, items from small local retailer tend to be less popular than those from large chain businesses, resulting in disproportionally less exposure in the recommendation platform, which means small local businesses systematically and unfairly suffer from the popularity bias. What could be even more worrying

is, the popularity bias would be further amplified due to the feedback loop in recommendation, where the resulting user-item interactions will be used as new training data [28]. Besides, popularity bias may also contribute to other pernicious effects such as the *filter bubble*, which is a phenomenon that "users become isolated from the information that varies from their dominant preferences" [14]. It typically occurs when a recommendation system only exposes items that are narrowly aligned with a user's historical interests, which eventually decreases the user's satisfaction toward the recommendation system [15]. Thus, filter bubble can be understood as a form of item over-exposure related to the presence of popularity bias in a user's historical interactions.

In this paper, we study the *fairness* issues arising from the popularity bias[1], aiming to mitigate the popularity bias in the recommendation results. As a branch of algorithmic fairness, popularity bias has attracted increasing attention in recent years [3, 53]. A common idea is to "reshape" the population [25] through various means such as propensity scoring [17] and causal inference [39, 53], in order to smooth the impact of the unevenness of popularity. The smoothing would mitigate the unevenness between the popular and less popular groups, thereby generating equitable outcomes for all groups toward fairness [4, 5]. For instance, Fig. 1(c) illustrates "fairer" recommendation results for users than a conventional model in (b) would produce, where the difference in prec@1 between the two user groups reduces from 0.67 in (b) to 0.17 in (c). That is, the two groups of users shown in (c) receive recommendations of similar effectiveness [24], and thus can both benefit from the recommendations to a similar extent. Further note that the overall prec@1 in both (b) and (c) maintains the same, which means (c) is able to achieve fairness for users without compromising the overall effectiveness.

However, most existing works only consider the popularity bias from the user-side as we have shown in Fig. 1(c), but ignore its negative impact on the item-side. This would lead to the under-exposure of less popular items. For instance, in Fig. 1(b) and 1(c), the less popular item group (hollow triangle) occupies only 20% of the top-1 recommendation, which is disproportionally less than their proportion accounting for $\frac{3}{7}$ of the total interactions or 60% of the total items in the training set. In many scenarios, item-side fairness is as important as user-side fairness, in order to give small item providers (e.g., neighborhood businesses in an e-commerce platform or minority authors in a book/library service) equitable exposure. To date, only very few works have dealt with the impact of popularity bias on the item-side [46, 54], which expects both popular and unpopular items to have more comparable share of exposure. For example, in Fig. 1(d), the less popular item group now occupies 60% of the top-1 recommendation, which is much closer to their proportion in the total items and/or training data. However, these models with *only* item-side fairness could still compromise user-side fairness, resulting in recommendations of divergent effectiveness for the two user groups, e.g., as shown in (d). Ideally, both user- and item-side fairness are important and should be considered jointly, as illustrated by the example ranking in Fig. 1(e).

Hence, toward fairness w.r.t. the popularity bias in recommendation systems, the first open challenge is: *How can we jointly mitigate the popularity bias for both user- and item-side?* A naïve solution is a multi-phase model [32] to tackle the bias from each side one by one, but it would ignore the intrinsic relationships between the two sides. Moreover, the popularity bias may vary in extent and nature for each user and item, given their different frequencies in the training data and individual characteristics. Despite various mechanisms to tackle popularity bias, previous methods train one global model to handle the bias for all users and items, in which the model is frozen after training completes. That means the same model is applied in a uniform manner to any user or item alike at test time, lacking the opportunity to

[1]While fairness issues can also be related to other biases or consequences [9], to avoid any ambiguity hereafter we use the term "fair(ness)" to specifically describe the group fairness [33] of recommendation models or results that are not affected by the popularity bias, following some previous work [1]. We will give formal definitions of fair(ness) in Sect. 3

fully adapt to each individual user or item with potentially different frequencies or characteristics. Hence, the second open challenge is: *How can we adaptively mitigate the bias to tailor to each individual user or item?* In other words, an adaptive debiasing model is warranted to account for the uniqueness of each input user or item in both training and testing.

To address these challenges, we propose FAiR, a **F**airness-centric model that **A**dapt**i**vely mitigates the popularity bias in both users and items for **R**ecommendation. For the first challenge, we propose two *explicit discriminators* to simultaneously detect the bias in users and items locally, and an *implicit discriminator* to preserve fairness globally in an end-to-end framework. This design allows us to mitigate popularity bias for both user- and item-side jointly so that their intrinsic relationships can be captured, whilst preserving the overall effectiveness of recommendation in the end-to-end training process. Despite using a simple architecture with multilayer perceptrons, empirical results show that the explicit and implicit discriminators complement and collaborate well with each other to improve fairness from the two sides. For the second challenge, we resort to Feature-wise Linear Modulation (FiLM) [34] to modulate the generators so that they become conditioned on the input user or item. To our best knowledge, this is the first attempt to utilize FiLM layers for recommendation fairness. Specifically, the input conditioning enables the generators to sufficiently adapt to the differences and individual characteristics associated with each user or item.

In summary, the contribution of this work is threefold.

- We identify the impacts of popularity bias for both user- and item-side, and propose to jointly mitigate the two-sided popularity bias in our model FAiR in an end-to-end fashion.
- We recognize that the extent and nature of popularity bias often vary remarkably for different users or items, and propose to employ a FiLM mechanism for adaptive debiasing tailored to each individual user or item in our model FAiR.
- We conduct comprehensive experiments to show that FAiR can significantly improve the fairness of recommendation at a competitive level of effectiveness.

2 RELATED WORK

In this section, we will review related work in three aspects, including general fairness learning, fairness for popularity bias in recommendation, and other related problems and approaches.

2.1 General Fairness Learning

As an emerging topic in the wider machine learning community, different fairness learning approaches have been proposed. One family of methods only involve pre- and post-processing strategies [12, 22] to simply alter the original distribution of training data, or re-rank the predictions to enhance the fairness. Such methods do not address the algorithmic cause of unfairness, thus their solutions cannot fundamentally address the bias. Furthermore, model accuracy could be significantly compromised due to the pre- or post-processing steps. Other methods integrate various fairness objectives or constraints into deep models including auto-encoders [27], graph neural networks [47] and generative adversarial networks [51], which are able to mitigate biases at the model level and achieve desirable outcomes in term of both accuracy and fairness for different problems such as classification [5] and representation learning [27].

It is worth-nothing that a typical objective of previous fairness learning is to debias certain sensitive attributes such as demographic information [5, 27, 47], in order to enforce statistical parity [13, 50] or equal opportunity [18]

for different groups. While sensitive attributes may also be subject to uneven distributions in the training data, in this paper we specifically deal with the popularity bias not particularly associated with any attribute.

2.2 Fair Recommendation for Popularity Bias

The popularity bias in recommendation systems is less explored until recent years. Like in general fairness learning, some studies resort to pre- or post-processing techniques, such as data re-balancing [37, 49] to change the distribution of the training data, or adjust the ranking of some items to maintain a balance between popular and unpopular groups [24, 41]. To better cope with the popularity bias in an end-to-end manner to achieve both fair and effective recommendations, adversarial learning [16] has been exploited. A typical adversarial framework consists of two players: generator and discriminator. The two players compete with each other in a minimax game, where the generator aims to mitigate bias from the input, and the discriminator aims to recover any residual bias in the generator's output. Such adversarial learning has been shown as a powerful strategy in improving recommendation accuracy or fairness [5, 7, 54]. However, existing adversarial strategies deal with the fairness for either users [5] or items [54] only, but in reality both sides can suffer from the popularity bias. Note that a small number of works deal with two-sided fairness [30, 32] for both items and users. However, they have significant differences from our work. On one hand, Mondal et. al. [30] have proposed two-sided fairness for non-personalized recommendation. Specifically, their goal is to find a global list of recommendations which would be liked by most of the users, whereas our work makes personalized recommendation to each user. On the other hand, Patro et. al. [32] define the user-side fairness as envy-freeness [35] and the item-side fairness as maximin share of exposure, which differ from our goal of mitigating the popularity bias for equitable outcomes among different user or item groups (the formal definition of our fairness objective is given in Sect. 3). Besides, [32] is different from our paper in its non end-to-end optimization manner. Furthermore, a few studies utilize causal inference to mitigate the popularity bias [46, 53]. However, similar to existing adversarial strategies, they usually focus on one side fairness only. Moreover, these models are frozen once training is finished, lacking the flexibility of adapting to different input users and items.

In another line of popularity bias, some researchers focus on the missing-not-at-random (MNAR) problem in recommendation systems [6, 29, 38]. In the MNAR problem, it is hypothesized that "low ratings are much more likely to be missing from the observed data than high ratings" [42], and could be regarded as a fairness problem caused by uneven popularity across item ratings [6]. However, it is irrelevant to the user- and item-side fairness in our paper as defined in Sect. 3.2. Furthermore, existing counter-measures against the unevenness associated with low or high ratings are designed to improve the overall model effectiveness.

2.3 Other Fair Recommendation and Related Work

Some other perspectives of bias in recommendation have also been explored. For example, some works aim to tackle model bias [40, 48] caused by biased assumptions and priors of the model itself, or deal with the sensitive attributes of users in recommendation [47]. The feedback loop [28] is another widely explored topic, as users tend to interact with recommended items and these interaction data are often used as new training data. That means the presence of bias in the initial recommendation would be amplified during the feedback process. Moreover, there exist many diversity models for recommendation [10], whose goal is to maximize the exposure of different item groups in the recommendation results, which helps item-side fairness to some extent, but they do not aim for equal exposure between popular and unpopular groups. Besides, there is a branch of machine learning called imbalanced learning [11, 23], and some of which have also utilized the adversarial learning strategy [36]. Although the skewed data distribution caused

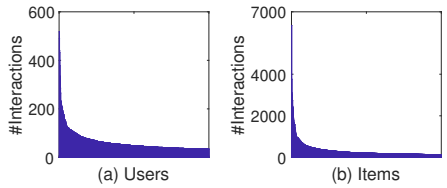


Fig. 2. Uneven popularity of users and items in the Amazon dataset.

by popularity bias is a key factor in both imbalanced and fairness learning, their objectives diverge. That is, imbalanced learning focuses on improving the overall model effectiveness, whereas fairness learning concerns about the differential treatment and aims to achieve uniform treatment of all popularity group.

3 MOTIVATION AND PROBLEM STATEMENT

In this section, we will motivate and formalize the fairness problem caused by popularity bias for both user- and item-side. First, we present a case study to further motivate the problem beyond the toy example in introduction. Then, we lay down the formal definitions of our fairness objectives in this paper.

3.1 Case Study

We use a case study on a Amazon review dataset to motivate the fairness concerns arising from the popularity bias in users and items. Specifically, we focus on the grocery and gourmet food category on Amazon[2], a popular public dataset in recommendation systems research. The detailed introduction of the Amazon dataset is given in Sect. 5.1.1. On this dataset, we first showcase the uneven popularity of users and items, and further deploy two conventional recommendation models to reveal the fairness concerns in the recommendation results.

