Balanced Quality Score (*BQS*): Measuring Popularity Debiasing in Recommendation

ERICA COPPOLILLO[*][†], University of Calabria, Rende, Italy and ICAR-CNR, Rende, Italy
MARCO MINICI[*], University of Pisa, Pisa, Italy and ICAR-CNR, Rende, Italy
ETTORE RITACCO, University of Udine, Udine, Italy
LUCIANO CAROPRESE, University of Chieti-Pescara, Chieti, Italy
FRANCESCO SERGIO PISANI, ICAR-CNR, Cosenza, Italy
GIUSEPPE MANCO, ICAR-CNR, Cosenza, Italy

Popularity bias is the tendency of recommender systems to further suggest popular items while disregarding niche ones, hence giving no chance for items with low popularity to emerge. Although the literature is rich in debiasing techniques, it still lacks quality measures that effectively enable their analyses and comparisons.

In this paper, we first introduce a formal, data-driven, and parameter-free strategy for classifying items into low, medium, and high popularity categories. Then we introduce *BQS*, a quality measure that rewards the debiasing techniques that successfully push a recommender system to suggest niche items, without losing points in its predictive capability in terms of global accuracy.

We conduct tests of *BQS* on three distinct baseline collaborative filtering (CF) frameworks: one based on history-embedding and two on user/item-embedding modeling. These evaluations are performed on multiple benchmark datasets and against various state-of-the-art competitors, demonstrating the effectiveness of *BQS*.

CCS Concepts: • **General and reference** → **Metrics**; *Evaluation*; **Metrics**; • **Information systems** → **Recommender systems**; **Retrieval effectiveness**; **Information retrieval diversity**; **Collaborative filtering**.

1 INTRODUCTION

Recommender Systems based on collaborative filtering [6] are affected by a relevant problem: they are prone to suggest very popular items and neglect niche ones [9, 26, 46]. This phenomenon is known as *popularity bias* and it is a direct consequence of the underlying data distribution used for training the recommender. Within scenarios involving sparse interactions among large amounts of users and items, we typically observe a *long tail* distribution following the so-called *80-20 rule*, referring to the fact that 80% of users express preferences for only 20% of the available items. As a consequence, within the recommendation framework, the most popular items become more and more popular, while the items with low popularity do not get adequate exposure.

The literature offers numerous techniques to mitigate the popularity bias in recommendation. In particular, the most popular solutions embed procedures to enhance long-tail recommendations, while minimizing the impact on the math and implementation of their underlying algorithms.

[*] Both authors contributed equally to the paper.
[†] Corresponding author.

Authors' addresses: Erica Coppolillo, University of Calabria, Rende, Italy and ICAR-CNR, Rende, Italy, erica.coppolillo@unical.it; Marco Minici, University of Pisa, Pisa, Italy and ICAR-CNR, Rende, Italy, marco.minici@icar.cnr.it; Ettore Ritacco, University of Udine, Udine, Italy, ettore.ritacco@uniud.it; Luciano Caroprese, University of Chieti-Pescara, Chieti, Italy, luciano.caroprese@unich.it; Francesco Sergio Pisani, ICAR-CNR, Cosenza, Italy, francescosergio.pisani@icar.cnr.it; Giuseppe Manco, ICAR-CNR, Cosenza, Italy, giuseppe.manco@icar.cnr.it.

ACM 2157-6904/2024/3-ART
https://doi.org/10.1145/3650043

However, currently, there are no existing metrics that effectively capture both the mitigation ability and the capability to generate high-quality recommendation lists, thus limiting the in-depth analysis and comparison of debiasing techniques. In other words, there is no measure that adequately assesses both the exposure of items in the long tail and the predictive ability of a recommender system.

In this paper, we present the *Balanced Quality Score* (*BQS*) measure that fills this gap. The objective of *BQS* is to reward debiasing techniques that boost low-popular item exposure, without degrading global accuracy. We claim that, differently from standard metrics used for measuring popularity debiasing (that we reviewed in Section 3.3), *BQS* quantifies the underlying benefits of overexposing long-tail items, still taking into account the global accuracy.

The proposed metric is based on a partition of items into popularity classes. The standard approach in the literature is to rely on the 80%-20% method introduced by [1, 2]. However, this method does not necessarily fit the underlying data distribution. We overcome it by proposing a novel data-driven strategy that formally categorizes items as either low-, medium-, and high- popular, based on the intrinsic popularity distribution shape.

To prove the effectiveness of *BQS*, we apply several mitigation techniques (see Section 3.6) on three distinct baseline CF frameworks based on pairwise comparison and embedding modeling. Two rely on *user-/item-embedding* modeling, hence combining both users and items embeddings for producing the preferences; the third one relies on *history-embedding* modeling, computing preferences by projecting the user's history into the latent space. Paradigmatic of the first approach is *BPR* (Bayesian Personalized Ranking) [14, 17, 30, 31, 40], which translates users and items relatedness into geometric closeness within the latent space. Belonging to the family of *user-/item-embedding* approaches as well, we also consider *SimGCL* [44] which embeds users and items through graph convolutional layers and employs an additional self-supervised loss to improve the model robustness.

Representative of the second modeling approach is *RVAE* (Ranking Variational Autoencoder) [23, 25, 32, 34, 35], a ranking extension of *Mult-VAE* (Multinomial VAE) [25], which directly maps a user preference history into latent features that can be exploited to build a suitable ranking predictor.

Our contribution can be hence summarized as follows:

- We propose a novel technique to formally categorize items into low-, medium-, and high- popular, according to the popularity distribution shape.
- We study the effect of popularity bias for the two classes of models on an extensive set of diverse benchmark datasets.
- Since our objective is boosting low-popular item exposure without degrading global accuracy, we introduce a new metric in order to evaluate the improvement in low-popular items exposure, compared to the loss on global accuracy.
- We show that the proposed strategy is effective and competitive with respect to state-of-the-art approaches.

The rest of the paper is structured as follows. Section 2 analyses the current literature that studies the popularity bias phenomenon. The popularity-based categorization of the items and the details about *BQS* are discussed in Section 3. An experimental evaluation, supporting our claims, is shown in Section 4, while, in Section 5, we finally set some pointers for further research.

2 POPULARITY BIAS

Bias in computer systems can be defined as a *"systematic and unfair discrimination against certain individuals or groups of individuals in favor of others."* [16]. It is a phenomenon that deeply affects the recommendation algorithms in various forms since they are fed with data whose gathering process is observational rather than experimental [10]. One of the forms of bias is the so-called *popularity bias*, which implies an over-exposure of already high-popular items, neglecting niche ones. This pernicious phenomenon reduces not only the personalization (i.e., exacerbating user experience homogenization) but also the fairness and the variety of the suggested items.

Long-term consequences triggered by the feedback loop between the user, recommender, and data can be even more detrimental [27, 37]. In particular, as mentioned in [3, 8], unfair recommendations are concentrated on groups of users interested in long-tail and less popular items.

To mitigate the popularity bias, a conservative approach is to equip existing recommendation solutions with components whose goal is to amplify the exposure of rare item, trying to match users' interest. In particular, in this analysis we are interested in empowering pure collaborative filtering approaches, since they only need user-item interactions to produce suggestions. Hence, three possible strategies can be devised: *(i)* preprocessing the training dataset; *(ii)* altering the optimization/training process; *(iii)* calibrating the recommendation scores.

Pre-processing techniques. A first approach is to deliberately modify the input dataset to train a model favoring low-frequency items. Regarding popularity debiasing, several works [15, 20] propose to resample items inversely with the popularity to boost the presence of low-popular items in the top-k rankings. Boratto et al. [7] focus their approach on halving the negative examples with respect to popularity by sampling equally from popular and unpopular items.

Indeed, differently from us, all the aforementioned approaches adopt fixed oversampling strategies, regardless of the underlying item properties.

In-processing techniques. Another approach is to modify the learning phase of a recommender system by slightly altering its optimization process. Seminal work was proposed by [33], which adapts the IPS framework to preference modeling. Kamishima et al. [21] introduce a constraint aimed at minimizing the Normalized Mutual Information between the recommendation score and the popularity of the candidate item. Boratto et al. [7] also propose a regularization penalty that correlates the prediction of an item to its popularity in an attempt to quantify how much the recommendation depends on the popularity, with the consequent objective to minimize such dependency. Abdollahpouri et al. [1] propose a regularization that fairly chooses between two sets of items: one containing the short-head items and the other containing the medium-tail items.

Chen et al. [11] implement a co-training disentangled domain adaptation network, able to co-train both biased and unbiased models.

Ding et al. [13] propose a distillation framework that combines the losses of two models, one trained over the original (hence biased) dataset, the other one over a controlled debiased trial. Zhu et al. [47] propose a more sophisticated technique by reconsidering the Bayesian Personalized Ranking model [31] in an adversarial setting. They introduce a discriminator whose task is to derive the popularity group an item belongs to. The model needs to minimize the recommendation error while preventing the discriminator from correctly classifying the items.

Xv et al. [42] proposes a strategy that encourages all items embeddings to be orthogonal, thus disentangled and popularity neutral.

Causal analysis is a growing research line that recently found room in debiasing recommender systems. For instance, Zheng et al. [45] distinguish between two factors that cause the user-item interaction: the user interest in an item and the user conformity (i.e., how much the user follows trends); while Wei et al. [41] define a framework that jointly models any neural recommender and the item and user biases.

Nevertheless, conversely to our conservative approaches, performing an in-processing debiasing necessarily affects the adaptability of the strategy, since it requires retouching the model or necessarily resorting to a specific models category.

Post-processing techniques Strategies based on postprocessing were proposed in [2, 4, 36]. The core idea is to calibrate the recommendation list in order to give priority to less popular objects or to detect a *miscalibration* between groups of users.

Basically, these approaches try to modify the generated recommendation list to boost the exposure of low-popular items. For instance, Abdollahpouri et al. [2] define a gain function that alters the recommendation list by searching for other (not recommended) items within the catalog that could still match the user's taste. Xv et al. [42] propose to isolate one direction of the items embeddings to be popularity biased during training and neutralize it in a second phase.

The risk of this kind of strategy is to reduce the predictive capability of the chosen recommendation algorithm, thus downgrading the global performance.

3 CONTRIBUTION

Various metrics and strategies have been proposed to assess the performance of a recommendation model, such as Hit-Rate, NDCG (Normalized Discounted Cumulative Gain), and Precision. However, besides estimating how well the model predicts relevant content for users, in our opinion other factors should be taken into account.

One of them is the capability of the recommendation algorithm to surprise the user, by suggesting non-trivial items, that otherwise would not be able to reach them. By incorporating unexpected recommendations, the algorithm can improve user experience, broadening the user's horizons, exposing them to diverse options, and potentially introducing them to items they may have overlooked. This element of novelty and serendipity adds an extra layer of value to the recommendation process, going beyond the mere predictive accuracy of the system.

Another factor we want to consider is the fairness in offering equity (not equality) in exposure for the items (and their producers). Equality means that the recommender system treats every item the same, irrespective of their status or context. Although this may seem fair, items, supported by higher visibility or better advertising, are more likely to reach the user. Equity means that, in some circumstances, items need to be treated differently in order to provide meaningful parity of opportunity in reaching users, making the latter the real judges in the recommendation.

However, there is no standard solution on how to quantify serendipity and fairness in this context, since content providers are not able to distinguish between items that do not match user preferences from those which are not popular enough to be discovered. For this reason, serendipity and fairness are typically associated with the model capability of effectively exposing items belonging to the long-tail. Thus given, it's clear that the evaluation of a recommendation system is influenced by two forces pushing in opposite directions: on the one hand, we desire to obtain suggestions that match the user preferences; on the other hand, there is the drive to look for content that can positively surprise users who otherwise would not have visibility of niche but interesting products. As shown in Section 2, current literature is rich in solutions that try to improve the quality of a recommender system in such a sense, and quantify the exposure of long-tail items by adopting standard metrics, such as *Average Recommendation Popularity* (*ARP*) [43], *Average Percentage of Long Tail items* (*APLT*) [1], and *Average Coverage of Long Tail items* (*ACLT*) [2]. Notably, such metrics focus on estimating the exposure of the long-tail items in the recommendation list, without providing further information about the quality of such suggestions.

