User-centered Evaluation of Popularity Bias in Recommender Systems

Himan Abdollahpouri
Northwestern University
Evanston, USA
himan.abdollahpouri@northwestern.edu

Masoud Mansoury
Eindhoven University of Technology
Eindhoven, Netherlands
m.mansoury@tue.nl

Robin Burke
University of Colorado Boulder
Boulder, USA
robin.burke@colorado.edu

Bamshad Mobasher
DePaul University
Chicago, USA
mobasher@cs.depaul.edu

Edward Malthouse
Northwestern University
Evanston, USA
ecm@northwestern.edu

ABSTRACT

Recommendation and ranking systems are known to suffer from popularity bias; the tendency of the algorithm to favor a few popular items while under-representing the majority of other items. Prior research has examined various approaches for mitigating popularity bias and enhancing the recommendation of long-tail, less popular, items. The effectiveness of these approaches is often assessed using different metrics to evaluate the extent to which over-concentration on popular items is reduced. However, not much attention has been given to the user-centered evaluation of this bias; how different users with different levels of interest towards popular items are affected by such algorithms. In this paper, we show the limitations of the existing metrics to evaluate popularity bias mitigation when we want to assess these algorithms from the users' perspective and we propose a new metric that can address these limitations. In addition, we present an effective approach that mitigates popularity bias from the user-centered point of view. Finally, we investigate several state-of-the-art approaches proposed in recent years to mitigate popularity bias and evaluate their performances using the existing metrics and also from the users' perspective. Our experimental results using two publicly-available datasets show that existing popularity bias mitigation techniques ignore the users' tolerance towards popular items. Our proposed user-centered method can tackle popularity bias effectively for different users while also improving the existing metrics.

CCS CONCEPTS

• **Information systems** → *Recommender systems*; • **Human-centered computing** → *Interactive systems and tools.*

UMAP '21, June 21–25, 2021, Utrecht, Netherlands

ACM ISBN 978-1-4503-8366-0/21/06…$15.00
https://doi.org/10.1145/3450613.3456821

KEYWORDS

recommender systems, popularity bias, long-tail recommendation, calibration, fairness

ACM Reference Format:
Himan Abdollahpouri, Masoud Mansoury, Robin Burke, Bamshad Mobasher, and Edward Malthouse. 2021. User-centered Evaluation of Popularity Bias in Recommender Systems. In *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization (UMAP '21), June 21–25, 2021, Utrecht, Netherlands.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3450613.3456821

1 INTRODUCTION

Historically, recommendation algorithms were developed to maximize the accuracy of the delivered recommendations to the users. However, as other researchers have noted, there are other important characteristics of recommendations that must be considered, including diversity, serendipity, novelty [11, 15, 40], and fairness [44]. These characteristics can have enormous impacts on the utility of the recommendations.

In this paper, we focus on the problem of *popularity bias* [13], the tendency of recommender systems to favor a small set of popular items (items that receive a large number of interactions from the users) in their recommendations, even more than their popularity would warrant, and to, unfairly, disfavor items that lie outside of this set, even when these items are preferred by a significant number of interested users [19, 29, 36].

Recommending long-tail items (items with low popularity) is generally considered to be valuable to the users [8, 35], as these are items that users are less likely to know about. Brynjolfsson et al. showed that 30-40% of Amazon book sales are represented by titles that would not normally be found in brick-and-mortar stores at the time of their writing [10]. They pointed out that access to long-tail items is a strong driver for e-commerce growth: the consumer surplus created by providing access to these less-known book titles is estimated at more than seven to ten times the value consumers receive from access to lower prices online. Long-tail items are also important for generating a fuller understanding of users' preferences. Systems that use active learning to explore each user's profile will typically need to present more long-tail items because these are the ones that the user is less likely to have rated, and where the user's preferences are more likely to be diverse [28, 30]. Finally, long-tail recommendation can also be

understood as a social good. A market that suffers from popularity bias will lack opportunities to discover more obscure products and will be, by definition, dominated by a few large brands or well-known artists [12]. Such a market will be more homogeneous and offer fewer opportunities for innovation and creativity. For all the aforementioned reasons, it is crucial to make sure these long-tail, less popular, items are fairly recommended on a recommender system platform by developing algorithms that give more chance to these items.

The research on popularity bias and long-tail has mainly taken an item-centered perspective. In other words, the focus has been largely on how recommendation algorithms amplify the popularity of already popular items in the recommendations given to the users. This item-centered focus, however, ignores the fact that not every user is equally interested in popular or less popular content and the popularity bias may affect different users differently. For example, Abdollahpouri et al. [4] demonstrated that popularity bias can cause unfair treatment of different user groups based on how interested they are in popular movies. In particular, they defined three user groups *Blockbuster-focused*, *Diverse*, and *Niche-focused*, and showed that the impact of popularity bias on niche-focused users is much greater than the other groups. In their work, they defined *impact* as the extent to which different user groups with a varying degree of interest towards popular movies were exposed to movies with different levels of popularity. The same patterns were found later by Kowald et al. [23] on the music domain where users with a lesser interest in mainstream songs were impacted more severely by the popularity bias. These findings showed that popularity bias impacts different users differently and hence it is important to take users into account when mitigating this bias.

There have been numerous attempts for developing algorithms that can help to reduce the over-concentration on popular items. These approaches mainly aim at improving three aspects of the recommendations in order to mitigate this bias: 1) making sure less popular items are recommended [45], 2) increasing the number of unique recommended items, regardless of their popularity, across all users [6, 20], and 3) making sure different items are *fairly* and *evenly* exposed to different users [25, 42]. See section 3.1 for more details on three different metrics that measure each of these aspects. However, none of these aspects take the users' tolerance towards popularity into account.

In this paper, we investigate popularity bias in recommender systems from the users' perspective. In other words, we are interested in assessing the algorithm's impact on users when it comes to the popularity level (aka mainstreamness) of the recommended items. More specifically, we would like to determine the extent to which the popularity level of recommended items match users' past interest in popular items. In other words, our goal is to measure the degree to which item popularity is personalized for each user. We first show that existing metrics for evaluating the performance of popularity bias mitigation algorithms fail to capture this aspect of the recommendations. We then, propose a metric, *User Popularity Deviation (UPD)* (see section 3.1.2), to measure the impact of popularity bias on different users based on their interest in item popularity. In addition, we investigate several state-of-the-art methods for popularity bias mitigation proposed in recent years and see how they tackle this bias for different users with a varying degree of interest in popular items. Finally, for ensuring a fairer treatment of different users in terms of how they are impacted by popularity bias, we present an approach that calibrates the recommendation lists given to different users according to their tolerance towards item popularity. We show that this popularity calibration technique, not only does it mitigate the popularity bias based on the existing (i.e. item-centered) metrics, it also tackles this bias fairer from the users' perspective according to our measure of user-centered impact.