First, for each user and item we count its number of user-item interactions in the dataset. The number of interactions can represent the popularity of a user or an item. We plot the distribution of their popularity (i.e., number of interactions) in descending order in Fig. 2(a) and (b) for the first 1,000 most popular users and items, respectively. It can be observed that very few users or items have extremely large number of interactions while the remaining majority of users or items only have limited interactions. In particular, if we focus on the group of top 5% most popular users/items in the dataset, we can observe that the very small proportion of users/items (5%) makes up a significant proportion of interactions (27%/53% for the group of top users/items as shown in Table 2). Such unevenness of popularity in users and items shows a typical characteristic of long-tailed distribution [31], which eventually leads to the popularity bias and the related fairness issues as we will show next [24, 43].

Next, we analyze how the unevenness of popularity can cause the so-called popularity bias and harm the fairness of the recommendation results, if a recommendation model does not recognize or mitigate the bias. To reveal the unfairness, we train and evaluate[3] two conventional models on the Amazon dataset, namely, CFI[4] [26] and NCF [20], which are based on representative methods including matrix factorization and neural networks. However, they only focus on the *overall* recommendation effectiveness and do not deal with the popularity bias.

[2]Henceforth we will simply call this subset "the Amazon dataset" or "Amazon".
[3]The detailed experimental settings will be described in Sect. 5.1.4.
[4]In this case study, we adopt the alternating least squares implementation of CFI.

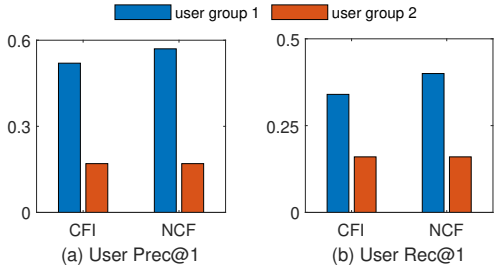


Fig. 3. Impact of popularity bias on users, which results in differential recommendation quality for user groups of different popularity. Recommendation quality is measured using (a) precision@1 and (b) recall@1.

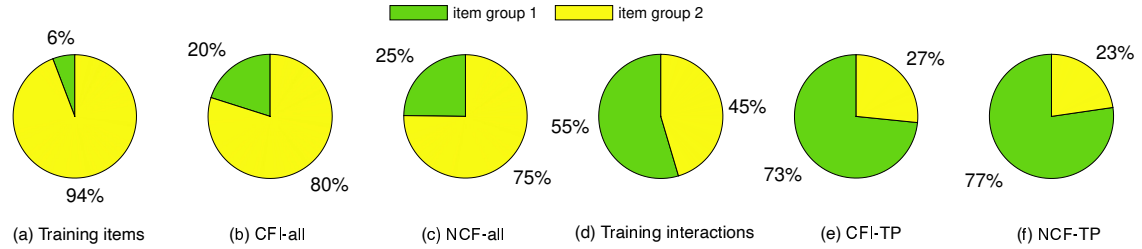


Fig. 4. Impact of popularity bias on items, which results in differential opportunity of exposure on the recommendation platform for item groups of different popularity. The figures present the proportional share of each item group in (a) training items, (b)/(c) all recommendations generated by CFI/NCF, (d) training interactions, (e)/(f) true-positive (TP) recommendations generated by CFI/NCF.

From the perspective of user-side fairness, we first divide the users into two groups: the top 5% most popular users in "user group 1" and the remaining 95% in "user group 2". We report the recommendation effectiveness in each user group including precision and recall on top-1 recommendation, denoted as prec@1 and rec@1 respectively, in Fig. 3. It can observed that, on average, users in group 1 receive significantly higher recommendation quality in terms of both prec@1 and rec@1 than users in group 2. The reason is that the training instances (i.e., user-item interactions) associated with users in group 1 are denser than those associated with users in group 2. More concretely, in the training data the number of interactions per user in group 1 is about 6 times of that in group 2. Hence, users in group 1 tend to have more robust and less noisy training data than users in group 2 do. This observation reflects how the popularity bias can lead to imbalanced effectiveness for users in groups of different popularity. As a result, users in group 1 (only representing 5% of the total users) benefit more from a superior recommendation service, while users in group 2 (95% of the total users) would be unfairly subjected to a substandard recommendation service. This differential treatment of users is a form of unfairness [24].

From the perspective of the item-side fairness, the items are similarly divided into two groups based on the 5%–95% splitting. In item-side fairness, we are concerned in the differential exposure across item groups [54]. As shown in Fig. 4, we compare the exposure of the two item groups w.r.t. their proportion in training data in two different ways. Firstly, in Fig. 4(a)–(c), we simply count the items without considering the ground-truth. That is, in (a), we report the proportion of items in the two groups in training data (6% belong to group 1, and 94% belong to group 2). However, items in group 1 get much more exposure in testing than in training, accounting for 20% or 25% of the top-1 recommendations generated by CFI in (b) or NCF in (c) in testing. Secondly, in Fig. 4(d)–(f), we only consider items with the so-called "true positive"

exposure, i.e., items belonging to the ground-truth interactions. Intuitively, here we investigate the impact of bias only on the effective recommendations, which could be more important to item providers. That is, in (d), we report the proportion of training interactions involving items in each group (55% of the interactions involve group 1 items, and 45% involve group 2 items). However, items in group 1 represent 73% or 77% of the *true positive* top-1 recommendations generated by CFI in (e) or NCF in (f) in testing, which is significantly more than the share of group 1 in the training interactions. It means popular items get disproportionally more exposure in testing than in training, in terms of not only raw item counts, but also the true positive counts. As a result, providers of the less popular items (e.g., small local business in an e-commerce platform, minority authors in a book/library service) are unfairly subjected to systematically less exposure than they deserve, a form of discrimination brought by the recommendation model.

3.2 Problem Statement

The unfairness issues revealed in our case study above is manifested by virtue of uneven popularity in different user or item groups. In this section, we formalize the problem statement on the mitigation of popularity bias for fairness-centric recommendation models. We first define the recommendation task, followed by the fairness problem on both user- and item-side.

A typical recommendation task involves a user set $\mathcal{U}$ and an item set $\mathcal{I}$, where users may interact with some items. As a user's subjective preferences on items are different, let $y_{u,i} \in \mathbb{R}$ denote the rating of user $u \in \mathcal{U}$ on item $i \in \mathcal{I}$. If $y_{u,i}$ is given for certain u and i in the dataset, u and i are said to have a user-item interaction which belongs to the ground truth; otherwise, we regard that there is no interaction between them. A recommendation system will generate a top-K recommendation list for each user u according to the predicted $\hat{y}_{u,i}$ for items in the test candidates.

Upon the recommendation task, in our paper, we investigate group-based fairness [33] caused by popularity bias on both user- and item-side. In particular, each user and item are assigned into a certain group. Suppose there are m user groups and n item groups, i.e.,

$$\mathcal{U} = G_1^{\mathcal{U}} \cup G_2^{\mathcal{U}} \cup \cdots \cup G_m^{\mathcal{U}}, \tag{1}$$

$$\mathcal{I} = G_1^{\mathcal{I}} \cup G_2^{\mathcal{I}} \cup \cdots \cup G_n^{\mathcal{I}}, \tag{2}$$

where all groups $G_j^{\mathcal{U}} \subset \mathcal{U}$ and $G_j^{\mathcal{I}} \subset \mathcal{I}$ are disjoint. The groups can be defined on popularity directly as we did in the case study, or on certain attributes that suffer from the popularity bias (e.g., gender of users, and vendor of items). It means that some groups can be much more popular than the others, resulting in unfairness for users and/or items in less popular groups. In this paper, we pursue group fairness that the algorithm should treat groups similarly, i.e., reduce differential treatment across groups [33].

More specifically, for the user-side fairness, as motivated earlier we expect that a fair model could retain similar recommendation quality or effectiveness for users in different groups, regardless of their popularity. Let $M(u)$ denote the effectiveness metric of recommendation (e.g., precision or recall) for user u. Following previous work [24], we give the following definition on the user-side fairness.

Definition 3.1 (User-oriented Group Fairness [24])**.** Given m user groups $G_1^{\mathcal{U}}, G_2^{\mathcal{U}}, \ldots, G_m^{\mathcal{U}}$, the user-oriented group fairness (UGF) requires

$$\mathbb{E}_{u \sim G_1^{\mathcal{U}}}[M(u)] = \mathbb{E}_{u \sim G_2^{\mathcal{U}}}[M(u)] = \ldots = \mathbb{E}_{u \sim G_m^{\mathcal{U}}}[M(u)]. \tag{3}$$

While this definition depicts an idealized binary classification of fairness, in practical models there often exist different degrees of unfairness on a continuous scale. Hence, to evaluate user-side fairness of a recommendation model, we measure the numerical difference in the recommendation effectiveness between different user groups. More concretely, given two user groups $G_1^{\mathcal{U}}$ and $G_2^{\mathcal{U}}$, the user-oriented group fairness (UGF) metric [24] w.r.t. an effectiveness metric M is defined as

$$\text{UGF}_M = \left| \frac{1}{|G_1^{\mathcal{U}}|} \sum_{u \in G_1^{\mathcal{U}}} M(u) - \frac{1}{|G_2^{\mathcal{U}}|} \sum_{u \in G_2^{\mathcal{U}}} M(u) \right|. \tag{4}$$

where the subscript M denotes an effectiveness metric for the recommendation. Smaller UGF values indicate that the difference in the effectiveness M between groups are smaller, thus better user-side fairness. More generally, when there are more than two groups, we can easily extend UGF_M by considering the average or maximum differences in M between pairwise groups.

Finally, for the item-side fairness, we expect that a fair recommendation model could balance the exposure rate and opportunity between different item groups in the top-K recommendation list. Let $P_j(K)$ be the probability of an item in the j-th item group $G_j^{\mathcal{I}}$ being ranked in the top-K recommendation list and $P_j(K|y=1)$ be the probability of a item in $G_j^{\mathcal{I}}$ being ranked among the true positives in the top-K recommendation list. Then, the two kinds of item-side fairness can be defined as below.

Definition 3.2 (Item Ranking-based Statistical Parity and Equal Opportunity [54])**.** Given n item groups $G_1^{\mathcal{I}}, G_2^{\mathcal{I}}, \ldots, G_n^{\mathcal{I}}$, the item ranking-based statistical parity (RSP) requires

$$P_1(K) = P_2(K) = \ldots = P_n(K), \tag{5}$$

and the item ranking-based equal opportunity (REO) requires

$$P_1(K|y=1) = P_2(K|y=1) = \ldots = P_n(K|y=1). \tag{6}$$

Unlike user-side fairness, there are two alternative definitions of item-side fairness, namely, RSP and REO, which are based on the concepts of statistical parity [13] and equal opportunity [18], respectively. They concern the fairness issue caused by popularity bias in different aspects: RSP encourages the equality of *overall exposure rate* among groups without factoring in user preferences; REO encourages the equality of *true positive exposure rate* among groups, which also aligns with user preferences on items. Both definitions are commonly adopted. On one hand, the former could benefit small item providers especially in raising their overall exposure, but their exposure may be less effective (e.g., with a lower conversion rate). On the other hand, the latter could benefit small item providers with more high-quality exposure, but does not help with their overall exposure. To investigate item-side fairness comprehensively, we adopt both definitions. Similar to user-side fairness, on practical models with non-binary fairness, we follow previous work [54] to evaluate the following two metrics corresponding to RSP and REO, respectively.

$$\text{RSP}\ @K = \frac{\text{std}(P_1(K), P_2(K), \ldots, P_n(K))}{\text{mean}(P_1(K), P_2(K), \ldots, P_n(K))}, \tag{7}$$

$$\text{REO}\ @K = \frac{\text{std}(P_1(K|y=1), P_2(K|y=1), \ldots, P_n(K|y=1))}{\text{mean}(P_1(K|y=1), P_2(K|y=1), \ldots, P_n(K|y=1))}, \tag{8}$$

where $P_j(K)$ can be calculated as the ratio between the number of top-K recommendations in the j-th item group and the total number of ranking candidates belonging to the j-th group, and $P_j(K|y=1)$ can be calculated as the number of

Table 1. Summary of main notations used.