In the following, we aim at addressing two fundamental challenges belonging to this context:

(1) There is a lack of consensus over the portion of long-tail items to bring out;
(2) The existing metrics are not *self-contained*, i.e., they are not able to express both the exposure of long-tail items and the recommendation quality.

To discuss about the challenges and the proposed solutions we are introducing the notation that will be exploited in the rest of the work.

3.1 Notation

Given $U = \{1, \ldots, M\}$ a set of M users and $I = \{1, \ldots, N\}$ a set of N items, let $\mathbf{X} \in \{0, 1\}^{M \times N}$ be a preference matrix, so that $x_{u,i} = 1$ whenever user $u \in U$ expressed a preference for item $i \in I$, and $x_{u,i} = 0$ otherwise (the item can be both unknown to or disliked by u). We denote by $\mathbf{x}_u$ the u-th row in $\mathbf{X}$ and by $I_u = \{i \in I \mid x_{u,i} = 1\}$ the set of $n_u = |I_u|$ items chosen by u. The preference matrix induces a natural ordering among items, where $i \succ_u j$ means that u prefers i to j, i.e., $x_{u,i} > x_{u,j}$ in the rating matrix: for each user u, we denote as $\mathcal{D}_u \subset \{(i,j) \mid i,j \in I; i >_u j\}$ their associated set of pairwise comparisons.

In addition, we define $\rho_i = \sum_{u \in U} x_{u,i}$ as the popularity (i.e., absolute frequency) of the item i, and $\boldsymbol{\rho} = \{\rho_{i_1}, \rho_{i_2}, \ldots, \rho_{i_N}\}$ as the popularity vector over the whole item catalog, where $i_k \in I$, $k \in \{1, \ldots, N\}$ and $\rho_{i_k} < \rho_{i_{k+1}}$.

3.2 Choosing from the Long-Tail

To address the first challenge, we need to understand the underlying nature and the structural properties of the data distribution that governs the recommendation. The preference matrix of a recommender system is actually an instantiation of Preference Networks, which have been extensively studied in the literature [29]. These networks are characterized by a bipartite graph that connects two sets of nodes: users and items. Consequently, the degree of an item i within the item-set represents the number of times users have interacted with it: its popularity ρ_i.

We can consider the Complementary Cumulative Degree Distribution (CCDD) which defines the probability $P(\rho_i \geq \rho)$ of a generic item i in the network having a degree greater than or equal to ρ. Notice that this probability can be equivalently expressed in terms of the frequency $f(\rho) = |\{i \in I | \rho_i \geq \rho\}|$, since $f(\rho) \approx P(\rho_i \geq \rho) \cdot N$. As stated in [12, 28, 29], CCDDs in preference networks typically exhibits shapes resembling power-law functions, like the ones shown in Figure 1, where the X-axis represents the item popularity ρ, while the Y-axis represents the complementary cumulative frequency of the items $f(\rho)$, both on a logarithmic scale. A perfectly power-law shaped function (Figure 1a) is extremely unlikely in real preference networks. Therefore, we will focus on concave distributions (Figure 1b).

The CCDD and $\boldsymbol{\rho}$ are strictly related since the former contains the same information of the latter but in an aggregate form. Specifically, the shape of $\boldsymbol{\rho}$ is a sequence of steps, where each step collects all the items that are characterized by the same popularity value. Conversely, the CCDD, through a suitable transformation[1], overlaps with $\boldsymbol{\rho}$, but aggregates its steps in a unique data point (Figure 2). Hence, we can state that the shape of $\boldsymbol{\rho}$ characterizes the preference networks.

The logarithmic values of $\boldsymbol{\rho}$ for concave CCDDs exhibit a shape that is similar to the one shown in Figure 3b, where we have evidence of an inflection point and two elbows. The first elbow is generated by the flat portion of the untransformed CCDD, whose ending negative slant is cause of the second elbow. Both the elbows can be exploited to define a data-driven strategy to formally categorize items into popularity classes, eliminating the reliance on biased approaches that employ fixed thresholds along the $\boldsymbol{\rho}$ distribution curve [1, 2]. [2] In fact, we can define two natural thresholds, namely τ_L and τ_H, by considering the popularity values of the elbows. These thresholds split I into three distinct classes, highlighting the different item exposure: $I_L = \{i \in I \mid \rho_i \leq \tau_L\}$ the set of items with a very low exposure, $I_M = \{i \in I \mid \tau_L < \rho_i \leq \tau_H\}$ the set of items with progressively increasing exposure, and $I_H = \{i \in I \mid \rho_i > \tau_H\}$ the set of items with an out-of-scale exposure. Without loss of generality, in the (rare) case of a perfect power-law-shaped distribution, $\boldsymbol{\rho}$ will show only one elbow, as shown in Figure 3a. Consequently, I will be split into two classes, namely low- and high-popular ones.

The set I_L contains the *low-popular items*, which we deem to possess the highest potential in terms of novelty and fairness, since they represent niche products or content that, on one hand, users would not naturally interact with, and, on the other hand, would not have been able to emerge during the recommendation process. Upon closer examination of the neighborhood around τ_L, we observe an abrupt shift in the growth pattern of the

[1] CCDD Transformation. Assume the CCDD function is encoded as $f(x_i)$ for $i \in \{1, \ldots, n\}$:

(1) Set $y_1 = 0$ and $y_i = y_{i-1} + f(x_{n-i+1}) - f(x_{n-i})$, for $2 \leq i \leq n$
(2) Define $\tilde{\rho}(y_i) = x_{n-i}$, for $1 \leq i \leq n$.

The resulting $\tilde{\rho}(y)$ represents the popularity of the y-th item in the popularity rank.

[2] These approaches poorly fit data properties, since they are the result of empirical analyses. Fixed thresholds may generate noisy popularity classes, where either long-tail and well-exposed items, or regularly-exposed and popular items are mixed.

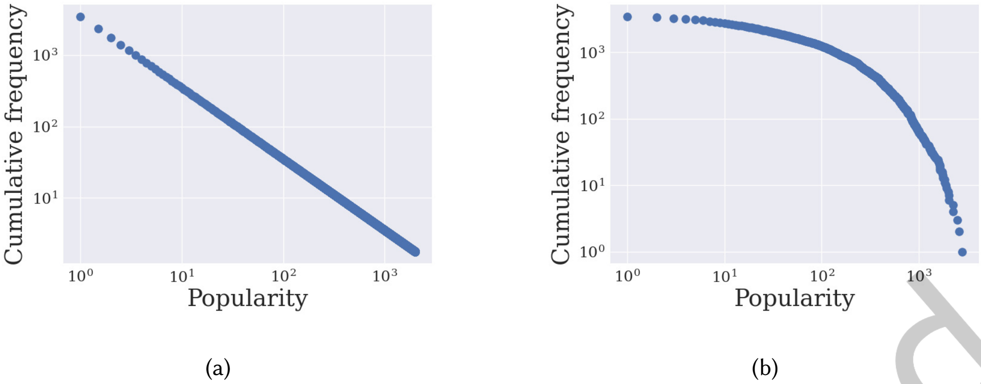
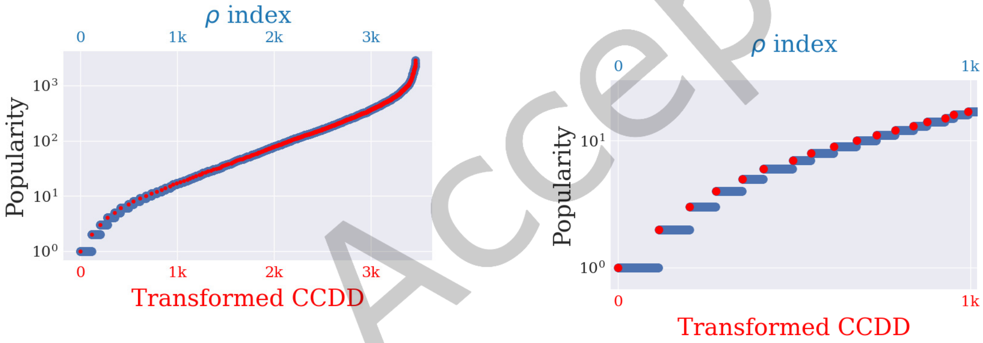


Fig. 1. Typical (log-log) Complementary Cumulative Degree Distributions in Preference Networks. X axis represents the degree of the items, i.e. their popularity. Y axis maps the complementary cumulative frequency for each degree.



(a) Full representation of the popularity score $\boldsymbol{\rho}$ and CCDD.

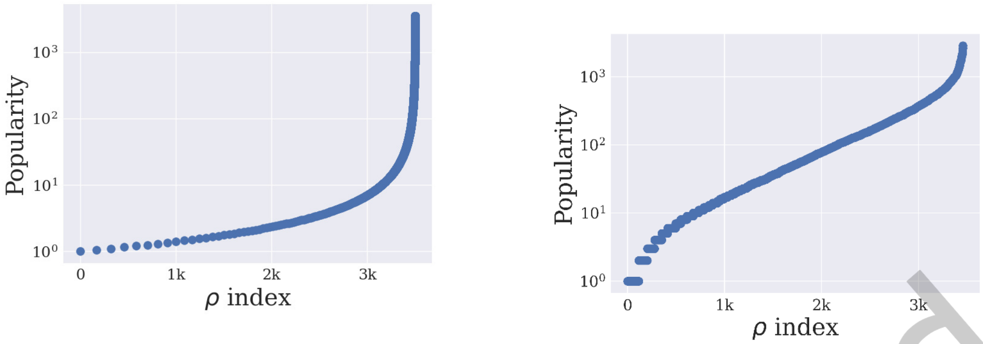
(b) Zoom of the bottom-left corner of Figure 2a.

Fig. 2. Overlapping of $\boldsymbol{\rho}$ and transformed cumulative distribution function. The red points represent the CCDD, whereas the blue ones represent the $\boldsymbol{\rho}$ index. Y axis represents item popularity, in log-scale.

data distribution. On the left side, items strongly differ in exposure, while on the right side, items in I_M exhibit relatively similar popularity, that smoothly grows till τ_H. This dynamic casts low-popular items off the chessboard, while items in I_M can compete and potentially gain prominence.

For this reason, we state that the so-defined low-popular items are the best candidates within the long-tail to be promoted, thereby enhancing fairness and serendipity in recommendation. Additionally, the threshold-based strategy proposed to identify low-popular items offers several advantages:

- It's data-driven, relying solely on the data distribution with no human bias;
- It's general: it can be applied in any scenario where data are represented as preference matrix;
- It's parameter-free, eliminating the need for manual parameter tuning.



(a) $\boldsymbol{\rho}$ distribution for perfectly power-law shaped CCDDs.

(b) $\boldsymbol{\rho}$ distribution for concave shaped CCDDs.

Fig. 3. $\boldsymbol{\rho}$ distributions. X axis is the $\boldsymbol{\rho}$ index. Y axis is the item popularity, in log-scale.

Notably, these advantages are unique for the proposed strategy, as the most used approaches in the literature rely on human-designed choices, such as employing a fixed threshold on the popularity score or a fixed number of items composing the long-tail.

3.3 Leveraging Exposure and Predictive performance

To evaluate and quantify the effects of popularity and novelty in the field of recommender systems, various metrics have been developed. These metrics provide insights into how recommendation algorithms prioritize popular items and the extent to which they incorporate novel suggestions. However, they fail to provide a comprehensive understanding of how the suggested items fit the user's taste. While focusing on recommendation fairness, they overlook the importance of recommendation quality and its alignment with the user's preferences.