In summary, we make the following contributions:

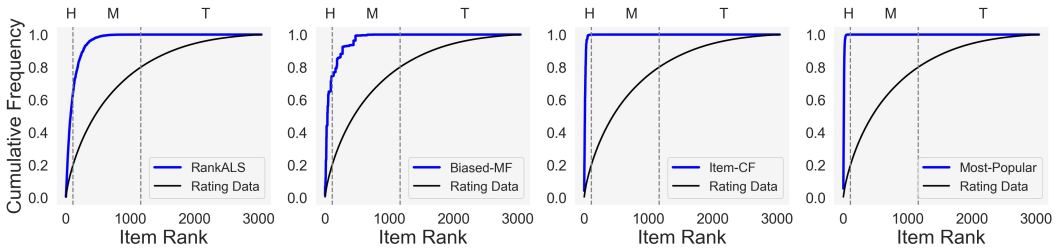
- We show the limitation of the existing, widely used, metrics for popularity bias mitigation when we take a user-centered point of view.
- We propose a new metric for measuring the impact of popularity bias on different users according to their interest in item popularity.
- We investigate several state-of-the-art recent approaches for mitigating popularity bias via both item-centered and user-centered evaluation and point out their limitation in bias mitigation from a user-centered point of view.
- We present a new approach to mitigate popularity bias that tackles this bias mainly from a user-centered point of view.

2 POPULARITY BIAS IN RECOMMENDER SYSTEMS

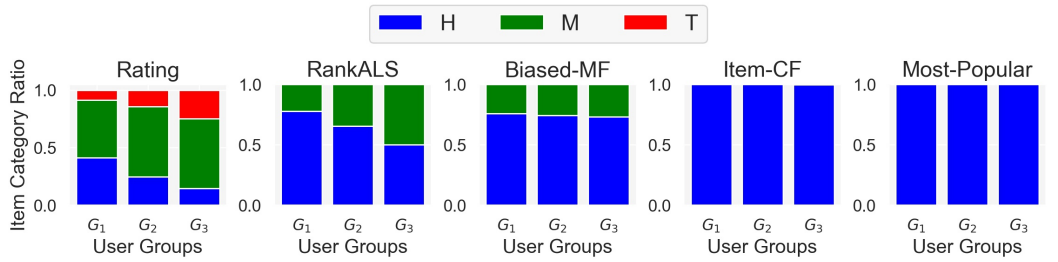
Recommendation algorithms are known to suffer from popularity bias where a few popular items dominate the recommendations given to all users. This leads to ignoring the majority of other less popular ones. In this section, we discuss the impact of popularity bias on different items based on their popularity and also on users based on their tolerance towards item popularity.

2.1 Impact on Items

The popularity bias can cause unfair treatment of less popular items. Consider the distributions shown in Figure 1(a). These four plots contrast original item popularity in the data and item popularity in the recommendations generated by four well-known recommendation algorithms (`RankALS` [39], *Biased Matrix Factorization (`Biased-MF`)* [22], *Item-based Collaborative Filtering (`Item-CF`)* [33], and a simple `Most-Popular` recommendation that recommends most popular items to every user) using the MovieLens 1M [17] dataset (see section 4.1 for more details on this dataset). The horizontal axis shows the rank of each item when sorted from most popular to least popular. The black curve shows the cumulative frequency of the ratings for different items at a given rank. As we can see, a few highly ranked items dominate the entire rating history. For instance, only 111 items (less than 3%) make up more than 20% of the ratings. In many consumer taste domains, where recommender systems are commonly deployed, the same highly-skewed distribution of user interest is seen. A music catalog might contain artists whose songs have been played millions of times (Beyoncé or Ed Sheeran) and others whose songs have a much smaller audience (Iranian musician Kayhan Kalhor, for example). These few popular items are referred to as the *Head* (shown by H in the plots) in the literature and make up roughly 20% of the interactions according to the Pareto Principle [31]. The rest of the items are usually referred to as *long-tail* and can be divided into two parts [12]: *Tail* items (T)



(a) Impact on items



(b) Impact on users

Figure 1: The impact of algorithmic popularity bias on items and users. H items are a small number of very popular items that take up around 20% of the entire ratings. T is the larger number of less popular items that collectively take up roughly 20% of the ratings, and M includes those items in between that receive around 60% of the ratings, collectively. Also, G_1 are users who are mainly interested in popular items. G_3 are users who high interest in niche and less popular items, and G_2 are users in between.

are the larger number of less popular items that collectively make up roughly 20% of the interactions, and *Mid* items (M) include a relatively large number of items in between that receive around 60% of the interactions, collectively. These item groups are shown on the top of each plot partitioning the items based on their popularity into the most popular items (H), items with medium popularity (M), and less popular items (T).

The blue curves in each plot show popularity bias at work across the four algorithms. In the most extreme cases, `Most-Popular` and `Item-CF`, the head of the distribution constituting less than 3% of the total items make up 100% of the recommendations given to the users. The other algorithms are only slightly better in this regard, with the H items making up more than 64% and 74% of the recommendations in `RankALS` and `Biased-MF`, respectively. These plots illustrate the impact of popularity bias on different items and how it creates a "rich get richer" dynamic.

2.2 Impact on Users

Although popular items are popular for a reason and many users are interested in them, this interest is not equal for all users. There are many users whose interests might be well outside of the popular items. As mentioned in section 1 and shown also by authors in [4, 23], popularity bias could also impact the users of the recommender system by distorting their recommendation lists and not serving them items with the right range of popularity. However, not much attention has been given to this aspect of the popularity bias. To illustrate this impact on users with a varying degree of interest in popular items, we define three equal-size groups of users based on their interest in popular items, similar to how they were defined in [4] (the ratio of popular items in their profile):

- **Blockbuster-focused Users (G_1)**: Users who have a high interest towards popular items.
- **Diverse Taste Users (G_2)**: Users with diverse interest towards popular and less popular items.
- **Niche-focused Users (G_3)**: Users who are mainly interested in less popular items.

For the rest of the paper, we will use G_1, G_2, and G_3 to refer to the corresponding user groups.

The first plot in Figure 1(b) shows the ratio of rated items from the three item groups H, M, and T in the profiles of different user groups in the MovieLens 1M dataset. The vertical axis shows the average proportion of different item groups in the profiles of users in different user groups. Note that all user groups, even G_1, have rated many items from the *Mid* (green) and *Tail* (red) parts of the distribution, and this makes sense: there are only a few really popular movies, and even the most blockbuster-focused viewer will eventually run out of them.

The vertical axis in other plots in Figure 1(b) shows the average proportion of different item groups in the recommended items delivered by different algorithms to the users in three user groups.