Symbol	Definition
$\mathcal{U}, u$	Set of users and a user instance
$\mathcal{I}, i$	Set of items and an item instance
$\mathbf{x}_u, \mathbf{x}_i$	Original biased embedding for user u and item i
$\mathbf{x}'_u, \mathbf{x}'_i$	Debiased embedding for user u and item i
$G_j^{\mathcal{U}}, G_j^{\mathcal{I}}$	Sets of user and item groups
$\mathcal{F}$	Adaptive fairness filters
$\mathcal{D}^{\mathcal{U}}, \mathcal{D}^{\mathcal{I}}$	Explicit fairness discriminators
Δ	Implicit fairness discriminator
$\Theta_\bullet$	Parameters of different components
α_u, β_u	Scaling and shifting factors for modulation
$\mathbf{g}_u, \hat{\mathbf{g}}_u$	Ground-truth and predicted user group probabilities (categorical data)
$\mathbf{g}_i, \hat{\mathbf{g}}_i$	Ground-truth and predicted item group probabilities (categorical data)
$\mathbf{r}_0, \mathbf{r}_u$	reference rating scale and rating scale of user u
$y_{u,i}, \hat{y}_{u,i}$	Ground-truth and predicted interaction between u and i
$\delta_\phi, \hat{\delta}_\phi$	Ground-truth and predicted probability whether sampled rating belongs to $\mathbf{r}_0$

positive top-K recommendations in the j-th item group and the total number of positive ranking candidates belonging to the j-th group. Similar to UGF, lower values in RSP and REO indicate better item-side fairness.

In summary, the main objective of this paper is to reduce popularity bias in recommendation models, in order to achieve user- and item-side fairness simultaneously across different popularity groups. Of course, we also aim to preserve the competitiveness of the recommendation model, which should only incur minimal compromise in the overall model effectiveness.

4 PROPOSED METHOD

To address the fairness issues as stated in the last section, we present the framework as well as the details of the proposed FAiR model. The overall framework is shown in Fig. 5, and main notations used are shown in Table 1.

4.1 Overall Framework

As Fig. 5 shows, the overall framework consists of several parts: (a) input data; (b) adaptive fairness filters; (c) explicit and implicit fairness discriminators; (d) base recommendation model.

In Fig. 5(b), we design *adaptive fairness filters* for both users and items, which are able to adapt to each user/item by flexibly mitigating popularity bias of potentially different extents for the given user/item with the help of a FiLM layer. In Fig. 5(c), we design *adversarial fairness discriminators* that are either explicit (c1) or implicit (c2). On one hand, the explicit discriminators aim to tackle bias from a local perspective, targeting each user or item directly. On the other hand, the implicit discriminator aims to tackle bias from a global perspective, targeting the overall rating scale. Together, the fairness filters and discriminators are trained adversarially, i.e., the filters act as a set of generators and the discriminators play the role of adversary.

Lastly, in Fig. 5(d), we utilize a base recommendation model to predict user ratings toward items. Given a pair of user-item (u, i) with embeddings $\mathbf{x}_u, \mathbf{x}_i \in \mathbb{R}^d$, the fairness filters first generate their debiased embeddings $\mathbf{x}'_u, \mathbf{x}'_i \in \mathbb{R}^d$, respectively. The base model then operates on the debiased embeddings to generate fair recommendation outcomes.

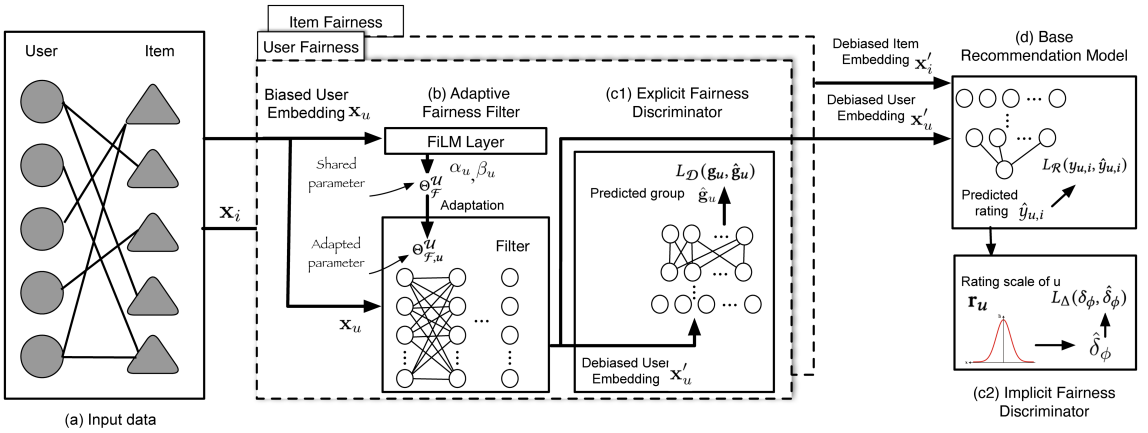


Fig. 5. Framework of the FAiR model. The input embeddings are first filtered by adaptive fairness filters (b) to produce debiased embeddings, based on which the base recommendation model (d) would make predictions on user-item pairs. Explicit and implicit discriminators (c1 & c2) are used in model training to detect any residual unfairness after applying the filters. Note that the diagram is focused on the view of user-side fairness only, and the view of item-side fairness is omitted as it is similar to user-side fairness with an adaptive fairness filter and explicit fairness discriminator.

In the following, we will elaborate on each component as well as the process of model training.

4.2 Adaptive Fairness Filters

A fairness filter is a nonlinear transformation $\mathcal{F} : \mathbb{R}^d \rightarrow \mathbb{R}^d$ on the original user or item embedding ($\mathbf{x}_u$ or $\mathbf{x}_i$), to reduce the popularity bias from them and generate a debiased embedding ($\mathbf{x}'_u$ or $\mathbf{x}'_i$). A straightforward solution is to materialize $\mathcal{F}$ with a neural network, as follows.

$$\mathbf{x}'_u = \mathcal{F}(\mathbf{x}_u; \Theta^{\mathcal{U}}_{\mathcal{F}}), \tag{9}$$

$$\mathbf{x}'_i = \mathcal{F}(\mathbf{x}_i; \Theta^{\mathcal{I}}_{\mathcal{F}}), \tag{10}$$

where $\mathcal{F}$ is a multi-layer perceptron (MLP), and $\Theta^{\mathcal{U}}_{\mathcal{F}}, \Theta^{\mathcal{I}}_{\mathcal{F}}$ denote the parameters of the MLP for user and item fairness filter, respectively.

However, as motivated in Sect. 3, different users (or items) often have varying distributions in popularity. Thus, it is inflexible and ineffective to mitigate the popularity bias for all users (or items) using a single shared model $\Theta^{\mathcal{U}}_{\mathcal{F}}$ (or $\Theta^{\mathcal{I}}_{\mathcal{F}}$). We propose to make the filters *adaptive*, in which the parameters $\Theta^{\mathcal{U}}_{\mathcal{F}}$ or $\Theta^{\mathcal{I}}_{\mathcal{F}}$ can adapt to different input user or item, respectively. In contrast, in conventional neural networks, the model is frozen once training completes, which means the same parameters are used in testing regardless of the input. Specifically, we achieve the adaptability with FiLM [34]. The basic idea of FiLM is to generate a set of scaling and shifting operators conditioned on the input feature, which are then used to modulate the original parameters of a model. As a result, the modulated parameters would better adapt to different input. In our context, the FiLM is conditioned on the original user (or item) embedding, and generates the scaling and shifting operators for the user (or item) fairness filter. Taking the user side as an example as shown in

Fig. 5(b), the FiLM layers output

$$\alpha_u = \text{FiLM}(\mathbf{x}_u; \Theta_\alpha^{\mathcal{U}}), \tag{11}$$

$$\beta_u = \text{FiLM}(\mathbf{x}_u; \Theta_\beta^{\mathcal{U}}), \tag{12}$$

where $\alpha_u \in \mathbb{R}^{d \times d}$ and $\beta_u \in \mathbb{R}^d$ are respectively the scaling and shifting operators for user u, and $\Theta_\alpha^{\mathcal{U}}$ and $\Theta_\beta^{\mathcal{U}}$ are the trainable parameters of the corresponding FiLM layers. The FiLM layers can be simply implemented as another MLP. Similar FiLM layers can also be employed for items, with trainable parameters $\Theta_\alpha^{\mathcal{I}}$ and $\Theta_\beta^{\mathcal{I}}$.

The scaling and shifting operators are used to adapt the shared parameters of the adaptive fairness filters in Eq. (9)-(10). Again, we take the user side as an example. Since the filter $\mathcal{F}$ is an MLP, its parameters are given by the set of weights and biases in each layer, i.e., $\Theta_{\mathcal{F}}^{\mathcal{U}} = \{\mathbf{W}^{\mathcal{U},\ell}, \mathbf{b}^{\mathcal{U},\ell}\}_{\ell=1,2,\ldots}$ where ℓ is the layer number in the MLP. Subsequently, scaling and shifting operations are performed to adapt the above shared parameters into user-specific parameters for u, as follows.

$$\mathbf{W}_u^{\mathcal{U},\ell} = \mathbf{W}^{\mathcal{U},\ell} \alpha_u, \tag{13}$$

$$\mathbf{b}_u^{\mathcal{U},\ell} = \mathbf{b}^{\mathcal{U},\ell} + \beta_u. \tag{14}$$

Collectively, $\Theta_{\mathcal{F},u}^{\mathcal{U}} = \{\mathbf{W}_u^{\mathcal{U},\ell}, \mathbf{b}_u^{\mathcal{U},\ell}\}_{\ell=1,2,\ldots}$ denotes the set of parameters of the user fairness filter adapted specifically to u. Adapting the item parameters in the same manner, we obtain $\Theta_{\mathcal{F},i}^{\mathcal{I}}$ specific to item i. Hence, the fairness filters in Eq. (9)-(10) are reformulated as

$$\mathbf{x}'_u = \mathcal{F}(\mathbf{x}_u; \Theta_{\mathcal{F},u}^{\mathcal{U}}), \tag{15}$$

$$\mathbf{x}'_i = \mathcal{F}(\mathbf{x}_i; \Theta_{\mathcal{F},i}^{\mathcal{I}}), \tag{16}$$

which are adaptive to different input users and items.

Note that the user- (or item-) specific parameters are adapted from the shared parameters $\Theta_{\mathcal{F}}^{\mathcal{U}}$ (or $\Theta_{\mathcal{F}}^{\mathcal{I}}$). Here the shared parameters capture a common debiasing prior shared among all users (or items), while the FiLM layers further modulate the common prior to adapt to individual variations beyond their commonality. In contrast, training one separate model for every user (or item) could be more flexible, but it can easily overfit due to the blown-up parameter space, and cannot leverage the common prior among users (or items).