In this work, we review five standard metrics used to evaluate the recommendation quality and the effectiveness in boosting the exposure of niche items, namely *Hit-Rate* (*HR*), *Normalized Discounted Cumulative Gain* (*NDCG*), *Average Recommendation Popularity* (*ARP*), *Average Percentage of long-tail items* (*APLT*), *Average Coverage of long-tail items* (*ACLT*), *(Popularity-based) Ranking-based Statistical Parity* (*P-RSP*) and *(Popularity-based) Ranking-based Equal Opportunity* (*P-REO*). In this analysis, we assume to have:

- A subset $T \subset U$ as test users;
- A series of subsets $T_u \subset I_u$ as test items and $I_u^+ = I_u \backslash T_u$ as training items, for each $u \in T$;
- A random subset $Neg_{u,i} \subseteq I \backslash I_u$ of *negative items* (i.e., items that user u did not interacted with), for each $u \in T$ and $i \in T_u$;
- A recommendation list $L_u \subseteq I \backslash I_u^+$, for each $u \in T$.

Hit-Rate. *HR* measures the capability of a recommendation algorithm to retrieve hidden positive items among a large number of negative items. For each user $u \in T$ and item $i \in T_u$, *HR* counts a hit with cut-off k, if i is in the top-k recommended items belonging to the set $\{i\} \cup Neg_{u,i}$. Let H_u^k be the number of hits for the user u with cut-off k. We define the *Hit-Rate at k* as:

$$HR@k = \frac{\sum_{u \in T} H_u^k}{\sum_{u \in T} |T_u|} \,. \tag{1}$$

As it is easy to notice, HR does not take into account the popularity of the recommended items. Its sole objective is to successfully identify the positive items to suggest while filtering out the negative ones.

We can trivially specialize this definition for low-popular items, $HR_L@k$, by considering only items in T_u that belong to that specific class. However, both *HR* and HR_L are not able to provide complete information about recommendation quality and recommendation fairness. The same reasoning can be applied to other quality measures of recommendation, such as ***NDCG***, ***Precision*** and ***Recall***. Unfortunately, even in these cases, a consistent measure that balances predictive accuracy and novelty of suggestions is not achieved.

Normalized Discounted Cumulative Gain. To assess the effectiveness of recommender models in prioritizing relevant items at the top of the recommendation list, we employ the Normalized Discounted Cumulative Gain (*NDCG*) metric [38]. This metric incorporates a logarithmic discount factor based on the position of the relevant item within the ranked list. For a given user u, we compute the Discounted Cumulative Gain (*DCG*) considering the top-k items as follows:

$$DCG(u)@k = \sum_{j=1}^{k} \frac{rel_{j^u}}{log_2(j^u + 1)} \tag{2}$$

where j^u represents the rank of *j-th* item for user u, and rel_{j^u} is the relevance of that item for the user (i.e. 1 for a positive item and 0 for a negative one). The value is divided by the ideal *DCG* (i.e. *iDCG*) representing a perfect ranking to obtain the *NDCG(u)@k*. Then the overall *NDCG* value on the test set is obtained by averaging across all users the equation 2:

$$NDCG@k = \frac{1}{|T|} \sum_{u \in T} \frac{DCG(u)@k}{iDCG(u)@k} \tag{3}$$

In contrast to employing a random subset $Neg_{u,i}$ for negative sampling, as in the case of the *Hit-Rate*, we consider the ranking obtained by scoring the entire set of items $I \backslash I_u^+$ (excluding only those seen in the training set). Notably, a recent study [19] highlights how different negative sampling strategies may yield contradictory performance outcomes: we here assess the robustness of our metric by adopting two evaluation strategies.

Average Recommendation Popularity. *ARP* is a standard metric for evaluating popularity debiasing widely adopted in literature [2–4, 22]. It estimates the average popularity of the items in each recommendation list and is defined as:

$$ARP@k = \frac{1}{|T|} \sum_{u \in T} \frac{\sum_{i \in L_u} \rho_i}{k}, \tag{4}$$

where ρ_i is the (absolute) popularity of the item i and $k = |L_u|$.

By design, this measure can mislead the reader. The lower the value of *ARP*, the higher the exposure of the long-tail items; however, there is no requirement for the model to effectively align with the user's preferences, making the metric difficult to employ in optimization processes.

Average Percentage of long-tail items. *APLT* [1] computes the average percentage of low-popular items in the recommendation list, and is defined as follows:

$$APLT@k = \frac{1}{|T|} \sum_{u \in T} \frac{|\{i, i \in L_u \cap I_L\}|}{|L_u|}, \tag{5}$$

Again, *APLT* does not care if the low-popular items match user's interest, but it limits to measure their frequency.

Average Coverage of long-tail items. [2] introduce *ACLT* as an evaluation measure to address a problem with *APLT*, which could yield high values even if all users receive the same set of low-popular items. The authors state that *ACLT* measures the fraction of long-tail items covered by the recommender and propose the following formulation:

$$ACLT@k = \frac{1}{|T|} \sum_{u \in T} \sum_{i \in L_u} \mathbf{1}(i \in I_L). \tag{6}$$

Here, $\mathbf{1}(i \in I_L)$ is an indicator function that equals 1 if the recommended item $i \in I_L$ (i.e., a low-popular item), and 0 otherwise. The authors aimed at extending the concept of *aggregate diversity* measure (*AD*), which was extensively discussed by [5], by deploying it into the popularity bias field. *AD* measure is defined as the total number of distinct items recommended across all users: However, if we consider the term $\sum_{i \in L_u} \mathbf{1}(i \in I_L)$ in Equation 6, we can notice that it counts, for each user u in the test set T, how many items are within the intersection of L_u and I_L, thus:

$$\sum_{i \in L_u} \mathbf{1}(i \in I_L) = |\{i, i \in L_u \cap I_L\}| \qquad \Rightarrow \qquad APLT@k = \frac{ACLT@k}{k} \quad (since\,|L_u| = k)\,. \tag{7}$$

Given their formulations, the two metrics provide the same information, but scaled by a constant. Moreover, the aggregate diversity measure focuses on the diversity coverage of the entire catalog and fails to consider the quality of predictions, which renders it incomplete.

(Popularity-based) Ranking-based Statistical Parity. *P-RSP* [47] rewards recommender algorithms that enforce the ranking probability distributions of the different popularity groups of items to be the same. For each popularity class $I_g \in \{I_L, I_M, I_H\}$, *P-RSP* begins by summing the ratios between the number of suggested items (we are assuming $L_u \cap I_u^+ = \emptyset$) belonging to I_g and the number of un-interacted items in I_g across all users:

$$q(I_g)@k = \sum_{u \in T} \frac{\left|\left\{i, i \in L_u \cap I_g\right\}\right|}{\left|\left\{i, i \in I_g \cap (I \backslash I_u^+)\right\}\right|}, \tag{8}$$

then, *P-RSP* is defined as the coefficient of variation of the set $\{q(I_L)@k,\, q(I_M)@k,\, q(I_H)@k\}$:

$$P\text{-}RSP@k = \frac{std\,\{q(I_L)@k,\, q(I_M)@k,\, q(I_H)@k\}}{mean\,\{q(I_L)@k,\, q(I_M)@k,\, q(I_H)@k\}}\,. \tag{9}$$

Again, we do not have a ground truth match that provides us with information regarding the predictive ability of the recommendation model.

(Popularity-based) Ranking-based Equal Opportunity. *P-REO* [47] recommendation metric is an evolution of *P-RSP* that takes into account the prediction match with the test set. In particular, *P-REO* extends $q(I_g)$ in $q(I_g|T)$, that represents the summation of the ratios between the number of suggested items, belonging to I_g, that are contained within the test set, and the number of the test set items in I_g:

$$q(I_g|T)@k = \sum_{u \in T} \frac{\left|\left\{i, i \in L_u \cap I_g \cap T_u\right\}\right|}{\left|\left\{i, i \in I_g \cap T_u\right\}\right|}\,. \tag{10}$$

Similarly to Equation 9, *P-REO* is defined as a coefficient of variation:

$$P\text{-}REO@k = \frac{std\,\{q(I_L|T)@k,\, q(I_M|T)@k,\, q(I_H|T)@k\}}{mean\,\{q(I_L|T)@k,\, q(I_M|T)@k,\, q(I_H|T)@k\}}\,. \tag{11}$$

By incorporating the test set, *P-REO* offers a more comprehensive view than its predecessors. However, we are still far from defining a metric capable of clearly expressing both the predictive ability and the diversification ability of a recommendation algorithm. In addition, *P-REO* pushes the algorithm to balance among the popularity classes, even at the expense of its predictive quality.

We stress that, despite all these well-known metrics give information related to the exposure of low-popular items in the user list, they provide no information about the quality of such recommendations. For this reason, we want to compare their effectiveness in evaluating debiasing with our proposal, *Balanced Quality Score*.

3.4 Balanced Quality Score (*BQS*)

In Section 2, we have seen that the literature is rich in techniques that aim at improving the ability of a baseline model to successfully suggest long-tail items. However, as mentioned in Section 3.3, at the best of our knowledge, there is currently no self-contained metric that offers insights into both recommendation quality and exposure to low-popular items. This lack of a comprehensive metric makes it challenging to compare different improvement strategies.

For this reason, in this work, we propose the *Balanced Quality Score* (*BQS*) measure. The purpose of *BQS* is to quantify the improvement achieved on low-popular items through the integration of a debiasing technique within a baseline recommendation algorithm while considering the potential impact on overall recommendation quality. In other words, *BQS* can measure the extent to which a debiasing model D improves upon a baseline recommender system B, in terms of bias mitigation and recommendation quality.

Let QM be any traditional quality measure for the prediction (e.g., Hit-Rate, Recall, NDCG, ...). We define $\phi@k$ as the difference between QM^D, that represents the quality obtained by D, and QM^B, that represents the quality obtained by B, with k as cut-off:

$$\phi@k = QM^D@k - QM^B@k\,. \tag{12}$$

Similarly, we can obtain $\phi_L@k$ by computing the quality measures over the low-popular items only. Notably, $\phi@k$ can be either positive or negative, thus representing either a gain or a loss resulting from the adoption of the model at hand in place of the baseline. We exploit this difference in a gain/loss function $\Phi@k$:

$$\Phi@k = \begin{cases} \phi@k & \text{if } \phi@k \geq 0 \\ -(a\ \phi@k)^2 + \phi@k & \text{otherwise} \end{cases}, \tag{13}$$

that quadratically penalizes losses, while considering linear gains.[3] The constant term $a > 1$ is used to activate a dramatic penalization for values lower than $-\frac{1}{a}$ (corresponding to situations where the quadratic term dominates on the linear one). Similarly, we can devise $\Phi_L@k$ based on $\phi_L@k$ in Equation 13. The rationale is to ensure that empowered exposure of the Long-tail should not excessively affect the overall quality of the recommendation, hence, gains linearly contribute to *BQS*, while losses produce a quadratic cost.

Taking into account $\Phi@k$ and $\Phi_L@k$, we can define $BQS@k$ as the sigmoid function of their linear combination:

$$BQS@k = \sigma\left(\Phi@k + \Phi_L@k\right)\,, \tag{14}$$

where $\sigma(\omega) = (1 + e^{-\omega})^{-1}$. So defined, $BQS \in [0, 1]$ and denotes the balance between the relative improvement on the low-popular items and the loss in terms of global quality. Higher values correspond to substantial improvement in the exposure of low-popular, at the cost of negligible reductions or even gains in global accuracy.

We claim that, in comparison to other measures in the literature (discussed in Section 3.3), *BQS* is more effective in highlighting the underlying benefits or drawbacks of applying a debiasing strategy both in terms of prediction quality and Long-tail exposure. Furthermore, *BQS* can be used on its own to estimate the overall recommendation quality of the debiasing algorithm, as well as the exposure provided to long-tail items, with no additional support, making the metric suitable in optimization processes.

As a final comment, we would like to emphasize that *BQS* can also be utilized to evaluate the balance between the overall recommendation quality and the quality achieved on mid-popular items, or any specific group of items denoted by G, with minimal effort. Equation 12 can be in fact verticalized to compute $\phi_G@k$, instead of $\phi_L@k$.