The difference with the original user profiles in rating data is stark, especially in the case of `Most-Popular` and `Item-CF`, where the users' profiles are rich in diverse item groups, the generated recommendations are much less so. *Tail* items do not appear at all. In `Most-Popular` and `Item-CF`, 100% of the recommendations are from the *Head* category, even for the users with the most niche-oriented profiles (i.e. G_3). Generally speaking, none of the user groups (even the "blockbuster-focused" G_1 users) are getting recommendations that are well-matched to their interests in terms of item popularity, but the "niche-focused" G_3 users are quite poorly served, getting a steady diet of popular movies in which they are less interested with none of their long-tail (M and T items) interests served. Thus, we see that popularity bias has a differential impact across the user base: all are affected but some much more severely than others. Our measure of the impact of popularity bias should reflect this.

3 POPULARITY BIAS MITIGATION

To mitigate popularity bias, as mentioned earlier, algorithms often try to improve three aspects of the recommendations: 1) making sure less popular items are recommended, 2) increasing the number of unique recommended items across all users, and 3) making sure different items are *fairly* and *evenly* exposed to different users. Several metrics have been used to evaluate how algorithms improve these aspects of recommendations.

3.1 Evaluation Metrics

We first describe the existing metrics to evaluate the performance of a popularity bias mitigation algorithm. Let L be the combined list of all recommendation lists given to different users (note that an item may appear multiple times in L, if it occurs in recommendation lists of more than one user). L_u is the recommended list of items for user u. Let I be the set of all items in the catalog and U be the set of all users.

3.1.1 Existing metrics: item-centered evaluation. The existing metrics for evaluating popularity bias mitigation are mainly item-centered and the most commonly used ones are as follows:

- **Average Recommendation Popularity (ARP)**: This measure from [45] and further used in [3] calculates the average popularity of the recommended items in each list. For any given recommended item in the list, it measures the average number of ratings for those items. More formally:

$$ARP = \frac{1}{|U|} \sum_{u \in U} \frac{\sum_{i \in L_u} \phi(i)}{|L_u|} \tag{1}$$

  where $\phi(i)$ is the number of times item i has been rated in the training set (i.e. the popularity of item i). Sometimes *ARP* for an algorithm can be low just because the algorithm has recommended a few extremely non-popular items to everyone. In other words, even if it has not increased the number of unique recommended items, it will have a good (low) *ARP*. For this reason, another metric is also measured to compensate for this drawback of *ARP* and it is called *Aggregate Diversity* (aka catalog coverage).
- **Aggregate Diversity (Agg-Div):** The ratio of unique recommended items across all users:

$$Agg\text{-}Div = \frac{|\bigcup_{u \in U} L_u|}{|I|} \tag{2}$$

  Higher values for this metric indicate that the recommendations "cover" more of the item catalog. This metric is widely used in many prior articles [6, 9, 20] to assess the effectiveness of the proposed algorithms to mitigate popularity bias. *Agg-Div* is measuring the extent to which an algorithm has increased the number of unique recommended items and hence reduced popularity bias. However, this metric has one limitation: it does not differentiate between whether an item is recommended only once or thousands of times. In other words, as long as an item is included in the recommendations, it will be counted by Equation 2. For this reason, another metric is often also looked at to ensure the recommendations are fairly distributed across all the items and that metric is *Gini index*.
- **Gini Index:** Measures the inequality across the frequency distribution of the recommended items. If some items are recommended frequently while other items are ignored, the Gini index will be high. Therefore, lower values for this metric are desirable.

$$Gini(L) = 1 - \frac{1}{|I| - 1} \sum_{k=1}^{|I|} (2k - |I| - 1) p(i_k|L) \tag{3}$$

  where $p(i|L)$ is the ratio of occurrence of item i in L.
  An algorithm that recommends each item the same number of times (uniform distribution) will have a Gini index of 0 and the one with extreme inequality will have a Gini of 1. Gini Index is also used in many prior works [14, 25, 42] to assess the (in)equality of the exposure of different items due to popularity bias.

3.1.2 Proposed metric: user-centered evaluation. All three *ARP*, *Agg-Div*, and *Gini* are assessing popularity bias from the items' perspective. In other words, their focus is largely on measuring how a certain algorithm has treated different items fairly in the recommendations. However, as we mentioned earlier, users are not equally impacted by this bias, and measuring the extent to which different users are affected can be of great importance. Figure 2 shows the recommendations given by two hypothetical algorithms *Algorithm 1* and *Algorithm 2* to three different users u_1, u_2, and u_3 belonging to different user groups G_1, G_2, and G_3, respectively. That means, user u_1 is mainly interested in popular items (H items), u_3 is mainly interested in non-popular items (T items), and u_2 is somewhere in between. Both algorithms have recommended 11 unique items across all three users and the frequency of their appearance is also identical. Therefore, both algorithms have exactly the same *ARP*, *Agg-Div*, and *Gini* and, hence, they will be perceived the same by the existing metrics (item-centered) in terms of popularity bias. However, from the users' perspective, the situation looks quite different. We can see that *Algorithm 1* has not done a good job in matching the recommendations in terms of popularity to the interest of different users: u_1 has received many T and M items and u_3 has received many H items. *Algorithm 2*, on the other hand, has delivered recommendations to different users that better

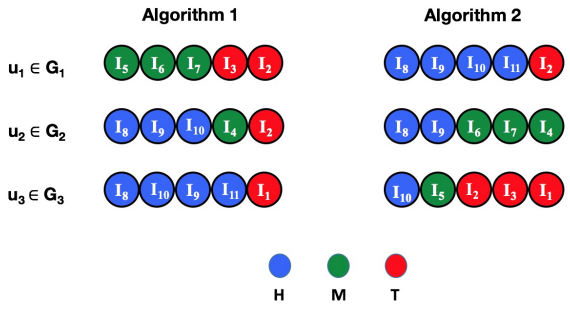


Figure 2: The recommendation lists of size 5 generated by two hypothetical algorithms for three users u_1, u_2, and u_3 belonging to three groups blockbuster-focused (G_1), diverse-taste (G_2), and niche-focused (G_3), respectively. The color of each recommended item indicates the group to which the item belongs.

match the tolerance and tendency of the users in terms of item popularity. This example highlights the importance of having a user-centered evaluation of popularity bias.

In this paper, we propose a new evaluation metric that captures this algorithmic impact on users in terms of the popularity of the recommended items. We intend to measure the impact of popularity bias on users both overall and on a group-level.

User Popularity Deviation (UPD): Users have different levels of tolerance towards different item popularity groups. The idea of measuring *UPD* is to find out how much the recommendations given to each user deviate from that of the user's historical data in terms of interest towards item popularity. For example, if a user has liked 30% H items, 20% M items, and 50% T items, a recommendation algorithm with zero *UPD* for that user should recommend a list of items that consists of the exact ratio of different item groups. That is, *UPD* compares the ratio of different item groups in the recommendations given to each user and their corresponding ratios in the historical data.