To sum up, $\Theta_{\mathcal{F}} = (\Theta_{\mathcal{F}}^{\mathcal{U}}, \Theta_\alpha^{\mathcal{U}}, \Theta_\beta^{\mathcal{U}}, \Theta_{\mathcal{F}}^{\mathcal{I}}, \Theta_\alpha^{\mathcal{I}}, \Theta_\beta^{\mathcal{I}})$ represents all learnable parameters of the adaptive fairness filters, including the common priors and FiLM parameters for both users and items.

4.3 Fairness Discriminators

The fairness filters above attempt to mitigate the popularity bias by debiasing user and item embeddings. To judge if the fairness filters are effective, we introduce a set of fairness discriminators to detect the presence of any residual popularity bias in the debiased embeddings. In particular, the filters and discriminators are trained adversarially. Intuitively, adversarial learning plays a minimax game between a generator and a discriminator [45]. In our case, the filters act as generators that generate fair samples, and the discriminators attempt to distill potential residual biases from the generated samples.

More concretely, we design *explicit* and *implicit* fairness discriminators. The explicit discriminators pursue fairness from a local perspective, dealing with each user or item directly. On the other hand, the implicit discriminator pursues fairness from a global perspective, dealing with the overall rating scale.

4.3.1 Explicit fairness discriminators. The general idea of explicit discriminators is to directly detect the popularity bias in each local user or item. We employ two explicit discriminators, one for the user side, and the other for the item side. Taking the user side as an example, consider a user u with debiased embedding $\mathbf{x}'_u$, which is generated by the user fairness filter. We deem that there is still residual bias in the debiased embedding $\mathbf{x}'_u$ if the user-side explicit discriminator is able to classify u into the correct user group based on $\mathbf{x}'_u$. Note that the ability to recover the user group is a natural indicator of the presence of popularity bias, as the goal of user-side fairness is to reduce differential treatment of user groups with varying popularity. The mechanism of item explicit discriminator is similar.

Formally, the explicit discriminators $\mathcal{D}^{\mathcal{U}} : \mathbb{R}^d \rightarrow \mathbb{R}^m$ for users and $\mathcal{D}^{\mathcal{I}} : \mathbb{R}^d \rightarrow \mathbb{R}^n$ for items are essentially multi-class classifiers, defined as follows.

$$\hat{\mathbf{g}}_u = \mathcal{D}^{\mathcal{U}}(\mathbf{x}'_u; \Theta^{\mathcal{U}}_{\mathcal{D}}), \tag{17}$$

$$\hat{\mathbf{g}}_i = \mathcal{D}^{\mathcal{I}}(\mathbf{x}'_i; \Theta^{\mathcal{I}}_{\mathcal{D}}), \tag{18}$$

where $\mathcal{D}^{\mathcal{U}}$ or $\mathcal{D}^{\mathcal{I}}$ is implemented as an MLP parameterized by $\Theta^{\mathcal{U}}_D$ for users or $\Theta^{\mathcal{I}}_D$ for items. Their output layers employ a softmax activation, such that $\hat{\mathbf{g}}_u \in \mathbb{R}^m$ and $\hat{\mathbf{g}}_i \in \mathbb{R}^n$ are the predicted group probabilities for user u and item i, respectively. More specifically, let $\hat{\mathbf{g}}_u[j]$ (or $\hat{\mathbf{g}}_i[j]$) denote the probability of user u (or item i) belongs to the j-th user group $G^{\mathcal{U}}_j$ (or item group $G^{\mathcal{I}}_j$). The goal of the discriminators is to predict the group correctly, which can be achieved by maximizing the following log-likelihood.

$$L_{\mathcal{D}}(\mathbf{g}_u, \hat{\mathbf{g}}_u) = \sum_{j=1}^{m} \mathbf{g}_u[j] \log \hat{\mathbf{g}}_u[j], \tag{19}$$

where $\mathbf{g}_u \in \mathbb{R}^m$ is the one-hot encoding of u's ground-truth group such that $\mathbf{g}_u[j] = 1$ iff u belongs to the j-th user group $G^{\mathcal{U}}_j$. The log-likelihood for item groups, $L_{\mathcal{D}}(\mathbf{g}_i, \hat{\mathbf{g}}_i)$, can be defined in the same way.

4.3.2 Implicit fairness discriminator. While the explicit discriminators target each user or item locally, the recommendation outcomes may still manifest unfairness globally. The key reason is that different users have different rating scales [54]. This would particularly impact item-side fairness in terms of exposure, as an item predicted with a high rating might still be ranked relatively low for a user who habitually gives high ratings to all observed items, thus not being recommended to the user (i.e., non-exposure). Conversely, an item predicted with a low rating might still be recommended to a user (i.e., exposure) who habitually gives low ratings to all observed items. That means, even though item ratings in different popularity groups have been homogenized by the fairness filters, the recommendation outcomes might still result in unfairness in item exposure to users of different rating scales.

To address this problem, we design the implicit discriminator. The key idea is to normalize users' predicted ratings toward a global scale, so that all users have similar rating scales on items, thereby reducing the popularity bias due to different scales. Given a reference rating scale $\mathbf{r}_0 \in \mathbb{R}^{|\mathcal{I}|}$ on items, fairness is achieved when the predicted rating scales of users become indistinguishable from the reference scale. In this context, the fairness filters and the base recommendation model jointly act as the generator, which aims to predict user ratings and subsequently generate a rating sample similar to the reference scale. At the same time, the implicit discriminator plays the role of adversary, to distinguish if a rating sample is drawn from a user's scale or the reference scale.

Specifically, the reference scale can be estimated from the training data, in which item i's reference rating, $\mathbf{r}_0[i]$, is defined as the average rating among users who rated i in training data. On the other hand, the rating scale of user u, $\mathbf{r}_u \in \mathbb{R}^{|\mathcal{I}|}$, consists of predicted item ratings from our model, i.e., $\mathbf{r}_u[i] = \hat{y}_{u,i}$, the predicted rating of user u on item i. The implicit discriminator Δ is essentially a binary classifier, trying to tell if a rating sample is drawn from the reference

scale or the user's scale. Let ϕ be a rating sample consisting of K ratings uniformly sampled from either the reference scale $\mathbf{r}_0$ or user scale $\mathbf{r}_u$ for some user u. Then,

$$\hat{\delta}_\phi = \Delta(\phi; \Theta_\Delta), \tag{20}$$

where Δ is implemented as an MLP parameterized by Θ_Δ, and $\hat{\delta}_\phi$ is the predicted probability of ϕ sampled from the reference scale $\mathbf{r}_0$. As an MLP can only take in fixed-length input, the sample size K is fixed as 128 in our experiments.

The goal of the implicit discriminator is to tell apart user and reference scales correctly, which can be achieved by maximizing the log-likelihood below.

$$L_\Delta(\delta_\phi, \hat{\delta}_\phi) = \delta_\phi \log \hat{\delta}_\phi + (1 - \delta_\phi) \log(1 - \hat{\delta}_\phi), \tag{21}$$

where $\delta_\phi = 1$ if ϕ is sampled from $\mathbf{r}_0$, or 0 otherwise.

Collectively, the implicit and explicit discriminators have learnable parameters given by $\Theta_{\mathcal{D},\Delta} = (\Theta_{\mathcal{D}}^{\mathcal{U}}, \Theta_{\mathcal{D}}^{\mathcal{I}}, \Theta_\Delta)$.

4.4 Model Training

Our model optimizes not only the conventional recommendation loss, but also the generators' and discriminators' loss.

4.4.1 Base recommendation model. As shown in Fig. 5(d), the base model $\mathcal{R} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ predicts a rating score $\hat{y}_{u,i}$ that user u will rate on item i. It operates on the debiased user and item embeddings, $\mathbf{x}'_u$ and $\mathbf{x}'_i$, i.e.,

$$\hat{y}_{u,i} = \mathcal{R}(\mathbf{x}'_u, \mathbf{x}'_i; \Theta_\mathcal{R}), \tag{22}$$

where $\mathcal{R}$ is implemented as an MLP parameterized by $\Theta_\mathcal{R}$. Similar to the filters, the base recommendation can also benefit from an adaptive model, since user-item preferences also vary among users or items. Hence, we can also employ a FiLM layer in the base recommendation model, by scaling and shifting the MLP parameters $\Theta_\mathcal{R}$ of the recommendation model $\mathcal{R}$. The base recommendation model is trained by minimizing the mean squared error w.r.t. the ground-truth $y_{u,i}$, i.e.,

$$L_\mathcal{R}(y_{u,i}, \hat{y}_{u,i}) = (y_{u,i} - \hat{y}_{u,i})^2. \tag{23}$$

4.4.2 Adversarial training. Finally, we train the fairness filters, discriminators and base model jointly. As in a typical adversarial training, we alternate the optimization between the discriminators and generators.

For the discriminators, we attempt to beat the generators by (1) recovering user and item groups with the explicit discriminators; (2) distinguishing user and reference rating scales with the implicit discriminators, as follows.

$$\max_{\Theta_{\mathcal{D},\Delta}} \overbrace{\mathbb{E}_{u\sim\mathcal{U}} L_\mathcal{D}(\mathbf{g}_u, \hat{\mathbf{g}}_u) + \mathbb{E}_{i\sim\mathcal{I}} L_\mathcal{D}(\mathbf{g}_i, \hat{\mathbf{g}}_i)}^{\text{explicit discriminators}} + \overbrace{\mathbb{E}_{u\sim\mathcal{U},\phi\sim\mathbf{r}_u} L_\Delta(0, \hat{\delta}_\phi) + \mathbb{E}_{\phi\sim\mathbf{r}_0} L_\Delta(1, \hat{\delta}_\phi)}^{\text{implicit discriminator}}. \tag{24}$$

For the generators, the goal of the filters and base model is three-fold: (1) fooling the explicit discriminators so that the debiased user/item embeddings carry no group information and any associated popularity signals; (2) fooling the implicit discriminator so that the predicted user rating scales are indistinguishable from the reference scale; (3) the debiased embeddings still enable the base model to learn user-item preferences accurately, as follows.

$$\min_{\Theta_\mathcal{R},\Theta_\mathcal{F}} \mathbb{E}_{u\sim\mathcal{U},i\sim\mathcal{I}} \Big[\overbrace{\lambda_1 L_\mathcal{D}(\mathbf{g}_u, \hat{\mathbf{g}}_u) + \lambda_2 L_\mathcal{D}(\mathbf{g}_i, \hat{\mathbf{g}}_i)}^{\text{generator for explicit discriminators}} + \overbrace{\lambda_3 \mathbb{E}_{\phi\sim\mathbf{r}_u} L_\Delta(0, \hat{\delta}_\phi)}^{\text{generator for implicit discriminator}} + \overbrace{L_\mathcal{R}(y_{u,i}, \hat{y}_{u,i})}^{\text{base model}} \Big], \tag{25}$$

Algorithm 1: Training pipeline of FAiR

Input: $\mathcal{U}$: user set, $\mathcal{I}$: item set, n_g: number of updates for base model and generators per iteration, n_d: number of updates for discriminators per iteration.
Output: Optimized parameters $\Theta_{\mathcal{R}}, \Theta_{\mathcal{F}}$ and $\Theta_{\mathcal{D},\Delta}$;

```
Initialize all parameters;
while not converged do
    /* train discriminators */
    for n = 0; n < n_d; n++ do
        Sample a batch of users {u} ~ U;
        Sample a batch of items {i} ~ I;
        Sample a batch of ratings {φ_u} ~ r_u, u ~ U;
        Sample a batch of ratings {φ_0} ~ r_0;
        Update Θ_{D,Δ} according to Eq. (24);
    end
    /* train base model & generators */
    for n = 0; n < n_g; n++ do
        Sample a batch of user-item pairs {u, i} ~ U × I;
        Sample ratings φ ~ r_u for each sampled user u;
        Update Θ_R, Θ_F according to Eq. (25);
    end
end
return updated parameters Θ_R, Θ_F and Θ_{D,Δ}.
```

where λ_1, λ_2 and λ_3 are hyper-parameters to control the trade-off of different goals.