[3] The additive term $\phi@k$ guarantees the absence of singular values for differentiability.

3.5 Baseline Models

We want to validate and compare the efficacy of the metrics in capturing the debiasing capability of the techniques without impoverishing the global prediction quality. To do so, we have chosen baseline models whose main feature is to compare items and generate a preference ranking for each user. We identify two main families:

- *User-/Item-Embedding* models, that combine both users and items embeddings for each expressed preference;
- *History-Embedding* models, that compute preferences by projecting the user's history into a latent space.

Paradigmatic of the first approach is the Bayesian Personalized Ranking (*BPR*) model introduced in [31]. Its underlying idea is that a preference $i \succ_u j$ can be directly explained as closeness in a latent space where both items and users can be mapped. This can be devised by computing a factorization rank $\mathbf{p}_u \cdot \mathbf{q}_i$ for each pair (u, i), with $\mathbf{p}_u$ (resp. $\mathbf{q}_i$) being the user (resp. item) embedding, and modeling precedences by means of a Bernoulli process: $i \succ_u j \sim \mathit{Bernoulli}(p)$, where $p = \sigma\left(\mathbf{p}_u \cdot (\mathbf{q}_i - \mathbf{q}_j)\right)$ and $\sigma(\alpha) = (1 + e^{-\alpha})^{-1}$ is the logistic function. The optimal embeddings $\mathbf{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_M\}$ and $\mathbf{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_N\}$ can hence be obtained by optimizing the loss:

$$\ell_{BPR}(\mathbf{P}, \mathbf{Q}) = -\sum_{u} \sum_{\substack{i,j \\ i \succ_u j}} \log \sigma\left(\mathbf{p}_u \cdot (\mathbf{q}_i - \mathbf{q}_j)\right) + \mathit{Regularization}\,. \tag{15}$$

As representative of the second algorithm family, we choose a ranking-based version of *Mult-VAE* framework proposed in [23, 25, 32, 34, 35], namely *Ranking Variational Autoencoder* (*RVAE*). The latter keeps the *Mult-VAE* network topology but alters the loss function by focusing on pair-wise comparisons, as follows:

$$\ell_{RVAE}(\phi, \theta) = -\sum_{u} \mathbb{E}_{\mathbf{z} \sim Q_\theta(\cdot|\mathbf{x}_u)}\left[\sum_{i \succ_u j} \log P_\phi(i \succ_u j|\mathbf{z})\right] + \mathbb{KL}\left[Q_\theta(\mathbf{z}|\mathbf{x}_u) \| P(\mathbf{z})\right] \,, \tag{16}$$

where Q_θ is the Encoder module, P_ϕ is the Decoder module, and $\mathbf{z}$ is the latent representation of the input data $\mathbf{x}_u$.

To generalize our findings, we also explore a more recent baseline, *SimGCL* [44], which leverages the expressive potential of Graph Neural Networks within the collaborative filtering domain. *SimGCL* is optimized using the BPR loss in Equation 15 and a self-supervised term which uses contrastive views of the user-item bipartite graphs. We refer the reader to the original paper [44] for the details.

Learning by Negative Sampling. In the above formulation, there are some details that are worth further discussion. In both *BPR*, *SimGCL* and *RVAE* models, optimizing the loss function requires that all pairs of items are considered within Equations 15 and 16. This is unrealistic with large item bases, and it is usual to consider the subset $\mathcal{D}_u$ of pairwise comparisons (see Section 3.1). This way, Equation 15 and Equation 16 can be respectively rewritten as:

$$\ell_{BPR}(\mathbf{P}, \mathbf{Q}) = -\sum_{u} \sum_{(i,j) \in \mathcal{D}_u} \log \sigma\left(\mathbf{p}_u^T (\mathbf{q}_i - \mathbf{q}_j)\right) + \mathit{Regularization}\,, \tag{17}$$

$$\ell_{RVAE}(\phi, \theta) = -\sum_{u} \mathbb{E}_{\mathbf{z} \sim Q_\theta(\cdot|\mathbf{x}_u)}\left[\sum_{(i,j) \in \mathcal{D}_u} \log P_\phi(i \succ_u j|\mathbf{z})\right] + \mathbb{KL}\left[Q_\theta(\mathbf{z}|\mathbf{x}_u) \| P(\mathbf{z})\right] \,. \tag{18}$$

The sampling of $\mathcal{D}_u$ represents a trade-off between accuracy and training efficiency in the underlying predictive model. A standard approach in literature is to uniformly sample for each user u and item i, a fixed number of n items $\{j_1, \ldots j_n\} \subset I - I_u$ with the assumption that $\forall k : i \succ_u j_k$. If such a subset exists, i is called a positive item and $j_1, \ldots j_n$ are called negative items.

3.6 Debiasing techniques

Here we list the debiasing techniques we are going to exploit to prove the effectiveness of our proposal. We want to point out that we have selected techniques from the literature that can be seamlessly integrated into any recommendation system with minimal or no significant modification to the underlying algorithm (in our case *BPR*, *SimGCL* or *RVAE*):

- **Uniform oversampling** (*S,u*) consists in the oversampling strategy in which the number of pairwise comparisons is kept constant during the training phase.
- **Dynamic oversampling** (*S,d*) refers to an oversampling strategy that, for each item $i \in I$, dynamically computes the n_i pairwise comparisons to sample. More details are specified in the Appendix A.
- **Ensemble** (*E*) is an ensembling strategy that combines the pretrained baseline with a pretrained variant that focuses on low-popular items only. The final score is defined as $\boldsymbol{\zeta}^E = Softmax(\boldsymbol{\zeta}^B) + \delta \cdot Softmax(\boldsymbol{\zeta}^L)$, where $\boldsymbol{\zeta}^B$ is the score vector produced by the baseline, and $\boldsymbol{\zeta}^L$ be the score produced by the low-only model. The ensemble parameter δ is computed through a greedy search (exploiting *BQS*, but forcing $\delta > 0$) and Table 2 shows the optimal results for *RVAE* and *BPR*.
- **Jannach** (*Jan*) [20] adopts an oversampling strategy to increase the occurrences of low-popular items in the training set.
- **IPS** (*IPS*) [33] weights the loss of each positive item with its inverse popularity score.
- **Boratto** (regularizer) ($b^{(r)}$) [7] adds a penalty in the loss equal to the correlation between the loss residuals and the items popularity.
- **Popularity Deconfounding** (*PD*) [39] applies the *do-calculus* used in causal inference to perform popularity debiasing.

4 EXPERIMENTS

We performed an empirical analysis, aimed at corroborating the hypothesis that our metric is able to better show the debiasing capability of the technique, thus providing information on its recommendation quality.

4.1 Datasets

We exploited the following popular benchmark datasets, coming from different domains and hence with specific features:

- **Movielens-1M**[4], containing movies ratings by users. The ratings are on a range $[1, 5]$. Since we work on implicit feedback, we binarized the data by associating to each user-item pairs, 1 if the rating provided by the user is strictly greater than 3 and 0 otherwise.
- **Amazon-GGF**[5], containing e-commerce review data. It is focused on products belonging to Grocery and Gourmet Food. Again, ratings are on a $[1, 5]$ scale and they have been binarized as for Movielens-1M.
- **Pinterest**[6], extracted from the social media *Pinterest.com*, which allows users to save or *pin* an image (item) to their board. The dataset denotes as 1 the pinned images for each user.
- **Citeulike-a**[7], obtained from the homonymous service which provides a digital catalog to save and share academic papers. A user preference is encoded as 1 if the user has saved the paper (item) in his/her library.
- **Yahoo-r3**[8], gathered from Yahoo! Music. It contains user-item ratings during normal interaction with the service. Again, ratings are on a $[1, 5]$ scale and they have been binarized by only keeping those strictly

[4] https://grouplens.org/datasets/movielens/1m/

[5] http://jmcauley.ucsd.edu/data/amazon/index_2014.html

[6] https://www.kaggle.com/minnieliang/rec-system/version/2

[7] https://github.com/js05212/Citeulike-a

[8] https://webscope.sandbox.yahoo.com/catalog.php?datatype=r

Dataset	#Users	#Items	#Low	%Low	#Med	%Med	#High	%High	#Ratings	Sparsity
MovieLens-1M	6031	3462	496	14.33%	2620	75.68%	346	9.99%	571450	97.25%
Amazon-GGF	74688	21800	2174	9.97%	17463	80.11%	2163	9.22%	636919	99.96%
Citeulike-a	5551	16875	2369	14.04%	12927	76.60%	1579	9.36%	204776	99.78%
Pinterest	55187	9892	1467	14.83%	7439	75.20 %	986	9.97%	1500779	99.73%
Yahoo-r3	9735	975	107	10.97%	771	79.08%	97	9.95%	117688	98.76%

Table 1. Summary statistics of benchmark datasets

greater than 3 as positives. A nice feature of this dataset is that a standard test set is pre-built to prevent specific biases: in fact, it is built by collecting the ratings obtained by exposing 10 random items to 5, 400 users.

Each dataset was preprocessed by removing outlier users who preferred more than 1, 000 or less than 5 items. The general properties (*number of users*, *items*, *ratings* and *sparsity index*) and the statistics related to low-, medium- and high-popular items are reported in Table 1.

4.2 Settings

We study the behavior of the *RVAE*, *SimGCL* and *BPR* model instances within the popularity classes. To do so, we adopt the following protocols.

Let us consider *RVAE* first. For each dataset (except for Yahoo-r3, where the split is predefined), the training set is composed of 70% of randomly sampled users. Each such user is associated with $\mathbf{x}_u$ and the set $\mathcal{D}_u$ of positive/negative item pairs. The remaining 30% of users is uniformly split into validation and test set. In particular, for each user u, we consider a random subset $T_u \subset I_u$ representing the 30% of the positive items. The vector $\mathbf{x}_u$ is masked to remove all elements in T_u. We then feed the masked $\mathbf{x}_u$ to obtain the score vector $\boldsymbol{\zeta}_{\boldsymbol{u}}$.

Concerning *BPR* and *SimGCL*, the above protocol requires some adaptation. The *BPR* (and *SimGCL*) algorithm computes an embedding for each user (resp. item). This requires each user (resp. item) in the validation/test set to be observed also in the training set. To ensure this requirement, for each user in the validation/test set, we enforce the following: *(i)* the 70% of the items for each user is inserted in the training set, *(ii)* the remaining items (i.e. the 30%) are considered when populating T_u. The alignment of T_u on *RVAE*, *SimGCL* and *BPR* guarantees that their performances are comparable.

The three models were trained on simple architectures: the *BPR* architecture is a regularized embedding of users and items (with latent size 32); while *RVAE* is an encoder-decoder with two layers of size 600 and 200 (as the original MVAE proposed in [24]). *SimGCL* model is instantiated using the default hyper-parameters reported in the original paper [44] and latent size equal to 32.

We choose, for each user u and item $i \in I_u$, a fixed number of $n = 4$ negative items to populate $\mathcal{D}_u$, and $k \in \{1, 5, 10\}$ as cut-offs since the impact of the bias effect can be easily observed with small recommendation lists [7].

Following from the discussion in Section 3.2, we need to practically identify the popularity thresholds (namely, τ_L and τ_H) on the $\boldsymbol{\rho}$ function in an automated manner. In practice, our claim is that the properties of $\boldsymbol{\rho}$ itself can be exploited to identify the popularity categories. For this purpose, we perform the following two-steps procedure over the $\boldsymbol{\rho}$ distribution. First, we apply the Savgol smoothing filter [34] to get rid of potential change points on the curve: this is a mandatory step, since for locating the elbows of the function, we need to first find its inflection points, and thus compute the derivatives. Second, in order to estimate the exact location of the elbows, we exploit the function `rotor` from the `kneebow` package [21], which provides an ad-hoc method for the purpose (i.e., `find_elbow`) based on the geometric properties of the underlying curve.

All the code and data to reproduce the experiments are available online [9].