To do this comparison, we need to compute a discrete probability distribution P for each user u, reflecting the popularity of the items found in her profile ρ_u over each item group $c \in C$. For example, the distribution P for the aforementioned example would be $P = \{0.3, 0.2, 0.5\}$. We also need a corresponding distribution Q over the given recommendation list L_u indicating what item popularity groups are found among the recommended items. For example, if the recommended list contains 70% items from H, 30% items from M, and no item from T, then we will have $Q = \{0.7, 0.3, 0.0\}$. For measuring the interest of each user towards each item popularity group, we use Vargas et al.'s [41] measure of category propensity. Specifically, we calculate the propensity of each user u towards each item group $c \in C$ in her profile ($p(c|u)$) and the ratio of such item group in her recommendation list ($q(c|u)$) as follows:

$$p(c|u) = \frac{\sum_{i \in \rho_u} r(u,i)\mathbb{1}(i \in c)}{\sum_{c' \in C} \sum_{i \in \rho_u} r(u,i)\mathbb{1}(i \in c')}, \quad q(c|u) = \frac{\sum_{i \in L_u} \mathbb{1}(i \in c)}{\sum_{c' \in C} \sum_{i \in L_u} \mathbb{1}(i \in c')} \tag{4}$$

$\mathbb{1}(.)$ is the indicator function returning zero when its argument is False and 1 otherwise. $r(u,i)$ is the rating that user u has given to item i.

For any user group g, $UPD(g) = \frac{\sum_{u \in g} \Im(P(\rho_u), Q(L_u))}{|g|}$. The average *UPD* across different groups of users is:

$$UPD = \frac{\sum_{g \in G} UPD(g)}{|G|} \tag{5}$$

where $|G|$ is the number of user groups and $\Im$ is Jensen–Shannon divergence [24], which measures statistical distance between two probability distributions. Lower values for *UPD* are desirable as it shows the algorithm has less deviation in terms of matching the recommendations with users' tolerance towards items with different levels of popularity.

3.2 Algorithmic Solutions

There are several attempts to tackle popularity bias by either modifying the underlying recommendation algorithm or by doing a post-processing step on top of the output of any existing recommendation algorithm. In the former, the popularity of the items is taken into account in the rating prediction [2, 5, 38, 42] so the generated recommendations are less biased towards popular items. In contrast, in the latter, a post-processing step is added on top of the output of any existing recommendation algorithm where a larger output recommendation list is taken as input and is re-ordered to extract a shorter final list with improved long-tail properties. Most of the solutions for tackling popularity bias fall into the latter [3, 6, 7, 9].

In this paper, we investigated several papers on bias mitigation published in some top machine learning and recommender systems conferences and, for our analysis, implemented four of them. These four algorithms have all used the existing metrics that we mentioned in section 3.1.1 and hence their reported improvements are also based on those metrics. Therefore, they can be great choices to highlight the limitation we mentioned about exiting metrics and how they ignore user-centered evaluation. In addition, we also present a new, user-centered, approach and compare its performance with the previous techniques. These existing approaches are as follows:

- **ReGularization (`RG`)** [2]: This is a model-based algorithm to mitigate popularity bias that controls the ratio of popular and less popular items via a regularizer added to the objective function of the `RankALS` [39] algorithm. The algorithm penalizes lists that contain only one group of items and hence attempting to reduce the concentration on popular items.
- **Discrepancy Minimization (`DM`)** [9]: In this method, the goal is to improve the total number of unique recommended items, also referred to as aggregate diversity (see Equation 2) of recommendations using minimum-cost network flow method [16] to find recommendation sub-graphs that optimize diversity. Authors in this work define a target distribution of item exposure (i.e. the number of times each item should appear in the recommendations) as a constraint for their objective function. The goal is therefore to minimize the discrepancy of the recommendation frequency for each item and the target distribution.

- **FA*IR (FS) [46]:** This method can make a balance between the representation of two groups of items in recommendations: protected and unprotected. Here, the protected items are long-tail ($M \cup T$) items and the unprotected ones are head (H) items. The algorithm creates queues of protected and unprotected items and merges them using normalized scoring such that protected items get more exposure.
- **Personalized Long-tail Promotion (XQ) [3]:** In this method, inspired by the xQuAD algorithm for query result diversification [32], the objective for a final recommendation list is a balanced ratio of popular and less popular (long-tail) items. We specifically included XQ since, similar to our approach, it leverages user propensity towards popular items in its calculations and hence it can be categorized as a personalized long-tail promotion technique. However, its main focus is on a balanced distribution of popular versus non-popular items in the recommendation lists, and user propensity is not considered as a first priority but rather as a tie-breaker. Authors of this technique only defined two item categories: short-head (H) and long-tail ($M \cup T$).

Next, we propose an approach that aims at mitigating the popularity bias from a user-centered point of view.

3.2.1 Calibrated Popularity (CP). Given the proposed metric for a user-centered evaluation of popularity bias mitigation (*UPD*), we intend to propose an approach that addresses the popularity bias from the users' perspective. Our proposed technique, *Calibrated Popularity* (CP), is a re-ranking method. We are inspired by the calibrated recommendation algorithm proposed by Steck [37] that intended to match the distribution of the recommended items over different movie genres to the user's historical interactions. Steck argued that when a user has watched, say, 70 romance movies and 30 action movies, then it is reasonable to expect the personalized list of recommended movies to be comprised of about 70% romance and 30% action movies as well. We believe the same argument can be made for the users' preferences towards different item popularity groups. That is, if a user has liked 30 popular items, 20 items with medium popularity, and 50 non-popular items, then it is reasonable to expect the recommendation list to contain 30% popular items, 20% items with medium popularity, and 50% non-popular items. In a nutshell, Calibrated Popularity makes the recommendations more *personalized* in terms of *popularity*.

CP algorithm operates on an initial recommendation list $L_u^{'}$ of size m generated by a base recommender to produce a final recommendation list L_u of size n ($m \gg n$) for user u. Similar to [37], we measure distributional differences in the categories (groups) to which items belong $C = \{c_1, c_2, ..., c_k\}$ between the recommendation list for each user and her profile. For our purposes, these are the three H, M, and T item groups described above (i.e. $C = \{H, M, T\}$).

Similar to [37, 43], we use a weighted sum of relevance and calibration for creating our re-ranked recommendations. In order to determine the optimal set L_u^* from the m recommended items, we use maximum marginal relevance:

$$L_u^* = \underset{L_u, |L_u|=n}{\arg\max} \; (1-\lambda) \cdot Rel(L_u) - \lambda \cdot \mathfrak{J}(P, Q(L_u)) \qquad (6)$$

where λ is the weight controlling the relevance versus the popularity calibration and $Rel(L_u)$ is the sum of the predicted scores for items in L_u. Since smaller values for Jensen–shannon divergence, $\mathfrak{J}$, are desirable, we used its negative for our score calculation. As mentioned in section 3.1.2, P and Q are the distribution of popularity of the items found in user's profile and in her recommendation list, respectively.