During the training of FAiR, we first update the discriminators ($\Theta_{\mathcal{D},\Delta}$) by Eq. (24), while keeping the generators including base model and filters ($\Theta_{\mathcal{R}}, \Theta_{\mathcal{F}}$) fixed. Next, we update the generators by Eq. (25), while keeping the discriminators fixed. We alternate between the two steps repeatedly, until the convergence of all parameters. The pseudocode of the training pipeline is sketched in Algorithm 1.

5 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the fairness and effectiveness of the proposed model FAiR.[5] We also conduct ablation studies and other analyses to further investigate the proposed model.

5.1 Experimental Setup

5.1.1 Dataset. To comprehensively evaluate FAiR for its effectiveness and fairness in different scenarios of recommendation, we employ three datasets in various domain, namely, Amazon[6] for e-commerce recommendation, Twitter[7] for friends recommendation and MovieLens[8] for movie recommendation. Essentially, these datasets all boil down to the prediction of interaction between entities, e.g., user-product interaction on Amazon, user-movie interaction on MovieLens, and user-user interaction on Twitter. These datasets are widely used by previous recommendation models

[5] We have released the code at https://github.com/mediumboat/FAiR
[6] https://jmcauley.ucsd.edu/data/amazon/
[7] https://recsys-twitter.com
[8] http://files.grouplens.org/datasets/movielens/

Table 2. Statistics of the datasets. "Pairs" denotes the user-item interaction pairs; "%Users/%Items" is the proportion of total users/items in the group; "%Pairs" is the proportion of interaction pairs involving users or items in the group.

Dataset	#Users	#Items	#Pairs	User group 1		Item group 1	
				%Users	%Pairs	%Items	%Pairs
Amazon	768,438	166,048	1,297,156	5%	27%	5%	53%
Twitter	290,049	-	3,492,386	5%	46%	-	-
MovieLens	943	1,682	100,000	71%	74%	5%	26%

including fairness-based models. In particular, the Twitter dataset is from RecSys Challenge 2021[9] on recommendation fairness. To construct user groups, we split MovieLens users into two groups by gender. For Amazon and Twitter, we choose the top 5% most frequent users as group 1, and the remaining as group 2. To construct item groups, for all datasets we choose the top 5% most frequent items as group 1 and the remaining as group 2.

For all datasets, we use the five-scale rating data for training and prediction. For Amazon, we use the "Grocery and Gourmet Food" category. For Twitter, we take the average number of engagements (capped at 5) in each observed user-user interaction as the rating. As the original Twitter dataset contains about 41M users and 291M interactions, to focus on the study of recommendation fairness rather than computational efficiency, we use breadth-first search (BFS) to sample a subset of about 290K users that can be fit into our memory and trained within a reasonable time period. Specifically, we start with a random user, and expands to its neighbors; and for each neighbor, we expand to their respective neighbors. The expansion is repeated until we reach a pre-determined threshold on the number of users. For MovieLens, we use the 100K version, where the number of user-movie pairs is 100,000. Note that on Amazon/MovieLens, the products/movies are items; while there are no items in Twitter dataset, we can still evaluate item-side fairness by treating the recommended friends as items.

The statistics of the datasets are summarized in Table 2.

5.1.2 Baselines. We compare FAiR with various state-of-the-art models in three broad categories, as follows.

- **Conventional recommendation methods:**
  (1) **CFI** [21]: A matrix factorization-based method for collaborative filtering on implicit feedback, which can be applied to explicit feedback as well. CFI has two commonly used implementations as follows. We denote the implementation with alternating least squares (ALS) algorithm as CFI_{ALS} and the other implementation with Bayesian personalized ranking (BPR) algorithm as CFI_{BPR}.
  (2) **NCF** [20]: A popular neural network-based collaborative filtering algorithm.
  (3) **LightGCN** [19]: A powerful graph convolution network-based recommendation algorithm.
- **General fairness and diversity models:**
  (4) **CFC** [5]: An adversarial framework with compositional fairness constraints to generate fair graph embeddings for users with sensitive attributes.
  (5) **FULTR** [41]: An optimization-based ranking framework to learn a fair ranking policy for users.
  (6) **FR** (FairRec) [32]: A post-processing multi-phrase model for two-sided fairness (i.e., not end-to-end), aiming to achieve envy-freeness for the user side and maximin share for the item side, which is not specifically designed to deal with the popularity bias.

[9] http://www.recsyschallenge.com/2021/

(7) **FaX** [10]: A determinantal point process-based algorithm for diversified recommendation, which helps item-side fairness to some extent by diversifying the exposure of different item groups, but it does not strive for equal exposure.

- **Fairness models for popularity bias:**
  (8) **DPR** [54]: An adversarial Bayesian model for item-side fairness.
  (9) **ADB** [8]: A meta-learning based method that automatically mitigates multiple kinds of bias for users including the popularity bias.
  (10) **UOF** [24]: An user-oriented fairness recommendation model which solves the popularity bias issue from the user perspective.
  (11) **MACR** [46]: The model-agnostic counterfactual reasoning framework to reduce popularity bias for items.

We implement the proposed FAiR using Tensorflow 2.2 in Python 3.6. All experiments were conducted on a Linux workstation with a 6-core 3.6GHz CPU, 128GB DDR4 memory and two RTX 2080Ti GPUs. For the baseline CFI, we used a third-party implementation[10]. For NCF[11], CFC[12], LightGCN[13], DPR[14], FULTR[15], FaX[16], FR[17], ADB[18], UOF[19] and MACR[20], we used their respective authors' implementations.

5.1.3 Metrics. We perform top-10 recommendation for each user, and evaluate the effectiveness, user-side fairness and item-side fairness of the top-10 recommendation list.

For effectiveness, we evaluate its precision and recall, denoted as prec@10 and rec@10, respectively. For both effective metrics, higher values indicate more effective recommendations.

For user-side fairness, we employ UGF_M defined in Eq. (4). We adopt the same effectiveness metric for M as in the effectiveness evaluation, i.e., prec@10 and rec@10. We denote the two options as UGF_{prec} and UGF_{rec}. For item-side fairness, we use RSP and REO defined in Eq. (7) and (8), with $K = 10$ (i.e., top-10 recommendations), and denote the two metrics as RSP@10 and REO@10, respectively. For all fairness metrics, lower values indicate better fairness.

5.1.4 Data splitting, hyper-parameters and features. We split the user-item interaction pairs into training/validation/test sets with the ratio of 0.8/0.1/0.1. To construct the validation/test candidates for a user u, we first add all items in the validation/test set that have interacted with u as positive examples, and then sample 10 non-interacting items as negative examples.

We determine the hyper-parameter and settings by tuning our model on validation set and following the guidance from literature. For FAiR, the trade-off hyper-parameters λ_1, λ_2 and λ_3 are set based on the validation set. They are set to 0.7/0.1/0.3 for Amazon, 0.4/-/0.2 for Twitter (λ_2 is not applicable as there is no item), and 0.4/0.3/0.8 for MovieLens. (An sensitivity analysis of λ_1, λ_2 and λ_3 is also presented in Sect. 5.7.) We set the embedding dimensions d to 128 for all datasets. For training, we use the Adam optimizer, and set the learning rate to 1e-3 and the batch size to 1,600. In each

[10] https://implicit.readthedocs.io/en/latest/
[11] https://github.com/hexiangnan/neural_collaborative_filtering
[12] https://github.com/joeybose/Flexible-Fairness-Constraints
[13] https://github.com/kuandeng/LightGCN
[14] https://github.com/Zziwei/Item-Underrecommendation-Bias
[15] https://github.com/him229/fultr
[16] https://github.com/laming-chen/fast-map-dpp
[17] https://github.com/gourabkumarpatro/FairRec_www_2020
[18] https://github.com/DongHande/AutoDebias
[19] https://github.com/rutgerswiselab/user-fairness
[20] https://github.com/weitianxin/MACR

iteration of training, we set the number of updates on discriminators and generators, n_d and n_g, to 5 and 1, respectively. For baselines, we set their hyper-parameters based on the validation set, following guidance from the literature.

For all methods (including FAiR) requiring user/item features as input, we generate the input user embeddings ($\mathbf{x}_u$) and item embeddings ($\mathbf{x}_i$) for each dataset using a collaborative filtering model[21] implemented in TensorFlow 2. For the baseline method CFC that requires a graph as input, we construct a bipartite graph based on the user-item interaction pairs. For the baselines CFI and CFC, we predict the rating $\hat{y}_{u,i}$ with the inner product of the corresponding output user/item embeddings.

5.2 Performance Comparison

Table 3 shows the performance of all models on the three datasets, where each model is repeated for five runs and we report their mean ± std-dev. Note that, as ADB runs out of memory on Amazon and Twitter, we only report its performance on the MovieLens dataset. From the results we draw three main conclusions.

Firstly, we observe that FAiR performs consistently well in fairness for both users and items compared to other methods. Among the three datasets, FAiR beats the *best conventional baseline* in each of the four fairness metrics by up to 44.3%, 38.0%, 97.5% and 60.9%, respectively, while outperforming the *best fairness baseline* in each metric by up to 11.1%, 5.3%, 15.6% and 17.1%, respectively. The results can be attributed to two reasons. On one hand, in our model design, we integrate both user- and item-side fairness in an end-to-end framework. The end-to-end two-sided fairness learning can mitigate the popularity bias while better preserving the intrinsic relationships between users and items. Hence, FAiR generally outperforms multi-phrase models such as FR. Besides, other fairness-aware baselines only focus on either user-side fairness (e.g., CFC and UOF) or item-side fairness (e.g., FULTR and DPR), which means their fairness performance on the other side is limited. For instance, while DPR and FULTR have a slight edge over FAiR in the item-side metric REO on Amazon or RSP on Twitter, they trail behind FAiR significantly on the user side metrics UGF_{prec} and UGF_{rec}. On the other hand, we design the adaptive fairness filters that are able to adapt to each user and item and effectively mitigate the popularity bias of different extents.

Secondly, there is a clear trade-off between effectiveness and fairness. Conventional methods do not pay any attention to fairness; not surprisingly, they often achieve high effectiveness at the expense of very poor fairness relative to fairness models as discussed in the first observation. Conversely, fairness models achieve much better fairness by compromising effectiveness to some extent. In particular, in our model, the trade-off between accuracy and fairness can be adjusted by tuning the hyperparameters in Eq. 25 (also see Sect. 5.7 on the influence of hyperparameters). Thus, to make meaningful fairness comparisons to other fairness models, our general principle is to maintain a similar level of effectiveness as other fairness models, while the effectiveness of conventional models merely provides a reference line for all the fairness-aware models. Intuitively, the effectiveness of conventional methods represents an "empirical upper bound" for the fairness models. In short, the effectiveness of our model FAiR remains competitive among the fairness methods, especially considering that we only implement a simple MLP as the base recommendation model. That implies FAiR is able to obtain a better trade-off given its superior fairness metrics.