Dataset	δ *RVAE*^{E}	δ *SimGCL*^{E}	δ *BPR*^{E}
MovieLens-1M	0.4	0.05	0.754
Amazon-GGF	0.2	0.05	0.000001
Citeulike-a	0.7	0.05	0.000001
Pinterest	0.6	0.6	0.000001
Yahoo-r3	0.55	0.75	0.000027

Table 2. Best δ score per dataset for *RVAE*, *SimGCL* and *BPR*.

4.3 Results

Tables 3, 4 and 5, 6, 7, 8, 9, 10, 11 summarize the evaluation results of the aforementioned strategies, over all the considered cut-offs, by adopting *RVAE*, *BPR* and *SimGCL*. The tables report the scores in terms of either *HR* and HR_L, or *NDCG* and $NDCG_L$, as well as *ARP*, *APLT*, *P-REO* and *BQS*.

Each experiment was obtained by averaging five different runs. Values in bold and underlined on *ARP*@$\{1, 5, 10\}$, *APLT*@$\{1, 5, 10\}$, *P-REO*@$\{1, 5, 10\}$, and *BQS*@$\{1, 5, 10\}$ represent the best and second-best results, respectively, according to ANOVA statistical significance [18]. Recalling the definition, the metrics have to be read differently: the best values in terms of *APLT* and *BQS* are the highest, while they are the lowest according to *ARP* and *P-REO*.

Notice that, we are not here interested in comparing the debiasing methods: indeed, we aim at evaluating the effectiveness of each reported metric (*ARP*, *APLT*, *P-REO*, and *BQS*) in capturing the trade-off between the accuracy gain over low-popular items and the impact on the global performance of a given debiasing approach.

Consider the results obtained with *RVAE* first (Tables 3, 4 and 5). Let us focus on Movielens-1M. We see that, in all the given cut-offs, either *ARP*, *APLT*, and *P-REO* agree in choosing the uniform oversampling as the optimal approach. However, if we consider the global performance, we can observe that it drastically decreases, showing a reduction of -0.13, -0.18, -0.15, respectively, in terms of *HR*@$\{1, 5, 10\}$ over the baseline, and of -0.23, -0.21, and -0.19, respectively, in terms of *NDCG*@$\{1, 5, 10\}$. Conversely, *BQS* produces really low scores, thus estimating the uniform oversampling as the worst strategy to adopt. This is because it quadratically penalizes global losses with respect to gain on low-popular items, thus downgrading the techniques that, despite boosting niche items, dramatically affect the overall performance. The best strategy, according to our metric, is the model ensemble in all the considered cut-offs (tied with the baseline at k@1 when adopting the *HR* as the accuracy metric, and at k@1, 5 in terms of *NDCG*). Indeed, at cut-offs k@$\{5, 10\}$, it induces a gain over low-popular items, in spite of a negligible loss over global accuracy.

Similar considerations can be made by looking at the results obtained on Amazon-GGF with cut-off k@1. Here, both *ARP* and *APLT* suggest the uniform oversampling as the best model. Indeed, also in this case the global quality is considerably affected, with a loss of about -0.18 in terms of *HR*, and of -0.03 in terms of *NDCG*, with respect to the vanilla model. According to *P-REO*, instead, the best strategy to consider is *Jannach*, which leads to a slight improvement over low-popular items in terms of *HR*, but severely affects the global quality as well.

BQS@1, instead, selects the dynamic oversampling strategy as optimal when we adopt the *HR* as accuracy measure, while picking Boratto regularizer as the best approach, when *NDCG* is adopted.

These strategies, in fact, lead to a gain both over low-popular items (+0.01 in terms of HR_L and +0.04 in terms of $NDCG_L$) and with respect to global quality (+0.01 over *HR* and +0.03 over *NDCG*).

[9] https://github.com/EricaCoppolillo/BQS

This underlines another limitation of the other metrics that our proposal overcomes: taking into consideration improvements in global quality as well as gains over low-popular items. Moreover, differently from the other debiasing metrics, our proposal is able to diversify the best strategy to adopt according to the chosen accuracy measure (e.g., *HR*, *NDCG*), thus offering a more flexible solution.

On Citeulike-a at k@1, instead, we can observe a further different case. While *ARP* and *APLT* identify *Jannach* as the winner strategy, *P-REO* and *BQS* match in estimating the best approach as the uniform oversampling. Notably, it is a pure coincidence: indeed, if we look at the second-best algorithm, *P-REO* chooses *PD*, while *BQS* the dynamic oversampling. In this case, besides slightly reducing the global performance, *PD* produces a really poor gain also in terms of low-popular items compared to the baseline; on the contrary, the dynamic oversampling approach induces both a higher low-popular gain and overcomes the baseline in terms of global quality. Here is another important limitation of the *P-REO* metric that our proposal solves: besides ensuring parity among the popularity groups, it provides no certainty about the quality of such recommendations. In other words, low-value scores could be also due to poor performances obtained over all the considered groups.

Similar comments can be made by observing the results table obtained by adopting the other backbone models, namely *BPR* and *SimGCL*.

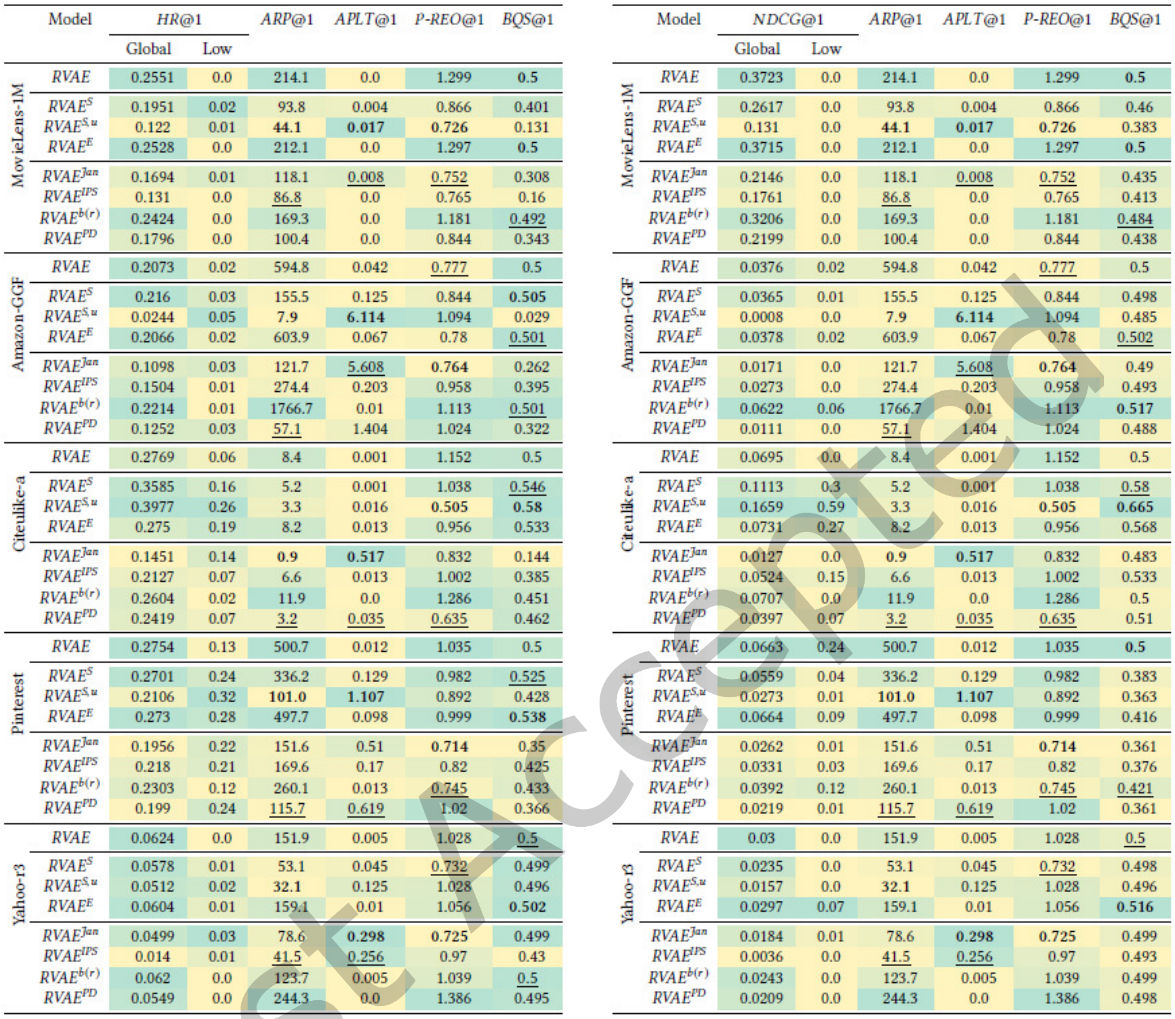
Let us consider the results obtained with *BPR* (Tables 6, 7 and 8).

On MovieLens-1M at cut-off k@1, according to *ARP* and *APLT*, the optimal approach is the uniform oversampling, which induces no gain both in terms of HR_L and $NDCG_L$, but severely degrading the global accuracy. On the other hand, *P-REO* selects *IPS*, which produces a slight improvement over low-popular items in terms of *HR*, but still decreases the overall performance. On the contrary, *BQS*@1 identifies either the ensemble model as the most convenient, when *HR* is adopted as the reference quality metric, or the dynamic oversampling strategy, when *NDCG* is considered. Indeed, in the former case, the approach gains +0.02 on low-popular items over the baseline almost without affecting global quality (-0.002); in the latter, besides not boosting the $NDCG_L$ measure, it obtains the highest global score.

Further, on Amazon-GGF with cut-off k@5, *ARP* selects the dynamic oversampling method as the most effective, while *APLT* and *P-REO* pick again the uniform oversampling as the best candidate. Here, the former approach considerably boosts the low-popular items, gaining +0.04 on HR_L, but decreases the global performance by -0.06. The latter similarly increases the low-popular accuracy of +0.04 but similarly drops the global one, by losing -0.04. On the contrary, *BQS* identifies the baseline as the best approach when *HR* is taken into account, suggesting debiasing methods are not able to induce a better trade-off between the overall recommendation quality and the niche items exposure. Similarly, when considering *NDCG* as reference accuracy measure, *BQS* selects the baseline as the best as well, tying with the ensemble strategy that gains the same scores.

Finally, we focus on the results obtained with *SimGCL* (tables 9, 10, 11), from which we can draw similar conclusions. Consider Pinterest with cut-off k@1. Here, all the competitor metrics disagree in choosing the best debiasing approach: *ARP* selects the ensemble strategy, *APLT* the uniform oversampling, and *P-REO* the *IPS* method. We see that especially the latter is the most inconvenient choice, since the *IPS* strategy gains +0.01 on HR_L over the baseline (nevertheless losing -0.2 in terms of $NDCG_L$), thus dramatically affecting the global performance. Our *BQS* metric, instead, agrees with *ARP* in picking the ensemble as the best method, both in terms of *HR* and *NDCG*, since this strategy greatly boosts the accuracy on low-popular items without degrading global quality. Notice again, it is a pure coincidence that the two measures select the best candidate: if we look at the second-best choice, indeed, *ARP* selects *IPS*, while *BQS* picks either the uniform oversampling (when *HR* is adopted as reference accuracy metric), or *PD* (when *NDCG* is taken into account). Both the strategies, indeed, offer a better solution than the baseline, greatly boosting the recommendation quality over low-popular items (+0.13 in terms of HR_L and +0.02 in terms of $NDCG_L$), without degrading global performance.

Similar considerations, that show the efficacy *BQS* compared to the standard metrics adopted in literature, can be observed in all the other datasets and cut-offs.