All five approaches (RG, XQ, DM, FS, and CP) have a hyperparameter that controls the trade-off between relevance and a second criterion: diversification (incorporating diverse item popularity groups in the recommendations) in XQ, fairness (between popular and non-popular items) in RG and FS, aggregate diversity in DM, and user popularity calibration (or deviation) in CP. This hyperparameter is shown by λ in the result section (section 5).

4 EXPERIMENTAL METHODOLOGY

4.1 Data

We have used two publicly-available datasets for our experiments: the first one is a sample of the **Last.fm** (LFM-1b) dataset [34] used in [23]. The dataset contains user interactions with songs (and the corresponding albums). We used the same methodology in [23] to turn the interaction data into rating data using the frequency of the interactions with each item (more interactions with an item will result in a higher rating). In addition, we used albums as the items to reduce the size and sparsity of the item dimension, therefore the recommendation task is to recommend albums to users. We removed users with less than 20 ratings so only consider users for which we have enough data. The resulting dataset contains 274,707 ratings by 2,697 users to 6,006 albums. The second dataset is the **MovieLens** 1M dataset [17]. The total number of ratings in the MovieLens 1M data is 1,000,209 given by 6,040 users to 3,706 movies.

4.2 Experimental Settings

We used 80% of each dataset as our training set and the other 20% for the test. As with other re-ranking techniques, our method also needs a base algorithm to generate the initial list of recommendations for post-processing and any standard algorithm can be used. We used RankALS and we call it *Base* for the rest of the paper. Similar to [21], we set the size of the recommendations generated by the *Base* algorithm to 100 ($m = 100$), and the size of the final recommendation list is 10 ($n = 10$). We used *librec-auto* [27] for generating the recommendations by the *Base* (RankALS) algorithm.

5 RESULTS

In this section, we discuss the performance of different algorithms relative to the existing, item-centered, and our proposed, user-centered, evaluation metrics.

5.1 Item-centered Analysis

Figure 3 shows the performance of five different popularity bias mitigation algorithms, described in section 3.2, when we vary the weight for the diversity of the recommended items in terms of popularity (higher values for λ indicates more weight for bias mitigation).

On MovieLens, we can see that all algorithms have lost some degree of precision when we increase λ, which is consistent with prior work studying a trade-off between relevance and popularity bias mitigation [14, 18]. Some algorithms such as XQ and FS seem to perform better in terms of not losing too much precision while other methods such as DM and RG have lost a more substantial level of precision. CP also has lost some precision up to a certain point and it flattens out afterward. In regards to the existing metrics for popularity bias mitigation that we mentioned in section 3.1.1, *ARP*, *Agg-Div*, and *Gini*, we can see that DM clearly outperforms all other methods especially for larger values of λ (lower *ARP*, higher *Agg-Div*, and lower *Gini*). XQ also seems to perform relatively well in terms of *Agg-Div* compared to the other three methods followed by CP. However, although both DM and XQ have performed well in terms of *Agg-Div* and *Gini*, later on in Figure 4 we will discuss how they have achieved that and discuss how these two metrics can be misleading. XQ has a better aggregate diversity compared to CP, but its *ARP* is worse. This is primarily due to the fact that XQ has recommended more items compared to CP but those recommendations are still more concentrated on popular items. So far, in terms of existing metrics (*ARP*, *Agg-Div*, and *Gini*) it seems DM would be considered the best as it has achieved high *Agg-Div*, low *Gini*, and low *ARP*. However, as mentioned in section 3.1.2 by looking at Figure 2, these metrics hide the overall picture of how the popularity bias is mitigated from the users' perspective. The *UPD* results clearly show that FS and especially CP have performed much better in terms of user-centered popularity bias mitigation. For instance, at $\lambda = 0.5$ where both DM and CP have equal precision and equal *ARP*, the *UPD* for CP is twice as good as the one for DM (0.15 versus 0.3) which shows CP has matched the recommendations to the users' tolerance towards popularity in a much better way.

On Last.FM, the results are more promising for CP as it has outperformed the other approaches in almost every metric, including the existing item-centered ones and the new user-centered metric. That indicates the fact that it is possible to perform well on the item-centered metrics in terms of popularity bias mitigation when we try to tackle this bias from the users' perspective. It is not completely clear when this would happen and it could be dependent on the characteristics of the datasets as our preliminary analysis showed that the popularity bias in MovieLens is more extreme than that of Last.fm. However, further analysis is needed. Regarding the other existing algorithms, if we ignore a slightly worse precision drop for FS, it seems this algorithm is also performing well in the bias mitigation metrics which is consistent with what we saw on MovieLens. In both datasets, it is clear that RG does not perform well which confirms the results reported in [1] where the author reported that model-based popularity bias mitigation generally does not perform that well compared to the re-ranking ones.

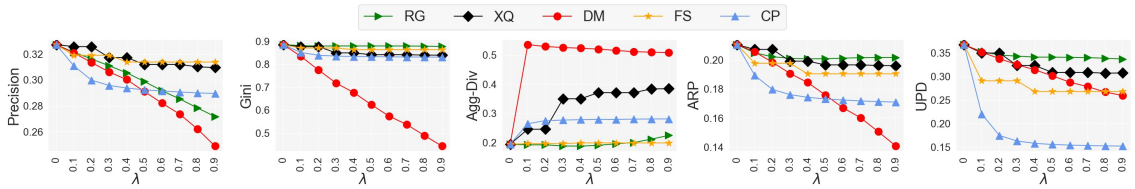
To establish a point of comparison across the algorithms, we picked the results for each algorithm at a particular λ that are close to each other in terms of precision, to be able to also analyze the performance of the algorithms in terms of popularity bias mitigation for each user group G_1, G_2, and G_3. We also reported the overall precision, *Agg-Div*, *Gini*, and *UPD* for each algorithm at this particular λ in Table 1 for easier comparison across algorithms. In this table, we can see that on MovieLens, as mentioned earlier, DM has performed really well on *Agg-Div* and *Gini* but not well on *UPD*. More or less is true about XQ. The reason can be explained using Figure 4 where we can clearly see how these two metrics can be misleading. This figure shows the exposure each algorithm has given to different items. In other words, the number of times each item is recommended across all users is calculated and plotted. The exposure values are sorted for better and easy-to-interpret visualization. Looking at the plot for Movielens in Figure 4 we can see a flat horizontal line very close to zero point (yet not zero) on the vertical axis for DM and, to some extent, for XQ. This explains the reason why DM and XQ achieved better results in terms of *Agg-Div* in Figure 3(a); these two algorithms have recommended a larger number of items only a few times since this metric does not care about the frequency of times each item is recommended. The very low *Gini* for DM can be also explained by the same plot; since many items are recommended with the same frequency (equal exposure), Equation 3 will be misleadingly low. The plot for Last.fm does not show this pattern and it is consistent with the results we saw in Figure 3(b) where DM and XQ did not have a misleadingly good *Agg-Div* and *Gini*. As mentioned earlier, this difference in performance on two datasets could be due to the characteristics of these datasets in terms of existing popularity bias and it needs further research (see section 6).