Finally, we also observe that general fairness learning methods which are not specially designed for popularity bias (i.e., CFC, FULTR, FR and FaX) also achieve better performance on at least one fairness metric toward popularity bias compared to conventional methods. This finding reveals that there exist some relationships between the objectives

[21] https://github.com/keras-team/keras-io/blob/master/examples/structured_data/collaborative_filtering_movielens.py

Table 3. Comparison between FAiR and baselines. Each result (mean ± std-dev) is obtained by averaging five runs. In each column, the best is bolded, and the next best is underlined.

	↑Prec@10	↑Rec@10	↓UGF_{prec}	↓UGF_{rec}	↓RSP@10	↓REO@10
	Amazon					
CFI_{ALS}	.013±.002	.885±.006	.068±.005	.461±.016	.099±.003	.020±.008
CFI_{BPR}	**.019**±.001	**.921**±.003	.069±.004	.470±.015	.107±.001	.036±.005
NCF	.017±.003	.907±.043	.069±.014	.460±.022	.109±.004	.023±.001
LightGCN	**.019**±.002	.906±.015	.061±.008	.451±.011	.091±.004	.026±.005
CFC	.013±.002	.880±.039	.036±.006	.301±.025	.089±.014	.032±.009
FULTR	.011±.002	.892±.035	.054±.013	.363±.015	.077±.008	**.010**±.002
FR	.012±.004	.872±.002	.061±.001	.459±.010	.104±.002	.030±.001
FaX	.009±.002	.875±.004	.052±.003	.359±.004	.079±.002	.016±.003
DPR	.013±.001	.901±.021	.158±.012	.338±.013	.064±.004	**.010**±.002
UOF	.010±.003	.884±.013	.041±.008	.316±.009	.096±.006	.021±.003
MACR	.011±.002	.896±.019	.163±.007	.342±.011	.071±.012	.015±.004
FAiR	.012±.003	.887±.028	**.034**±.011	**.285**±.023	**.054**±.016	.012±.005
	Twitter					
CFI_{ALS}	.047±.002	.668±.022	.554±.011	.352±.013	.437±.010	.160±.006
CFI_{BPR}	.079±.005	.737±.029	.595±.004	.363±.011	.441±.004	.168±.009
NCF	.064±.010	.710±.026	.564±.027	.349±.012	.436±.007	.164±.007
LightGCN	**.082**±.011	**.739**±.030	.522±.031	.330±.019	.448±.014	.165±.013
CFC	.057±.013	.704±.026	.451±.018	.314±.008	.053±.004	.133±.006
FULTR	.056±.010	.708±.019	.463±.012	.337±.011	.102±.008	.144±.008
FR	.033±.006	.515±.009	.603±.010	.350±.011	.424±.006	.151±.009
FaX	.049±.004	.698±.004	.461±.003	.325±.002	.133±.003	.149±.004
DPR	.065±.018	.712±.031	.459±.039	.310±.014	**.003**±.001	.098±.014
UOF	.054±.013	.698±.007	.455±.005	.318±.006	.379±.026	.161±.005
MACR	.045±.008	.616±.013	.471±.022	.329±.018	.080±.009	.142±.013
FAiR	.054±.016	.686±.027	**.443**±.012	**.308**±.014	.011±.005	**.087**±.011
	MovieLens					
CFI_{ALS}	.527±.005	.498±.005	.040±.005	.225±.002	.202±.015	.240±.024
CFI_{BPR}	.501±.014	.472±.013	.047±.006	.217±.005	.196±.038	.188±.048
NCF	.529±.011	.495±.011	.054±.004	.230±.006	.166±.046	.174±.063
LightGCN	**.536**±.006	**.501**±.002	.042±.002	.208±.006	.230±.013	.217±.014
CFC	.476±.011	.449±.001	.036±.003	.203±.003	.181±.029	.166±.040
FULTR	.474±.008	.445±.002	.039±.002	.208±.004	.145±.013	.157±.021
FR	.516±.003	.485±.002	.042±.009	.210±.004	.143±.006	.154±.010
FaX	.452±.009	.441±.003	.040±.004	.208±.003	.198±.011	.163±.014
DPR	.509±.006	.474±.006	.046±.013	.226±.005	.108±.001	.082±.009
ADB	.496±.012	.452±.007	.051±.007	.225±.006	.141±.010	.144±.007
UOF	.478±.005	.463±.009	.044±.008	.212±.004	.183±.011	.171±.013
MACR	.518±.015	.487±.010	.044±.008	.206±.015	.126±.012	.159±.008
FAiR	.527±.019	.497±.018	**.032**±.006	**.198**±.009	**.096**±.010	**.068**±.014

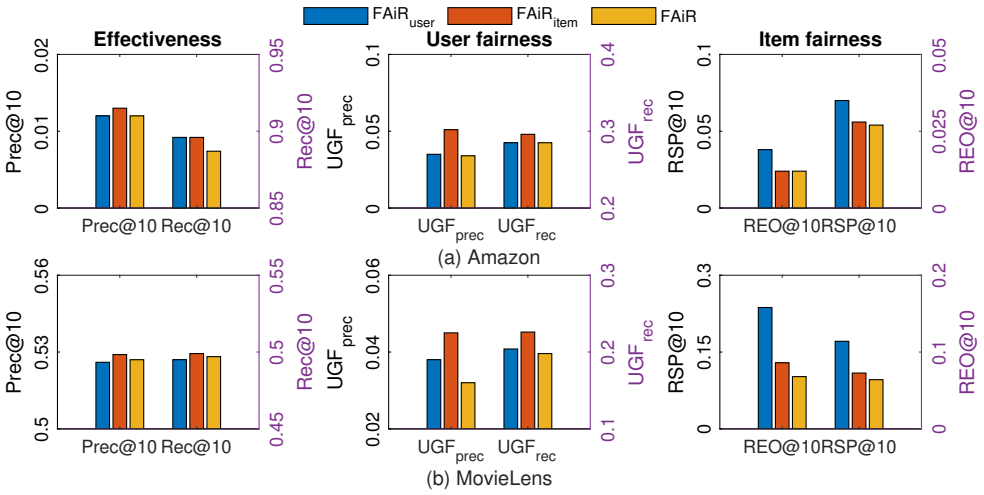


Fig. 6. Ablation study on the two-sided fairness.

of different debiasing methods toward different kinds of bias. Investigating such relationships could be a potential direction for future study.

5.3 Ablation Study

We conduct several ablations to examine the following three research questions (RQs).

5.3.1 RQ1: How does the two-sided fairness impact the performance? We derive two variants by implementing only the user-side fairness component or the item-side fairness component, and name them $\text{FAiR}_{\text{user}}$ and $\text{FAiR}_{\text{item}}$, respectively. We plot their performance in comparison to the full model FAiR in Fig. 6 (no results on Twitter as it only consists of users). The results show that $\text{FAiR}_{\text{user}}$ (or $\text{FAiR}_{\text{item}}$) fares well in user (or item) fairness metrics, but performs poorly in the fairness of the other side. It is also expected that, by filtering less information, the variants are able to produce slightly higher effectiveness metrics than the full model. In contrast, the full model is able to exploit the best of both worlds, with similar user-side fairness as $\text{FAiR}_{\text{user}}$, and simultaneously with similar item-side fairness as $\text{FAiR}_{\text{item}}$, with only marginal decrease in effectiveness. It shows that the full model integrates both sides well.

5.3.2 RQ2: How do the explicit and implicit discriminators contribute to FAiR model? We derive two variants of FAiR by implementing only the explicit or implicit discriminator(s), namely $\text{FAiR}_{\text{explicit}}$ and $\text{FAiR}_{\text{implicit}}$ respectively. We compare their performance with the full model FAiR in Fig. 7 on the three datasets. From the results we observe that both variants have slightly better effectiveness than FAiR, as less information is filtered by virtue of weaker adversarial training. However, FAiR significantly outperforms $\text{FAiR}_{\text{implicit}}$ in both user- and item-side fairness, implying that explicit discriminators are able to enhance the fairness of both sides. Comparing to $\text{FAiR}_{\text{explicit}}$, FAiR is much better in item-side fairness while achieving similar user-side fairness, implying that the implicit discriminator mainly improves item-side fairness. This is consistent with the motivation of implicit discriminator in Sect. 4.3, which is designed to mitigate the popularity bias w.r.t. item exposure across groups due to different user rating scales. In summary, both implicit and explicit discriminators are useful for their purposes.

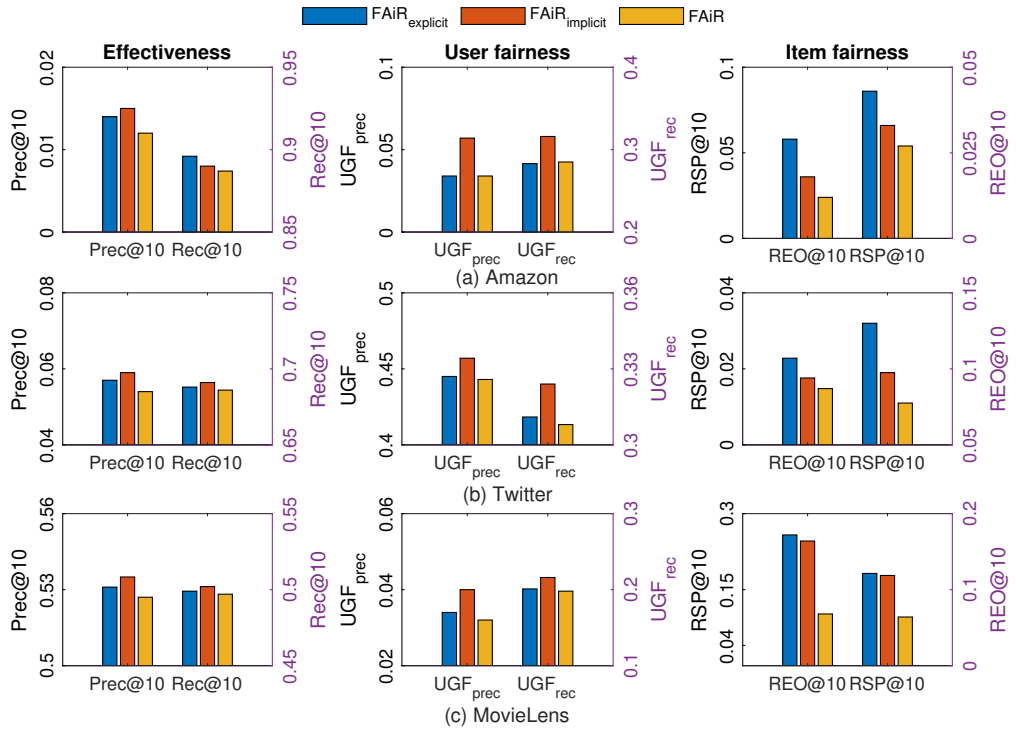


Fig. 7. Ablation study on the discriminators.

Table 4. Ablation study on FiLMs. Numbers reported are the absolute changes of FAiR when FiLMs are removed.