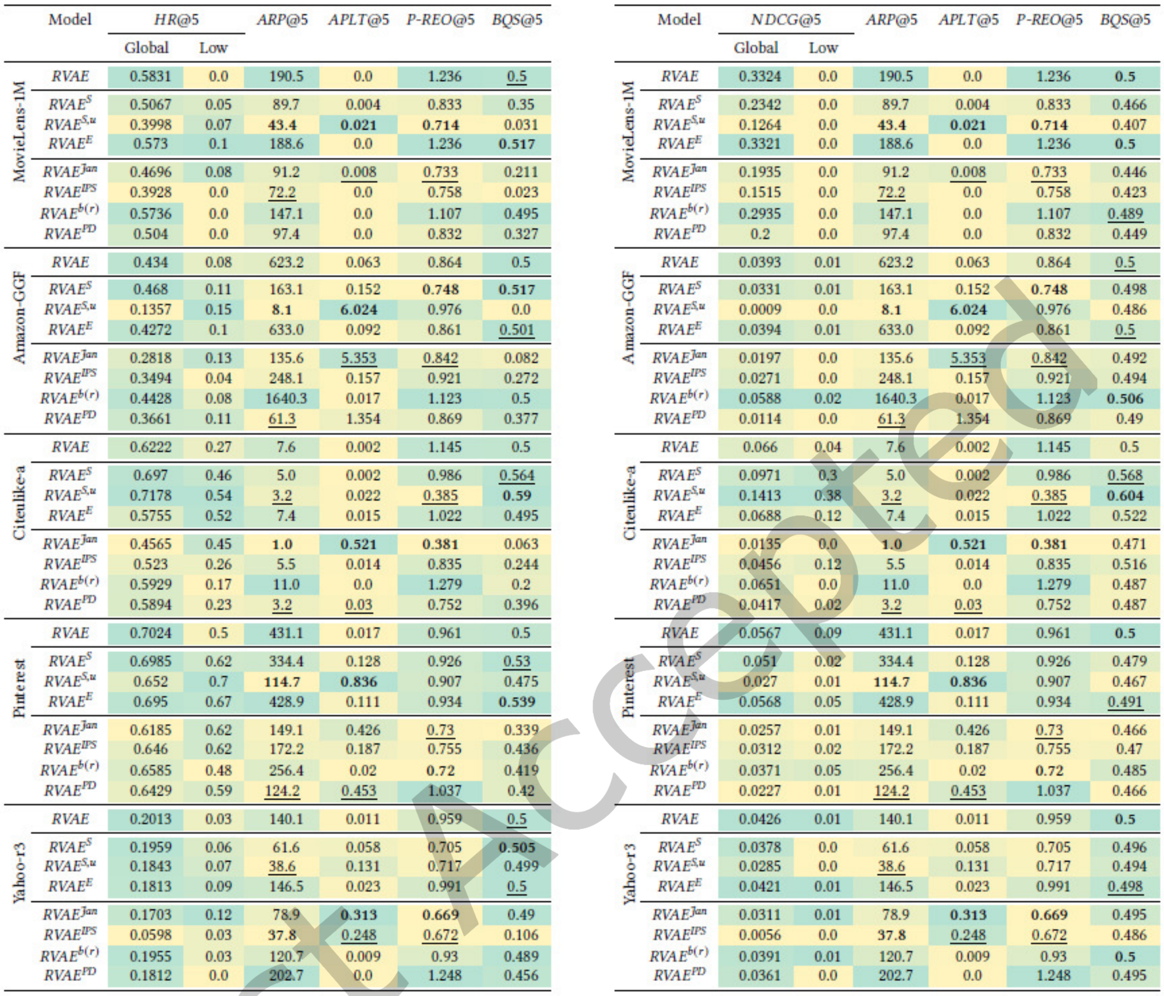
	Model	HR@1		ARP@1	APLT@1	P-REO@1	BQS@1
		Global	Low				
MovieLens-1M	*RVAE*	0.2551	0.0	214.1	0.0	1.299	**0.5**
	$RVAE^{S}$	0.1951	0.02	93.8	0.004	0.866	0.401
	$RVAE^{S,u}$	0.122	0.01	**44.1**	**0.017**	**0.726**	0.131
	$RVAE^{E}$	0.2528	0.0	212.1	0.0	1.297	**0.5**
	$RVAE^{Jan}$	0.1694	0.01	118.1	0.008	0.752	0.308
	$RVAE^{IPS}$	0.131	0.0	86.8	0.0	0.765	0.16
	$RVAE^{b(r)}$	0.2424	0.0	169.3	0.0	1.181	0.492
	$RVAE^{PD}$	0.1796	0.0	100.4	0.0	0.844	0.343
Amazon-GGF	*RVAE*	0.2073	0.02	594.8	0.042	0.777	0.5
	$RVAE^{S}$	0.216	0.03	155.5	0.125	0.844	**0.505**
	$RVAE^{S,u}$	0.0244	0.05	**7.9**	**6.114**	1.094	0.029
	$RVAE^{E}$	0.2066	0.02	603.9	0.067	0.78	0.501
	$RVAE^{Jan}$	0.1098	0.03	121.7	5.608	**0.764**	0.262
	$RVAE^{IPS}$	0.1504	0.01	274.4	0.203	0.958	0.395
	$RVAE^{b(r)}$	0.2214	0.01	1766.7	0.01	1.113	0.501
	$RVAE^{PD}$	0.1252	0.03	57.1	1.404	1.024	0.322
Citeulike-a	*RVAE*	0.2769	0.06	8.4	0.001	1.152	0.5
	$RVAE^{S}$	0.3585	0.16	5.2	0.001	1.038	0.546
	$RVAE^{S,u}$	0.3977	0.26	3.3	0.016	**0.505**	**0.58**
	$RVAE^{E}$	0.275	0.19	8.2	0.013	0.956	0.533
	$RVAE^{Jan}$	0.1451	0.14	**0.9**	**0.517**	0.832	0.144
	$RVAE^{IPS}$	0.2127	0.07	6.6	0.013	1.002	0.385
	$RVAE^{b(r)}$	0.2604	0.02	11.9	0.0	1.286	0.451
	$RVAE^{PD}$	0.2419	0.07	3.2	0.035	0.635	0.462
Pinterest	*RVAE*	0.2754	0.13	500.7	0.012	1.035	0.5
	$RVAE^{S}$	0.2701	0.24	336.2	0.129	0.982	0.525
	$RVAE^{S,u}$	0.2106	0.32	**101.0**	**1.107**	0.892	0.428
	$RVAE^{E}$	0.273	0.28	497.7	0.098	0.999	**0.538**
	$RVAE^{Jan}$	0.1956	0.22	151.6	0.51	**0.714**	0.35
	$RVAE^{IPS}$	0.218	0.21	169.6	0.17	0.82	0.425
	$RVAE^{b(r)}$	0.2303	0.12	260.1	0.013	0.745	0.433
	$RVAE^{PD}$	0.199	0.24	115.7	0.619	1.02	0.366
Yahoo-r3	*RVAE*	0.0624	0.0	151.9	0.005	1.028	0.5
	$RVAE^{S}$	0.0578	0.01	53.1	0.045	0.732	0.499
	$RVAE^{S,u}$	0.0512	0.02	32.1	0.125	1.028	0.496
	$RVAE^{E}$	0.0604	0.01	159.1	0.01	1.056	**0.502**
	$RVAE^{Jan}$	0.0499	0.03	78.6	**0.298**	**0.725**	0.499
	$RVAE^{IPS}$	0.014	0.01	41.5	0.256	0.97	0.43
	$RVAE^{b(r)}$	0.062	0.0	123.7	0.005	1.039	0.5
	$RVAE^{PD}$	0.0549	0.0	244.3	0.0	1.386	0.495

	Model	NDCG@1		ARP@1	APLT@1	P-REO@1	BQS@1
		Global	Low				
MovieLens-1M	*RVAE*	0.3723	0.0	214.1	0.0	1.299	**0.5**
	$RVAE^{S}$	0.2617	0.0	93.8	0.004	0.866	0.46
	$RVAE^{S,u}$	0.131	0.0	**44.1**	**0.017**	**0.726**	0.383
	$RVAE^{E}$	0.3715	0.0	212.1	0.0	1.297	**0.5**
	$RVAE^{Jan}$	0.2146	0.0	118.1	0.008	0.752	0.435
	$RVAE^{IPS}$	0.1761	0.0	86.8	0.0	0.765	0.413
	$RVAE^{b(r)}$	0.3206	0.0	169.3	0.0	1.181	0.484
	$RVAE^{PD}$	0.2199	0.0	100.4	0.0	0.844	0.438
Amazon-GGF	*RVAE*	0.0376	0.02	594.8	0.042	0.777	0.5
	$RVAE^{S}$	0.0365	0.01	155.5	0.125	0.844	0.498
	$RVAE^{S,u}$	0.0008	0.0	**7.9**	**6.114**	1.094	0.485
	$RVAE^{E}$	0.0378	0.02	603.9	0.067	0.78	0.502
	$RVAE^{Jan}$	0.0171	0.0	121.7	5.608	**0.764**	0.49
	$RVAE^{IPS}$	0.0273	0.0	274.4	0.203	0.958	0.493
	$RVAE^{b(r)}$	0.0622	0.06	1766.7	0.01	1.113	**0.517**
	$RVAE^{PD}$	0.0111	0.0	57.1	1.404	1.024	0.488
Citeulike-a	*RVAE*	0.0695	0.0	8.4	0.001	1.152	0.5
	$RVAE^{S}$	0.1113	0.3	5.2	0.001	1.038	0.58
	$RVAE^{S,u}$	0.1659	0.59	3.3	0.016	**0.505**	**0.665**
	$RVAE^{E}$	0.0731	0.27	8.2	0.013	0.956	0.568
	$RVAE^{Jan}$	0.0127	0.0	**0.9**	**0.517**	0.832	0.483
	$RVAE^{IPS}$	0.0524	0.15	6.6	0.013	1.002	0.533
	$RVAE^{b(r)}$	0.0707	0.0	11.9	0.0	1.286	0.5
	$RVAE^{PD}$	0.0397	0.07	3.2	0.035	0.635	0.51
Pinterest	*RVAE*	0.0663	0.24	500.7	0.012	1.035	**0.5**
	$RVAE^{S}$	0.0559	0.04	336.2	0.129	0.982	0.383
	$RVAE^{S,u}$	0.0273	0.01	**101.0**	**1.107**	0.892	0.363
	$RVAE^{E}$	0.0664	0.09	497.7	0.098	0.999	0.416
	$RVAE^{Jan}$	0.0262	0.01	151.6	0.51	**0.714**	0.361
	$RVAE^{IPS}$	0.0331	0.03	169.6	0.17	0.82	0.376
	$RVAE^{b(r)}$	0.0392	0.12	260.1	0.013	0.745	0.421
	$RVAE^{PD}$	0.0219	0.01	115.7	0.619	1.02	0.361
Yahoo-r3	*RVAE*	0.03	0.0	151.9	0.005	1.028	0.5
	$RVAE^{S}$	0.0235	0.0	53.1	0.045	0.732	0.498
	$RVAE^{S,u}$	0.0157	0.0	32.1	0.125	1.028	0.496
	$RVAE^{E}$	0.0297	0.07	159.1	0.01	1.056	**0.516**
	$RVAE^{Jan}$	0.0184	0.01	78.6	**0.298**	**0.725**	0.499
	$RVAE^{IPS}$	0.0036	0.0	41.5	0.256	0.97	0.493
	$RVAE^{b(r)}$	0.0243	0.0	123.7	0.005	1.039	0.499
	$RVAE^{PD}$	0.0209	0.0	244.3	0.0	1.386	0.498

Table 3. Results obtained with *RVAE*, by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off k@1. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.

5 CONCLUSIONS

In this work, we addressed the problem of evaluating and comparing debiasing techniques that enhanced recommender systems to empower the exposure of long-tail items within a catalog.