5.2 User-centered analysis

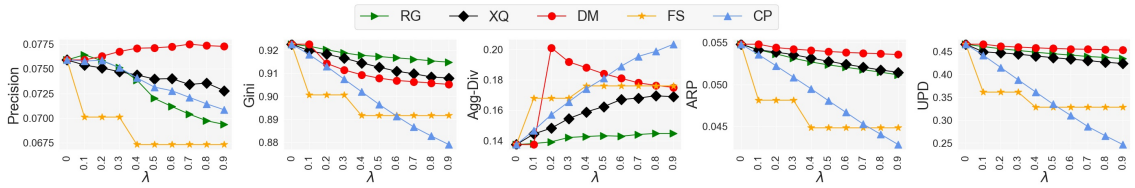
Figure 5 shows how each of the algorithms has performed in terms of mitigating popularity bias from the perspective of three user groups G_1, G_2, and G_3. These figures are similar to what we saw in Figure 1(b) where they compared the performance of four standard algorithms. The charts for the rating data and also for the base algorithm (RankALS) are also included for easier comparison.

Looking at charts for all the bias mitigation algorithms, except for RG, it is clear that they all have improved the recommendations in terms of a more balanced ratio of different item groups for three user groups. However, it is obvious that some have done a better job in doing so. For example, we can see that FS has certainly increased the exposure of M items for different user groups but not much has been done for the T items where the chart for this algorithm is still missing the T items for all user groups even for G_3 in which users have a significant interest towards these less popular items. Among XQ, DM, and CP, we can see that CP has matched these ratios much better and the recommendations given to different user groups are closer to what the users in these groups had indicated their interests in the rating data. DM and XQ have also included T items in the recommendations of different user groups, but a closer look reveals that this inclusion has been done not by reducing over-concentration of H (blue) items but rather by taking away from the M (green) items while the CP has balanced things smoother. On Last.fm, XQ and CP seem to have a better composition of different item groups but XQ has done so again by not removing bias from the H items but by shifting the recommendations from the M items towards T items. The extent to which each individual user group is affected by each of these algorithms is worth looking at.

Figure 6 shows the deviation of popularity (UPD) experienced by each user group ($UPD(g), g \in \{G_1, G_2, G_3\}$). This plot can further reveal how these different algorithms have performed for each of the user groups. First and foremost, for all user groups, CP has the



(a) MovieLens

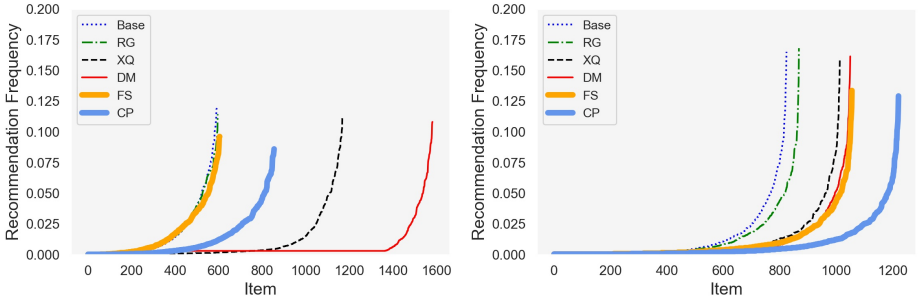


(b) Last.fm

Figure 3: The results of different popularity bias mitigation in terms of Precision, *Gini* Index, Average Recommendation Popularity (*ARP*), and User Popularity Deviation (*UPD*)

Table 1: The result of different popularity bias mitigation algorithms at one particular λ where algorithms have close precision. These values on MovieLens are 0.7, 0.9, 0.6, 0.9, 0.9 for RG, XQ, DM, FS, and CP, respectively. Similarly, on Last.fm these values are 0.9, 0.9, 0.9, 0.6, 0.9. Bold values are statistically significant compared to the second best value in each column with significance level of 0.05 (α=0.05). Up arrows indicate larger values are desirable while down arrows indicate smaller values are better.

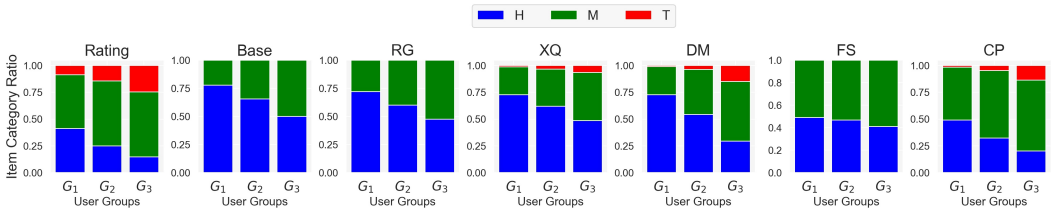
Algorithms	MovieLens					Last.MF				
	Precision↑	Agg-Div↑	*Gini*↓	*ARP*↓	*UPD*↓	Precision↑	Agg-Div↑	*Gini*↓	*ARP*↓	*UPD*↓
Base	0.327	0.194	0.885	0.21	0.368	0.076	0.137	0.922	0.055	0.466
RG	0.291	0.20	0.872	0.202	0.341	0.069	0.145	0.915	0.051	0.435
XQ	0.309	0.384	0.839	0.196	0.308	0.073	0.169	0.908	0.051	0.424
DM	0.300	**0.519**	**0.623**	0.167	0.302	0.077	0.175	0.905	0.054	0.453
FS	0.313	0.198	0.863	0.190	0.268	0.067	0.176	0.891	0.045	0.328
CP	0.289	0.281	0.831	0.171	**0.152**	0.071	**0.203**	**0.879**	**0.043**	**0.246**