	↑Prec@10	↑Rec@10	↓UGF_{prec}	↓UGF_{rec}	↓RSP@10	↓REO@10
Amazon	0	-0.012	+0.013	+0.021	+0.013	+0.006
Twitter	-0.003	-0.039	+0.010	+0.017	+0.004	+0.004
MovieLens	-0.008	-0.010	+0.007	+0.004	+0.007	+0.013

5.3.3 RQ3: How important is it to make the filters adaptive? To answer this question, we remove all FiLM layers from our model FAiR, and report the absolute performance changes in Table 4. Results show that the removal of FiLMs consistently lead to worse effectiveness and fairness metrics. Therefore, it can be concluded that the adaptive filters are useful, which can enable FAiR to effectively adapt to different input users and items.

5.4 Adversarial Training

To validate the effectiveness of adversarial training in Algorithm 1, we plot the changes in the adversarial losses and evaluation metrics during training, on the MovieLens dataset, in Fig. 8.

We first examine Fig. 8(a), which shows the training loss curves of the discriminator (the negative of Eq. 24) and generator (Eq. 25). It can be observed that the generator loss is very large at the initial stage, as the popularity bias is significant in the original user and item embeddings (i.e., $\mathbf{x}_u$ and $\mathbf{x}_i$). After some updates, the generators (i.e., fairness

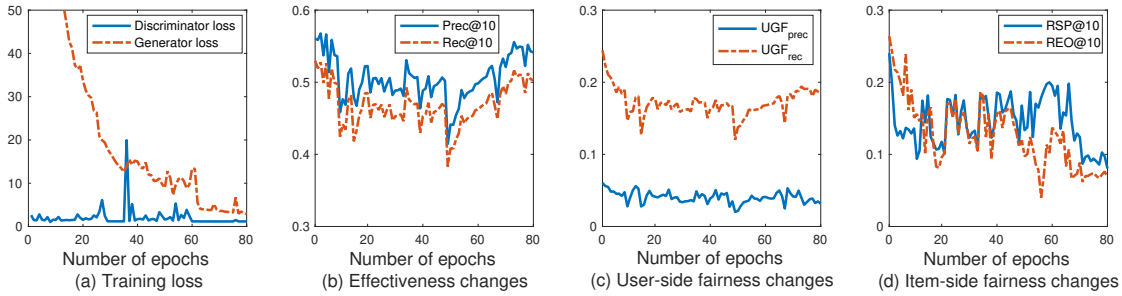


Fig. 8. Curves of training loss and evaluation metrics on MovieLens.

filters) begin to learn how to debias user/item embeddings and its loss decreases quickly. Then after about 30 epochs, it gradually becomes difficult for the discriminators to identify the bias. In short, the generators and discriminators play a minimax game to compete with each other, resulting in fluctuations in their losses while the overall loss is still trending down. Finally, after about 60 epochs both generators and discriminators become well trained and converge.

Next, we examine Fig. 8(b), which shows the changes in effectiveness metrics (Prec@10 and Rec@10) during training on validation data. At the beginning, we observe strong effectiveness metrics as we have initialized user/item features using effectiveness-oriented embeddings from a conventional collaborative filtering model. Subsequently, when the generator is trained to become better in mitigating bias as seen in (a), the effectiveness metrics begin to drop due to the trade-off between effectiveness and fairness. Finally, the effectiveness metrics increase again after about 50 epochs when the adversarial losses become near-optimal, which implies that the model can promptly refocus on the effectiveness of recommendation after the fairness filters have been well trained.

Lastly, we examine Fig. 8(c) and (d), which respectively show the changes in user-side and item-side fairness during training on validation data. In the initial stage, the fairness metrics for both sides have high values (i.e., worse fairness), revealing significant popularity bias which is also consistent with the high generator loss illustrated in (a). Later on, as the generator loss gradually decreases with better trained fairness filters, the values of the fairness metrics gradually decrease too (i.e., fairness improves). Finally, when the generator is sufficiently trained after about 60 epochs, the fairness metrics become more stable too with less fluctuations.

5.5 Visualization of Embeddings

To intuitively understand how the proposed fairness filters mitigate popularity bias, we visualize the user and item embeddings on the Amazon dataset using t-SNE [44]. Specifically, we show their distributions *before* and *after* the filters in Fig. 9. In Fig. 9(a), we observe that users in group 1 and group 2 are quite well separated in the embedding space before the fairness filter (i.e., $\mathbf{x}_u$), which means bias can exist leading to differential treatment to the two groups. In contrast, in Fig. 9(b), users in the two groups become mixed in the embedding space after the fairness filter (i.e., $\mathbf{x}'_u$), which are now harder to distinguish and more likely to get equitable outcomes. Similarly, the item embeddings before and after fairness filters (i.e., $\mathbf{x}_i$ and $\mathbf{x}'_i$) as shown in Fig. 9(c) and (d), respectively, demonstrate a similar pattern. Thus, the adaptive fairness filters can mitigate the popularity bias from both sides as expected.

Moreover, the visualizations also shed light on why there exists a trade-off between effectiveness and fairness. In essence, the filters have altered the original distributions of users and items in different groups, by forcing them to be

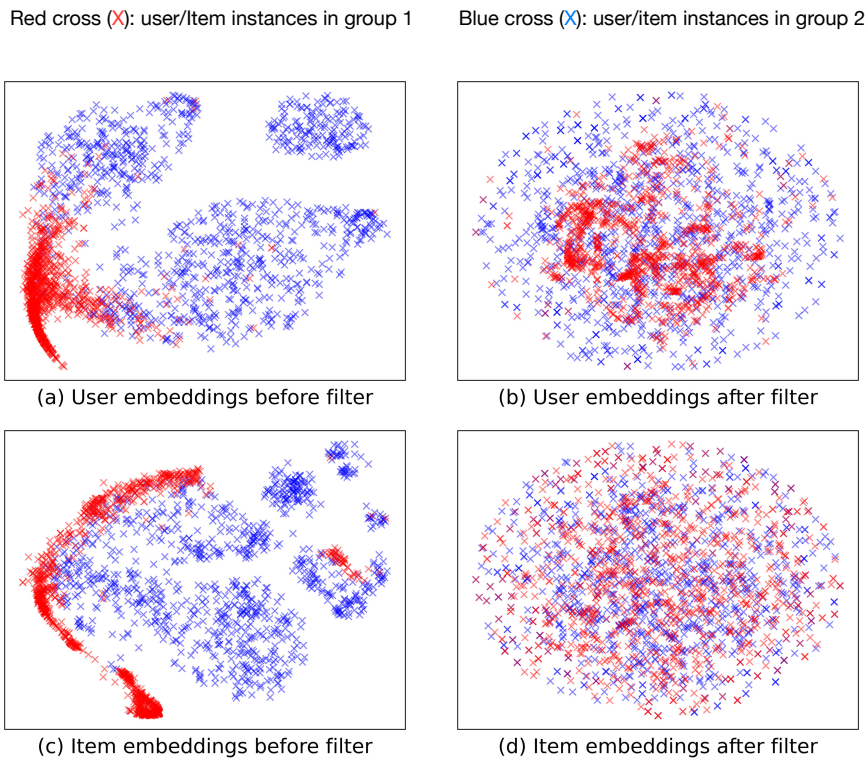


Fig. 9. The visualization of user and item embeddings before and after adaptive fairness filters on Amazon with t-SNE. The red and blue points indicate different user or item groups. To improve the readability of the figure, we uniformly randomly select 1000 instances for each group to show on each subplot after performing t-SNE.

similar across groups. The change in the distributions potentially harm the effectiveness of recommendation, while improving the fairness aspect.

5.6 Model Generalizability

To further evaluate the generalizability of the trained model, we re-evaluate its performance under alternative user/item groups, with different popularity across groups. Comparing to the group construction in the main experiment, here on the MovieLens dataset we construct three user (or item) groups by choosing the top 5%, the next 35% and the remaining 60% users (or items) according to their frequency of interaction. On the test data using the alternative groups, we compare the performance of FAiR with two other competitive baselines (CFC and DPR according to their performance in the main experiment) in Table 5. Results show that FAiR still maintains superior user- and item-side fairness when facing a different setting of user/item groups with different popularity.

Table 5. Generalizability evaluation on alternative popularity group construction. UGF_{prec} and UGF_{rec} are computed as the mean of corresponding UGF_M on all pairs of groups. The best result is bolded.

	↑Prec@10	↑Rec@10	↓UGF_{prec}	↓UGF_{rec}	↓RSP@10	↓REO@10
FAiR	**0.527**	**0.498**	**0.352**	0.131	**0.158**	**0.362**
CFC	0.476	0.449	0.392	**0.127**	0.178	0.402
DPR	0.509	0.474	0.388	0.169	0.159	0.391

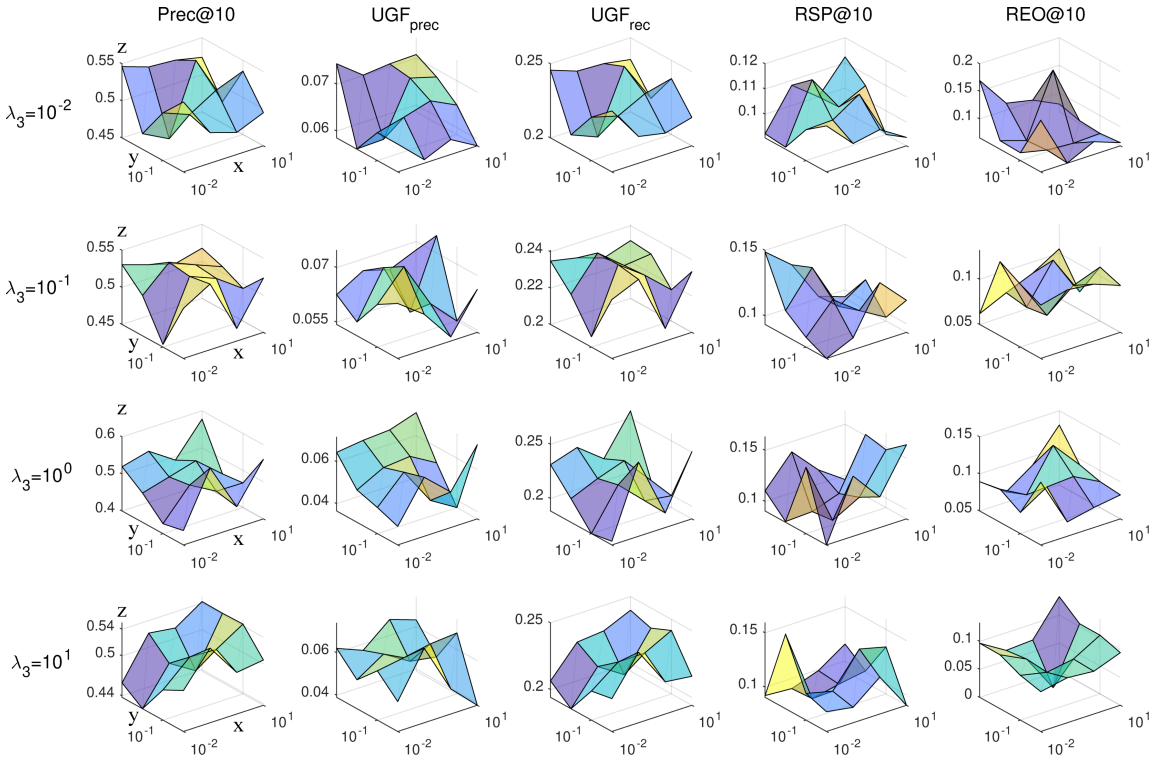


Fig. 10. Impact of λ_1, λ_2 and λ_3 on MovieLens. Each row shares the same λ_3 (as shown on the left), and each column shares the same metric (as shown at the top). In each plot, x-axis is λ_1, y-axis is λ_2 and z-axis is the corresponding metric (x, y, z as shown in the first column).