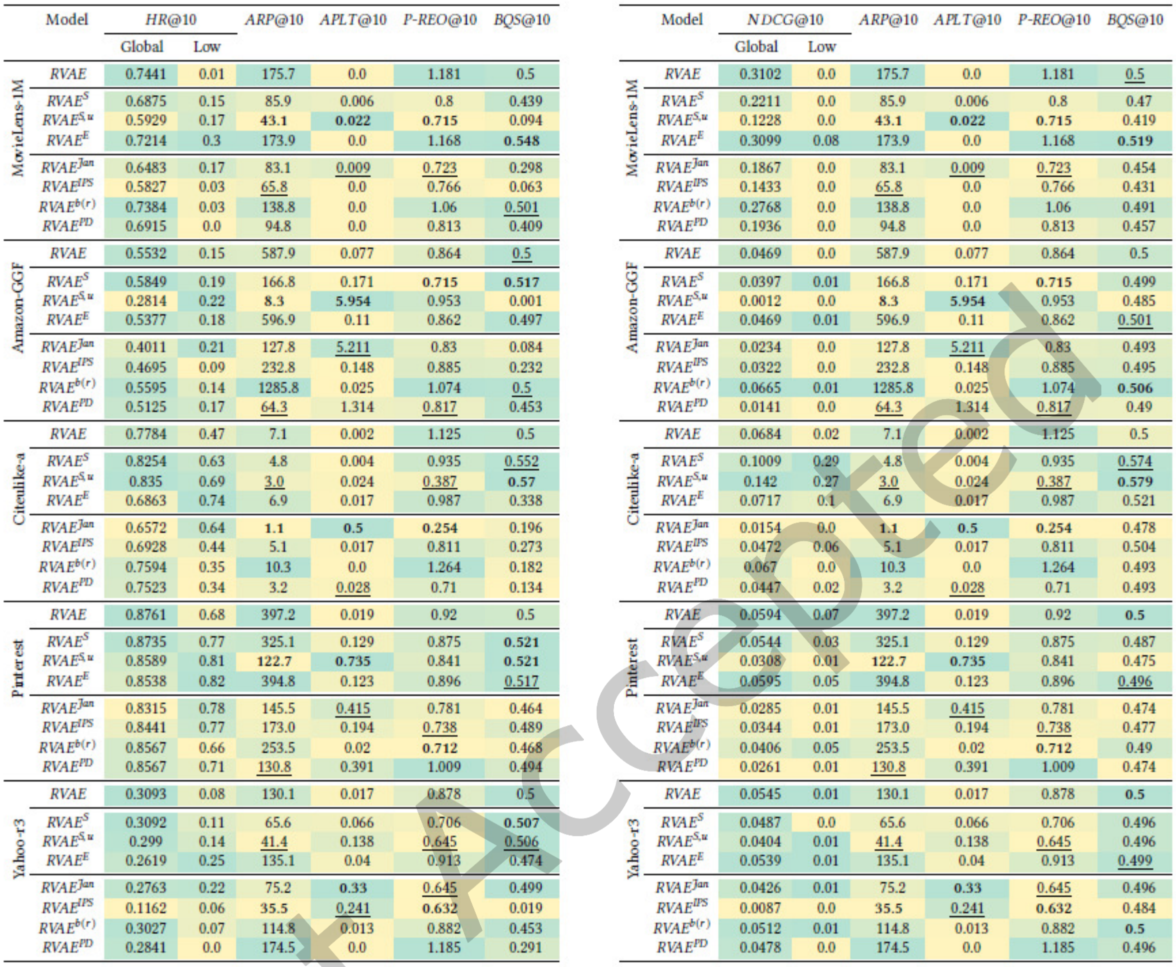
	Model	*HR@5*		*ARP@5*	*APLT@5*	*P-REO@5*	*BQS@5*
		Global	Low				
MovieLens-1M	*RVAE*	0.5831	0.0	190.5	0.0	1.236	0.5
	$RVAE^{S}$	0.5067	0.05	89.7	0.004	0.833	0.35
	$RVAE^{S,u}$	0.3998	0.07	**43.4**	**0.021**	**0.714**	0.031
	$RVAE^{E}$	0.573	0.1	188.6	0.0	1.236	**0.517**
	$RVAE^{Jan}$	0.4696	0.08	91.2	0.008	0.733	0.211
	$RVAE^{IPS}$	0.3928	0.0	72.2	0.0	0.758	0.023
	$RVAE^{b(r)}$	0.5736	0.0	147.1	0.0	1.107	0.495
	$RVAE^{PD}$	0.504	0.0	97.4	0.0	0.832	0.327
Amazon-GGF	*RVAE*	0.434	0.08	623.2	0.063	0.864	0.5
	$RVAE^{S}$	0.468	0.11	163.1	0.152	**0.748**	**0.517**
	$RVAE^{S,u}$	0.1357	0.15	**8.1**	**6.024**	0.976	0.0
	$RVAE^{E}$	0.4272	0.1	633.0	0.092	0.861	0.501
	$RVAE^{Jan}$	0.2818	0.13	135.6	5.353	0.842	0.082
	$RVAE^{IPS}$	0.3494	0.04	248.1	0.157	0.921	0.272
	$RVAE^{b(r)}$	0.4428	0.08	1640.3	0.017	1.123	0.5
	$RVAE^{PD}$	0.3661	0.11	61.3	1.354	0.869	0.377
Citeulike-a	*RVAE*	0.6222	0.27	7.6	0.002	1.145	0.5
	$RVAE^{S}$	0.697	0.46	5.0	0.002	0.986	0.564
	$RVAE^{S,u}$	0.7178	0.54	3.2	0.022	0.385	**0.59**
	$RVAE^{E}$	0.5755	0.52	7.4	0.015	1.022	0.495
	$RVAE^{Jan}$	0.4565	0.45	**1.0**	**0.521**	**0.381**	0.063
	$RVAE^{IPS}$	0.523	0.26	5.5	0.014	0.835	0.244
	$RVAE^{b(r)}$	0.5929	0.17	11.0	0.0	1.279	0.2
	$RVAE^{PD}$	0.5894	0.23	3.2	0.03	0.752	0.396
Pinterest	*RVAE*	0.7024	0.5	431.1	0.017	0.961	0.5
	$RVAE^{S}$	0.6985	0.62	334.4	0.128	0.926	0.53
	$RVAE^{S,u}$	0.652	0.7	**114.7**	**0.836**	0.907	0.475
	$RVAE^{E}$	0.695	0.67	428.9	0.111	0.934	**0.539**
	$RVAE^{Jan}$	0.6185	0.62	149.1	0.426	0.73	0.339
	$RVAE^{IPS}$	0.646	0.62	172.2	0.187	0.755	0.436
	$RVAE^{b(r)}$	0.6585	0.48	256.4	0.02	**0.72**	0.419
	$RVAE^{PD}$	0.6429	0.59	124.2	0.453	1.037	0.42
Yahoo-r3	*RVAE*	0.2013	0.03	140.1	0.011	0.959	0.5
	$RVAE^{S}$	0.1959	0.06	61.6	0.058	0.705	**0.505**
	$RVAE^{S,u}$	0.1843	0.07	38.6	0.131	0.717	0.499
	$RVAE^{E}$	0.1813	0.09	146.5	0.023	0.991	0.5
	$RVAE^{Jan}$	0.1703	0.12	78.9	**0.313**	**0.669**	0.49
	$RVAE^{IPS}$	0.0598	0.03	**37.8**	0.248	0.672	0.106
	$RVAE^{b(r)}$	0.1955	0.03	120.7	0.009	0.93	0.489
	$RVAE^{PD}$	0.1812	0.0	202.7	0.0	1.248	0.456

	Model	*NDCG@5*		*ARP@5*	*APLT@5*	*P-REO@5*	*BQS@5*
		Global	Low				
MovieLens-1M	*RVAE*	0.3324	0.0	190.5	0.0	1.236	**0.5**
	$RVAE^{S}$	0.2342	0.0	89.7	0.004	0.833	0.466
	$RVAE^{S,u}$	0.1264	0.0	**43.4**	**0.021**	**0.714**	0.407
	$RVAE^{E}$	0.3321	0.0	188.6	0.0	1.236	**0.5**
	$RVAE^{Jan}$	0.1935	0.0	91.2	0.008	0.733	0.446
	$RVAE^{IPS}$	0.1515	0.0	72.2	0.0	0.758	0.423
	$RVAE^{b(r)}$	0.2935	0.0	147.1	0.0	1.107	0.489
	$RVAE^{PD}$	0.2	0.0	97.4	0.0	0.832	0.449
Amazon-GGF	*RVAE*	0.0393	0.01	623.2	0.063	0.864	0.5
	$RVAE^{S}$	0.0331	0.01	163.1	0.152	**0.748**	0.498
	$RVAE^{S,u}$	0.0009	0.0	**8.1**	**6.024**	0.976	0.486
	$RVAE^{E}$	0.0394	0.01	633.0	0.092	0.861	0.5
	$RVAE^{Jan}$	0.0197	0.0	135.6	5.353	0.842	0.492
	$RVAE^{IPS}$	0.0271	0.0	248.1	0.157	0.921	0.494
	$RVAE^{b(r)}$	0.0588	0.02	1640.3	0.017	1.123	**0.506**
	$RVAE^{PD}$	0.0114	0.0	61.3	1.354	0.869	0.49
Citeulike-a	*RVAE*	0.066	0.04	7.6	0.002	1.145	0.5
	$RVAE^{S}$	0.0971	0.3	5.0	0.002	0.986	0.568
	$RVAE^{S,u}$	0.1413	0.38	3.2	0.022	0.385	**0.604**
	$RVAE^{E}$	0.0688	0.12	7.4	0.015	1.022	0.522
	$RVAE^{Jan}$	0.0135	0.0	**1.0**	**0.521**	**0.381**	0.471
	$RVAE^{IPS}$	0.0456	0.12	5.5	0.014	0.835	0.516
	$RVAE^{b(r)}$	0.0651	0.0	11.0	0.0	1.279	0.487
	$RVAE^{PD}$	0.0417	0.02	3.2	0.03	0.752	0.487
Pinterest	*RVAE*	0.0567	0.09	431.1	0.017	0.961	**0.5**
	$RVAE^{S}$	0.051	0.02	334.4	0.128	0.926	0.479
	$RVAE^{S,u}$	0.027	0.01	**114.7**	**0.836**	0.907	0.467
	$RVAE^{E}$	0.0568	0.05	428.9	0.111	0.934	0.491
	$RVAE^{Jan}$	0.0257	0.01	149.1	0.426	0.73	0.466
	$RVAE^{IPS}$	0.0312	0.02	172.2	0.187	0.755	0.47
	$RVAE^{b(r)}$	0.0371	0.05	256.4	0.02	**0.72**	0.485
	$RVAE^{PD}$	0.0227	0.01	124.2	0.453	1.037	0.466
Yahoo-r3	*RVAE*	0.0426	0.01	140.1	0.011	0.959	**0.5**
	$RVAE^{S}$	0.0378	0.0	61.6	0.058	0.705	0.496
	$RVAE^{S,u}$	0.0285	0.0	38.6	0.131	0.717	0.494
	$RVAE^{E}$	0.0421	0.01	146.5	0.023	0.991	0.498
	$RVAE^{Jan}$	0.0311	0.01	78.9	**0.313**	**0.669**	0.495
	$RVAE^{IPS}$	0.0056	0.0	**37.8**	0.248	0.672	0.486
	$RVAE^{b(r)}$	0.0391	0.01	120.7	0.009	0.93	**0.5**
	$RVAE^{PD}$	0.0361	0.0	202.7	0.0	1.248	0.495

Table 4. Results obtained with *RVAE*, by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off *k*@5. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.

We first proposed a formal methodology to categorize items into low-, medium- and high- popular, relying on their underlying data distribution shape. To the best of our knowledge, this is the first attempt to overcome the standard 80/20% approach used in literature.

Next, we exploited these classes to define the *Balanced Quality Score* measure (*BQS*) that rewards the debiasing techniques that successfully push the recommender systems to suggest niche items, without losing points in their predictive capability in terms of global accuracy.