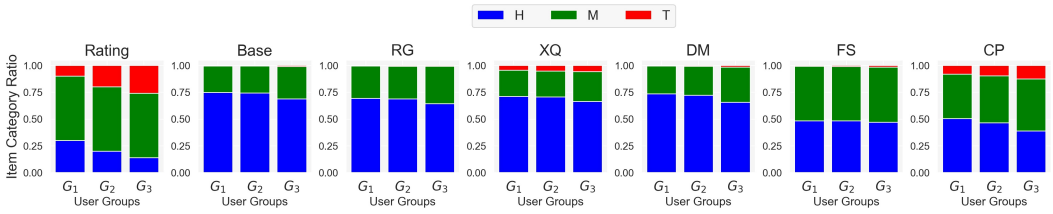
(a) MovieLens (b) Last.fm

Figure 4: The exposure frequency of different items recommended by different algorithms. The frequencies are sorted for easier comparison.

lowest deviation (*UPD*). Another important observation is that different user groups have clearly experienced different *UPD* values using some algorithms. For example, using **FS**, the deviation for G_1 is the lowest followed by G_2 and G_3 which has the highest *UPD*. That shows that this algorithm has done a better job in terms of



(a) MovieLens



(b) Last.fm

Figure 5: The impact of different popularity bias mitigation algorithms (RG, XQ, DM, FS, and CP) on different user groups

removing popularity bias for users who are more interested in popular items (G_1) than those who are not (G_3). CP, on the other hand, has yielded a consistent performance for all user groups giving them persistently low popularity deviation. Another interesting observation that Figure 6 facilitates to perceive is the fact that one algorithm can perform better than another algorithm for one user group but worse for another user group. For instance, on MovieLens, FS has a lower UPD for G_1 than DM but its UPD for G_3 is higher. In other words, if we care about mitigating bias from the blockbuster-focused users' perspective (G_1), FS outperforms DM but if we care more about Niche-oriented users (G_3) then DM performs better than FS.

To gain a more comprehensive picture of how different user groups are affected by popularity bias, looking at the overall UPD across all users may not be enough. For instance, looking at Table 1, we can see that XQ and DM have almost similar overall UPD (0.308 vs. 0.302) but, looking at Figure 6(a), we can see an interesting pattern: both algorithms have performed almost identical for user group G_2 yet XQ performs better for users in group G_1 and worse for users in G_3.

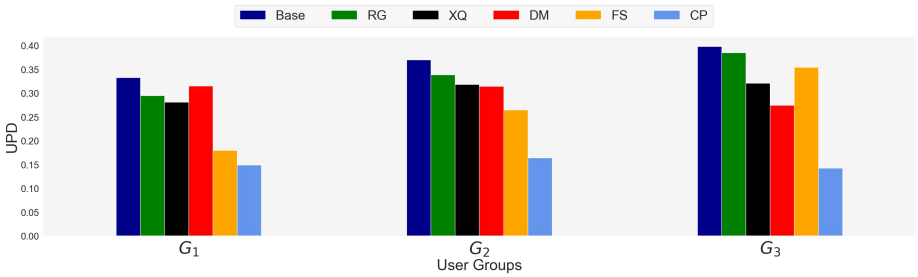
These are all observations that a typical, item-centered, evaluation procedure of popularity bias mitigation using existing metrics would have not been able to reveal and, hence, we believe, a user-centered evaluation can shed light on many important differences between the algorithms. Nevertheless, the user-centered evaluation proposed in this paper is not to replace the existing item-centered metrics but rather to complement them.

6 CONCLUSION AND FUTURE WORK

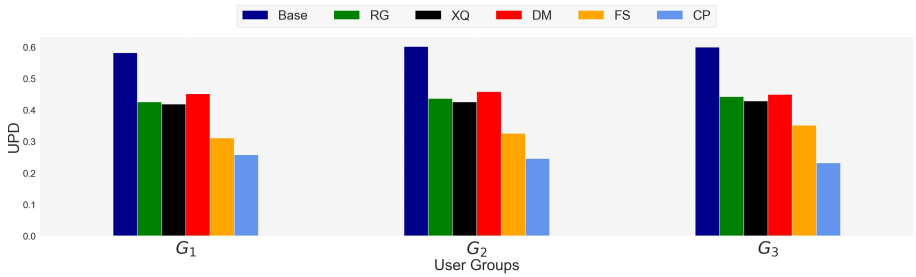
Popularity bias is a drawback of many recommendation algorithms that favor a few popular items while ignoring the majority of less popular ones. There have been numerous studies to investigate and also mitigate this bias. However, not much attention is given to the impact of this on users and the approaches that can be developed to tackle this bias from the users' perspective. In this paper, we studied the popularity bias from the users' perspective and showed that depending on users' tolerance towards popular items the impact they experience could be significantly different. We also highlighted some limitations of the commonly used metrics for measuring popularity bias mitigation so future usage of these metrics by other researchers can be done with more caution. In addition, we proposed a metric to measure popularity bias from the users' perspective and an approach to mitigate this bias from a user-centered point of view. We compared the performance of our proposed approach with several state-of-the-art approaches for mitigating popularity bias. As expected, the existing approaches, even though have achieved improvements on existing, item-centered, metrics, a further investigation revealed that they did not perform well from the users' perspective. Interestingly, we noticed that our approach which tackles popularity bias from the users' perspective was also able to improve the existing, item-centered, metrics to a great extent.

For future research, we intend to investigate the followings:

- We observed that some algorithms performed differently on two different datasets. Our preliminary analysis showed some differences in terms of the degree of popularity bias in the data for these two datasets and this could be one reason for that discrepancy in the performances. One future work could be to investigate how the characteristics of a dataset (e.g. degree of popularity bias, number of users and items, density, or other statistical features) can impact the degree to which a bias mitigation algorithm can tackle popularity bias.



(a) MovieLens



(b) Last.fm

Figure 6: User Popularity Deviation (*UPD*) for different user groups

- Another interesting future work is to study the impact of popularity bias on different user groups over time. Feedback loops are known to amplify popularity bias [26], and the extent to which this amplification differs for different user groups is an interesting question.

ACKNOWLEDGMENTS

Authors Abdollahpouri and Burke were supported in part by the National Science Foundation under Grant IIS-1911025.

REFERENCES

[1] Himan Abdollahpouri. 2020. *Popularity Bias in Recommendation: A Multi-stakeholder Perspective.* Ph.D. Dissertation. University of Colorado Boulder. https://arxiv.org/pdf/2008.08551.pdf

[2] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling Popularity Bias in Learning to Rank Recommendation. In *Proceedings of the 11th ACM conference on Recommender systems.* ACM, 42–46.

[3] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2019. Managing popularity bias in recommender systems with personalized re-ranking. In *The Thirty-Second International Flairs Conference.*

[4] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2019. The unfairness of popularity bias in recommendation. In *RecSys Workshop on Recommendation in Multistakeholder Environments (RMSE).*

[5] Panagiotis Adamopoulos and Alexander Tuzhilin. 2014. On over-specialization and concentration bias of recommendations: Probabilistic neighborhood selection in collaborative filtering systems. In *Proceedings of the 8th ACM Conference on Recommender systems.* 153–160.