5.7 Parameter Sensitivity

We investigate the impact of trade-off hyper-parameters λ_1, λ_2 and λ_3 in Eq. (25) to understand how they jointly influence the overall effectiveness and fairness of the proposed FAiR. In Fig. 10, we evaluate the Prec@10 and fairness metrics w.r.t. to different sets of the hyper-parameters on the MovieLens data, by performing a grid search over $\{10^{-2}, \ldots, 10^{1}\}^3$.

We first examine the fairness metrics in the right four columns of Fig. 10. Generally the curve surfaces of the fairness metrics are convex, with their lowest points (i.e., best fairness) in the central regions. It means that extreme hyper-parameter values would hurt the fairness. On one hand, larger values would make the generators stronger than

the discriminators, making the latter unable to catch up to advance the minimax game. On the other hand, smaller values would make the generators weaker and become inadequate to tackle the bias.

Next, we examine the effectiveness metric Prec@10 in the first column of Fig. 10, where a trade-off with fairness is observed under varying hyper-parameter values. In general, for hyperparameter values that result in good fairness metrics (i.e., with low values), Prec@10 scores tend to be poor. The changes in Rec@10 follow a similar pattern.

Overall, to obtain optimal and stable performance in both effectiveness and fairness, we recommend tuning each hyper-parameter in the range $(0.1, 1)$, i.e., the central regions shown in the figure.

6 CONCLUSION

In this paper, we proposed an end-to-end fairness-centric model that adaptively mitigates the popularity bias in both users and items for recommendation (FAiR). Given the advantage of an end-to-end adversarial framework with both explicit and implicit discriminators, FAiR is able to integrate fairness for both users and items at a local as well as global level. Given the advantage of FiLM layers, FAiR is able to customize the model to the changing input, and adaptively mitigate the popularity bias in a manner specific to each user and item. Finally, we conducted comprehensive experiments on three real-world datasets with pressing fairness issues in different application domains. The results show that FAiR significantly outperforms state-of-the-art baselines in fairness metrics especially in a two-sided setting, whilst obtaining competitive effectiveness w.r.t. other fairness models. In future work, we intend to study the relationship between the popularity bias and other kinds of bias, and extend our model to mitigate those biases.

ACKNOWLEDGMENTS

This research is supported by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funds (Grant No. A20H6b0151).

REFERENCES

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling Popularity Bias in Learning-to-Rank Recommendation. In *RecSys '17*. 42–46.

[2] Himan Abdollahpouri and Masoud Mansoury. 2020. Multi-sided Exposure Bias in Recommendation. In *ACM KDD '20 Workshop on Industrial Recommendation Systems*. 7 pages.

[3] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2020. The connection between popularity bias, calibration, and fairness in recommendation. In *RecSys '20*. 726–731.

[4] Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H Chi, et al. 2019. Fairness in recommendation ranking through pairwise comparisons. In *KDD '19*. 2212–2220.

[5] Avishek Bose and William Hamilton. 2019. Compositional Fairness Constraints for Graph Embeddings. In *ICML '19*. 715–724.

[6] Rocío Cañamares and Pablo Castells. 2018. Should I Follow the Crowd? A Probabilistic Analysis of the Effectiveness of Popularity in Recommender Systems. In *SIGIR '18*. 415–424.

[7] Huiyuan Chen, Kaixiong Zhou, Kwei-Herng Lai, Xia Hu, Fei Wang, and Hao Yang. 2022. Adversarial Graph Perturbations for Recommendations at Scale. In *SIGIR '22*. 1854–1858.

[8] Jiawei Chen, Hande Dong, Yang Qiu, Xiangnan He, Xin Xin, Liang Chen, Guli Lin, and Keping Yang. 2021. AutoDebias: Learning to Debias for Recommendation. In *SIGIR '21*. 21–30.

[9] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and debias in recommender system: A survey and future directions. *arXiv preprint arXiv:2010.03240* (2020).

[10] Laming Chen, Guoxin Zhang, and Hanning Zhou. 2018. Fast greedy map inference for determinantal point process to improve recommendation diversity. In *NIPS '18*. 5627–5638.

[11] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-Balanced Loss Based on Effective Number of Samples. In *CVPR '19*. 9268–9277.

[12] Brian d'Alessandro, Cathy O'Neil, and T. LaGatta. 2017. Conscientious Classification: A Data Scientist's Guide to Discrimination-Aware Classification. *Big data* 52 (2017), 120–134.

[13] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *ITCS '12*. 214–226.

[14] Chongming Gao, Wenqiang Lei, Jiawei Chen, Shiqi Wang, Xiangnan He, Shijun Li, Biao Li, Yuan Zhang, and Peng Jiang. 2022. CIRS: Bursting Filter Bubbles by Counterfactual Interactive Recommender System. *arXiv preprint arXiv:2204.01266* (2022).

[15] Zhaolin Gao, Tianshu Shen, Zheda Mai, Mohamed Reda Bouadjenek, Isaac Waller, Ashton Anderson, Ron Bodkin, and Scott Sanner. 2022. Mitigating the Filter Bubble While Maintaining Relevance: Targeted Diversification with VAE-Based Recommender Systems. In *SIGIR '22*. 2524–2531.

[16] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. In *NIPS '14*. 2672–2680.

[17] Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney, Samantha Hansen, Damien Tardieu, and Ben Carterette. 2019. Offline Evaluation to Make Decisions About PlaylistRecommendation Algorithms. In *WSDM '19*. 420–428.

[18] Moritz Hardt, Eric Price, Eric Price, and Nati Srebro. 2016. Equality of Opportunity in Supervised Learning. In *NIPS '16*. 3315–3323.

[19] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR '20*. 639–648.

[20] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW '17*. 173–182.

[21] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM '08*. 263–272.

[22] Faisal Kamiran and Toon Calders. 2012. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems* 33, 1 (2012), 1–33.

[23] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. 2019. Decoupling Representation and Classifier for Long-Tailed Recognition. In *ICLR '19*.

[24] Yunqi Li, Hanxiong Chen, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2021. User-Oriented Fairness in Recommendation. In *WWW '21*. 624–632.

[25] Lydia T. Liu, Sarah Dean, Esther Rolf, Max Simchowitz, and Moritz Hardt. 2018. Delayed Impact of Fair Machine Learning. In *ICML 18'*, Vol. 80. 3150–3158.

[26] Yong Liu, Lifan Zhao, Guimei Liu, Xinyan Lu, Peng Gao, Xiao-Li Li, and Zhihui Jin. 2018. Dynamic Bayesian Logistic Matrix Factorization for Recommendation with Implicit Feedback. In *IJCAI-18*. 3463–3469.

[27] Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard S. Zemel. 2016. The Variational Fair Autoencoder. In *ICLR '16*.

[28] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. 2020. Feedback Loop and Bias Amplification in Recommender Systems. In *CIKM '20*. 2145–2148.

[29] Elisa Mena-Maldonado, Rocío Cañamares, Pablo Castells, Yongli Ren, and Mark Sanderson. 2021. Popularity Bias in False-positive Metrics for Recommender Systems Evaluation. *ACM Transactions on Information Systems (TOIS)* 39, 3 (2021), 1–43.

[30] Aadi Swadipto Mondal, Rakesh Bal, Sayan Sinha, and Gourab K Patro. 2021. Two-Sided Fairness in Non-Personalised Recommendations (Student Abstract). In *AAAI '21*, Vol. 35. 15851–15852.

[31] MEJ Newman. 2005. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics* 46, 5 (2005), 323–351.

[32] Gourab K Patro, Arpita Biswas, Niloy Ganguly, Krishna P. Gummadi, and Abhijnan Chakraborty. 2020. FairRec: Two-Sided Fairness for Personalized Recommendations in Two-Sided Platforms. In *WWW '20*. 1194–1204.

[33] Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. 2009. Measuring discrimination in socially-sensitive decision records. In *Proceedings of the 2009 SIAM international conference on data mining*. 581–592.

[34] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. 2018. FiLM: Visual Reasoning with a General Conditioning Layer. In *AAAI '18*.

[35] G Polya. 1958. Puzzle-Math. *Science* 128, 3317 (1958), 195–195.

[36] Liang Qu, Huaisheng Zhu, Ruiqi Zheng, Yuhui Shi, and Hongzhi Yin. 2021. ImGAGN: Imbalanced Network Embedding via Generative Adversarial Graph Networks. In *KDD '21*. 1390–1398.

[37] Tahleen A Rahman, Bartlomiej Surma, Michael Backes, and Yang Zhang. 2019. Fairwalk: Towards Fair Graph Embedding. In *IJCAI '19*. 3289–3295.

[38] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased Recommender Learning from Missing-Not-At-Random Implicit Feedback. In *WSDM '20*. 501–509.

[39] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as Treatments: Debiasing Learning and Evaluation. In *ICML '16*. 1670–1679.

[40] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of exposure in rankings. In *KDD '18*. 2219–2228.

[41] Ashudeep Singh and Thorsten Joachims. 2019. Policy Learning for Fairness in Ranking. In *NIPS '19*, Vol. 32. 5426–5436.

[42] Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In *KDD '10*. 713–722.

[43] Harald Steck. 2011. Item Popularity and Recommendation Accuracy. In *RecSys '11*. 125–132.

[44] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[45] Kunfeng Wang, Chao Gou, Yanjie Duan, Yilun Lin, Xinhu Zheng, and Fei-Yue Wang. 2017. Generative adversarial networks: introduction and outlook. *IEEE/CAA Journal of Automatica Sinica* 4, 4 (2017), 588–598.

[46] Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. 2021. Model-Agnostic Counterfactual Reasoning for Eliminating Popularity Bias in Recommender System. In *KDD '21*. 1791–1800.

[47] Le Wu, Lei Chen, Pengyang Shao, Richang Hong, Xiting Wang, and Meng Wang. 2021. Learning Fair Representations for Recommendation: A Graph-Based Perspective. In *WWW '21*. 2198–2208.

[48] Himank Yadav, Zhengxiao Du, and Thorsten Joachims. 2021. Policy-Gradient Training of Fair and Unbiased Ranking Functions. In *SIGIR '21*. 1044–1053.

[49] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. Fa*ir: A fair top-k ranking algorithm. In *CIKM '17*. 1569–1578.

[50] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning Fair Representations. In *ICML '13*, Vol. 28. 325–333.

[51] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. 2018. Mitigating Unwanted Biases with Adversarial Learning. In *AIES '18*. 335–340.

[52] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Comput. Surv.* 52, 1 (2019), 1–38.

[53] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. 2021. Causal Intervention for Leveraging Popularity Bias in Recommendation. In *SIGIR '21*. 11–20.

[54] Ziwei Zhu, Jianling Wang, and James Caverlee. 2020. Measuring and mitigating item under-recommendation bias in personalized ranking systems. In *SIGIR '20*. 449–458.