[6] Gediminas Adomavicius and YoungOk Kwon. 2011. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Transactions on Knowledge and Data Engineering* 24, 5 (2011), 896–911.

[7] Gediminas Adomavicius and YoungOk Kwon. 2011. Maximizing aggregate recommendation diversity: A graph-theoretic approach. In *Proc. of the 1st International Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011).* Citeseer, 3–10.

[8] C. Anderson. 2008. *The long tail: Why the future of business is selling less of more.* Hyperion Books.

[9] Arda Antikacioglu and R. Ravi. 2017. Post processing recommender systems for diversity. In *In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 707–716.

[10] Erik Brynjolfsson, Yu Jeffrey Hu, and Michael D Smith. 2006. From niches to riches: Anatomy of the long tail. *Sloan Management Review* (2006), 67–71.

[11] Pablo Castells, Saúl Vargas, and Jun Wang. 2011. Novelty and diversity metrics for recommender systems: choice, discovery and relevance. In *Proceedings of International Workshop on Diversity in Document Retrieval (DDR).* ACM Press, 29–37.

[12] Òscar Celma and Pedro Cano. 2008. From hits to niches?: or how popular artists can bias music recommendation and discovery. In *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition.* ACM, 5.

[13] Giovanni Luca Ciampaglia, Azadeh Nematzadeh, Filippo Menczer, and Alessandro Flammini. 2018. How algorithmic popularity bias hinders or promotes quality. *Scientific reports* 8, 1 (2018), 1–7.

[14] Farzad Eskandanian and Bamshad Mobasher. 2020. Using Stable Matching to Optimize the Balance between Accuracy and Diversity in Recommendation. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization.* ACM, 71—-79.

[15] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. 2010. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems.* ACM, 257–260.

[16] Andrew V Goldberg. 1997. An efficient implementation of a scaling minimum-cost flow algorithm. *Journal of algorithms* 22, 1 (1997), 1–29.

[17] F Maxwell Harper and Joseph A Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5, 4 (2015), 19.

[18] Elvin Isufi, Matteo Pocchiari, and Alan Hanjalic. [n.d.]. Accuracy-diversity trade-off in recommender systems via graph convolutions. *Information Processing & Management* 58, 2 ([n. d.]), 102459.

[19] Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. 2015. What recommenders recommend: an analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction* 25, 5 (2015), 427–491.

[20] Mahmut Özge Karakaya and Tevfik Aytekin. 2018. Effective methods for increasing aggregate diversity in recommender systems. *knowledge and Information Systems* 56, 2 (2018), 355–372.

[21] Mesut Kaya and Derek Bridge. 2019. A comparison of calibrated and intent-aware recommendations. In *Proceedings of the 13th ACM Conference on Recommender*

Systems. 151–159.
[22] Y. Koren, R. Bell, and C. Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[23] Dominik Kowald, Markus Schedl, and Elisabeth Lex. 2020. The Unfairness of Popularity Bias in Music Recommendation: A Reproducibility Study. In *European Conference on Information Retrieval*. Springer, 35–42.
[24] Jianhua Lin. 1991. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information theory* 37, 1 (1991), 145–151.
[25] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. 2020. FairMatch: A Graph-based Approach for Improving Aggregate Diversity in Recommender Systems. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*. 154––162.
[26] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. 2020. Feedback loop and bias amplification in recommender systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2145–2148.
[27] Masoud Mansoury, Robin Burke, Aldo Ordonez-Gauger, and Xavier Sepulveda. 2018. Automating recommender systems experimentation with librec-auto. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 500–501.
[28] Tien T Nguyen, Pik-Mai Hui, F Maxwell Harper, Loren Terveen, and Joseph A Konstan. 2014. Exploring the filter bubble: the effect of using recommender systems on content diversity. In *Proceedings of the 23rd international conference on World wide web*. ACM, 677–686.
[29] Yoon-Joo Park and Alexander Tuzhilin. 2008. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM conference on Recommender systems*. 11–18.
[30] Paul Resnick, R Kelly Garrett, Travis Kriplean, Sean A Munson, and Natalie Jomini Stroud. 2013. Bursting your (filter) bubble: strategies for promoting diverse exposure. In *Proceedings of the 2013 conference on Computer supported cooperative work companion*. ACM, 95–100.
[31] Robert Sanders. 1987. The Pareto principle: its use and abuse. *Journal of Services Marketing* (1987).
[32] Rodrygo LT Santos, Craig Macdonald, and Iadh Ounis. 2010. Exploiting query reformulations for web search result diversification. In *Proceedings of the 19th international conference on World wide web*. ACM, 881–890.
[33] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. 2001. Item-Based Collaborative Filtering Recommendation Algorithms. In *10th International Conference on World Wide Web*. Hong Kong, China.
[34] Markus Schedl. 2016. The lfm-1b dataset for music retrieval and recommendation. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*. 103–110.
[35] Guy Shani and Asela Gunawardana. 2011. Evaluating recommendation systems. In *Recommender systems handbook*. Springer, 257–297.
[36] Harald Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*. 125–132.
[37] Harald Steck. 2018. Calibrated recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 154–162.
[38] Wenlong Sun, Sami Khenissi, Olfa Nasraoui, and Patrick Shafto. 2019. Debiasing the human-recommender system feedback loop in collaborative filtering. In *Companion Proceedings of The 2019 World Wide Web Conference*. 645–651.
[39] Gábor Takács and Domonkos Tikk. 2012. Alternating least squares for personalized ranking. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 83–90.
[40] Saúl Vargas and Pablo Castells. 2011. Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems* (Chicago, Illinois, USA) *(RecSys '11)*. ACM, New York, NY, USA, 109–116. https://doi.org/10.1145/2043932.2043955
[41] Saúl Vargas and Pablo Castells. 2013. Exploiting the diversity of user preferences for recommendation. In *Proceedings of the 10th conference on open research areas in information retrieval*. 129–136.
[42] Saúl Vargas and Pablo Castells. 2014. Improving sales diversity by recommending users to items. In *Proceedings of the 8th ACM Conference on Recommender systems*. 145–152.
[43] Jacek Wasilewski and Neil Hurley. 2018. Intent-aware Item-based Collaborative Filtering for Personalised Diversification. In *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization*. ACM, 81–89.
[44] Sirui Yao and Bert Huang. 2017. Beyond parity: Fairness objectives for collaborative filtering. In *Advances in Neural Information Processing Systems*. 2921–2930.
[45] Hongzhi Yin, Bin Cui, Jing Li, Junjie Yao, and Chen Chen. 2012. Challenging the long tail recommendation. *Proceedings of the VLDB Endowment* 5, 9 (2012), 896–907.
[46] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. Fa* ir: A fair top-k ranking algorithm. In *In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1569–1578.