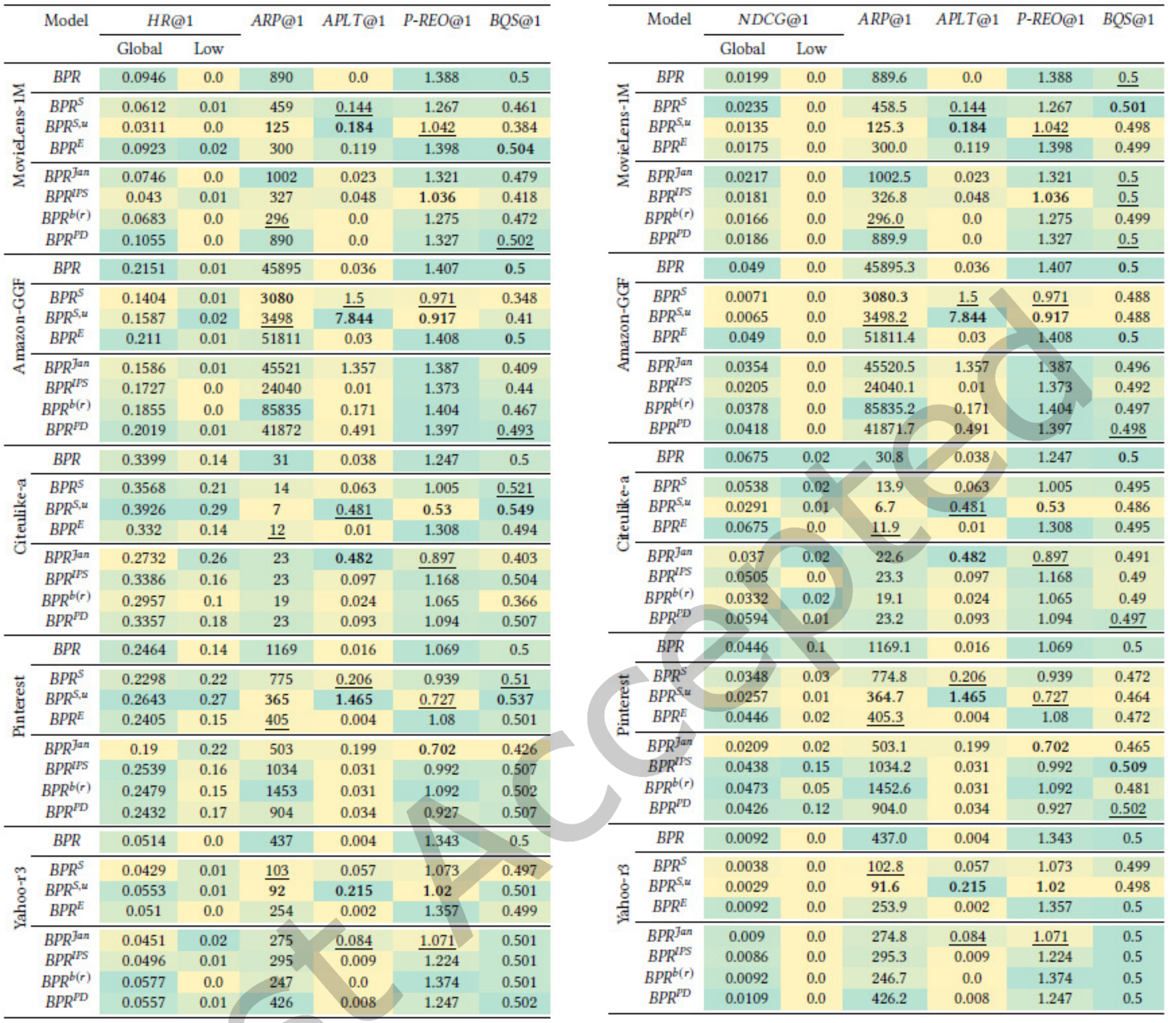
	Model	HR@10		ARP@10	APLT@10	P-REO@10	BQS@10
		Global	Low				
MovieLens-1M	*RVAE*	0.7441	0.01	175.7	0.0	1.181	0.5
	RVAE^{S}	0.6875	0.15	85.9	0.006	0.8	0.439
	RVAE^{S,u}	0.5929	0.17	**43.1**	**0.022**	**0.715**	0.094
	RVAE^{E}	0.7214	0.3	173.9	0.0	1.168	**0.548**
	RVAE^{Jan}	0.6483	0.17	83.1	0.009	0.723	0.298
	RVAE^{IPS}	0.5827	0.03	65.8	0.0	0.766	0.063
	RVAE$^{b(r)}$	0.7384	0.03	138.8	0.0	1.06	0.501
	RVAE^{PD}	0.6915	0.0	94.8	0.0	0.813	0.409
Amazon-GGF	*RVAE*	0.5532	0.15	587.9	0.077	0.864	0.5
	RVAE^{S}	0.5849	0.19	166.8	0.171	**0.715**	**0.517**
	RVAE^{S,u}	0.2814	0.22	**8.3**	**5.954**	0.953	0.001
	RVAE^{E}	0.5377	0.18	596.9	0.11	0.862	0.497
	RVAE^{Jan}	0.4011	0.21	127.8	5.211	0.83	0.084
	RVAE^{IPS}	0.4695	0.09	232.8	0.148	0.885	0.232
	RVAE$^{b(r)}$	0.5595	0.14	1285.8	0.025	1.074	0.5
	RVAE^{PD}	0.5125	0.17	64.3	1.314	0.817	0.453
Citeulike-a	*RVAE*	0.7784	0.47	7.1	0.002	1.125	0.5
	RVAE^{S}	0.8254	0.63	4.8	0.004	0.935	0.552
	RVAE^{S,u}	0.835	0.69	3.0	0.024	0.387	**0.57**
	RVAE^{E}	0.6863	0.74	6.9	0.017	0.987	0.338
	RVAE^{Jan}	0.6572	0.64	**1.1**	**0.5**	**0.254**	0.196
	RVAE^{IPS}	0.6928	0.44	5.1	0.017	0.811	0.273
	RVAE$^{b(r)}$	0.7594	0.35	10.3	0.0	1.264	0.182
	RVAE^{PD}	0.7523	0.34	3.2	0.028	0.71	0.134
Pinterest	*RVAE*	0.8761	0.68	397.2	0.019	0.92	0.5
	RVAE^{S}	0.8735	0.77	325.1	0.129	0.875	**0.521**
	RVAE^{S,u}	0.8589	0.81	**122.7**	**0.735**	0.841	**0.521**
	RVAE^{E}	0.8538	0.82	394.8	0.123	0.896	0.517
	RVAE^{Jan}	0.8315	0.78	145.5	0.415	0.781	0.464
	RVAE^{IPS}	0.8441	0.77	173.0	0.194	0.738	0.489
	RVAE$^{b(r)}$	0.8567	0.66	253.5	0.02	**0.712**	0.468
	RVAE^{PD}	0.8567	0.71	130.8	0.391	1.009	0.494
Yahoo-r3	*RVAE*	0.3093	0.08	130.1	0.017	0.878	0.5
	RVAE^{S}	0.3092	0.11	65.6	0.066	0.706	**0.507**
	RVAE^{S,u}	0.299	0.14	41.4	0.138	0.645	0.506
	RVAE^{E}	0.2619	0.25	135.1	0.04	0.913	0.474
	RVAE^{Jan}	0.2763	0.22	75.2	**0.33**	0.645	0.499
	RVAE^{IPS}	0.1162	0.06	**35.5**	0.241	**0.632**	0.019
	RVAE$^{b(r)}$	0.3027	0.07	114.8	0.013	0.882	0.453
	RVAE^{PD}	0.2841	0.0	174.5	0.0	1.185	0.291

	Model	NDCG@10		ARP@10	APLT@10	P-REO@10	BQS@10
		Global	Low				
MovieLens-1M	*RVAE*	0.3102	0.0	175.7	0.0	1.181	0.5
	RVAE^{S}	0.2211	0.0	85.9	0.006	0.8	0.47
	RVAE^{S,u}	0.1228	0.0	**43.1**	**0.022**	**0.715**	0.419
	RVAE^{E}	0.3099	0.08	173.9	0.0	1.168	**0.519**
	RVAE^{Jan}	0.1867	0.0	83.1	0.009	0.723	0.454
	RVAE^{IPS}	0.1433	0.0	65.8	0.0	0.766	0.431
	RVAE$^{b(r)}$	0.2768	0.0	138.8	0.0	1.06	0.491
	RVAE^{PD}	0.1936	0.0	94.8	0.0	0.813	0.457
Amazon-GGF	*RVAE*	0.0469	0.0	587.9	0.077	0.864	0.5
	RVAE^{S}	0.0397	0.01	166.8	0.171	**0.715**	0.499
	RVAE^{S,u}	0.0012	0.0	**8.3**	**5.954**	0.953	0.485
	RVAE^{E}	0.0469	0.01	596.9	0.11	0.862	0.501
	RVAE^{Jan}	0.0234	0.0	127.8	5.211	0.83	0.493
	RVAE^{IPS}	0.0322	0.0	232.8	0.148	0.885	0.495
	RVAE$^{b(r)}$	0.0665	0.01	1285.8	0.025	1.074	**0.506**
	RVAE^{PD}	0.0141	0.0	64.3	1.314	0.817	0.49
Citeulike-a	*RVAE*	0.0684	0.02	7.1	0.002	1.125	0.5
	RVAE^{S}	0.1009	0.29	4.8	0.004	0.935	0.574
	RVAE^{S,u}	0.142	0.27	3.0	0.024	0.387	**0.579**
	RVAE^{E}	0.0717	0.1	6.9	0.017	0.987	0.521
	RVAE^{Jan}	0.0154	0.0	**1.1**	**0.5**	**0.254**	0.478
	RVAE^{IPS}	0.0472	0.06	5.1	0.017	0.811	0.504
	RVAE$^{b(r)}$	0.067	0.0	10.3	0.0	1.264	0.493
	RVAE^{PD}	0.0447	0.02	3.2	0.028	0.71	0.493
Pinterest	*RVAE*	0.0594	0.07	397.2	0.019	0.92	**0.5**
	RVAE^{S}	0.0544	0.03	325.1	0.129	0.875	0.487
	RVAE^{S,u}	0.0308	0.01	**122.7**	**0.735**	0.841	0.475
	RVAE^{E}	0.0595	0.05	394.8	0.123	0.896	0.496
	RVAE^{Jan}	0.0285	0.01	145.5	0.415	0.781	0.474
	RVAE^{IPS}	0.0344	0.01	173.0	0.194	0.738	0.477
	RVAE$^{b(r)}$	0.0406	0.05	253.5	0.02	**0.712**	0.49
	RVAE^{PD}	0.0261	0.01	130.8	0.391	1.009	0.474
Yahoo-r3	*RVAE*	0.0545	0.01	130.1	0.017	0.878	**0.5**
	RVAE^{S}	0.0487	0.0	65.6	0.066	0.706	0.496
	RVAE^{S,u}	0.0404	0.01	41.4	0.138	0.645	0.496
	RVAE^{E}	0.0539	0.01	135.1	0.04	0.913	0.499
	RVAE^{Jan}	0.0426	0.01	75.2	**0.33**	0.645	0.496
	RVAE^{IPS}	0.0087	0.0	**35.5**	0.241	**0.632**	0.484
	RVAE$^{b(r)}$	0.0512	0.01	114.8	0.013	0.882	**0.5**
	RVAE^{PD}	0.0478	0.0	174.5	0.0	1.185	0.496

Table 5. Results obtained with *RVAE*, by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off k@10. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.

The experimentation, conducted on several benchmark datasets, three baselines and numerous competitors, shows that the proposed strategy is the best in highlighting the debiasing techniques with the highest improvements in the exposure of low-popular items without degrading global quality, exhibiting a competitive advantage over the state-of-the-art. In fact, *BQS* has proven to be used in optimization processes.

Still, other aspects can be investigated in future work. Bias can occur in other contexts besides popularity, where underexposure can result in unfair recommendations. In this context, it would be interesting to investigate whether new mitigation strategies can be defined with the related quality measures. Also, temporal effects (e.g. obsolescence or popularity decay) should be taken into account in implementing mitigation strategies.



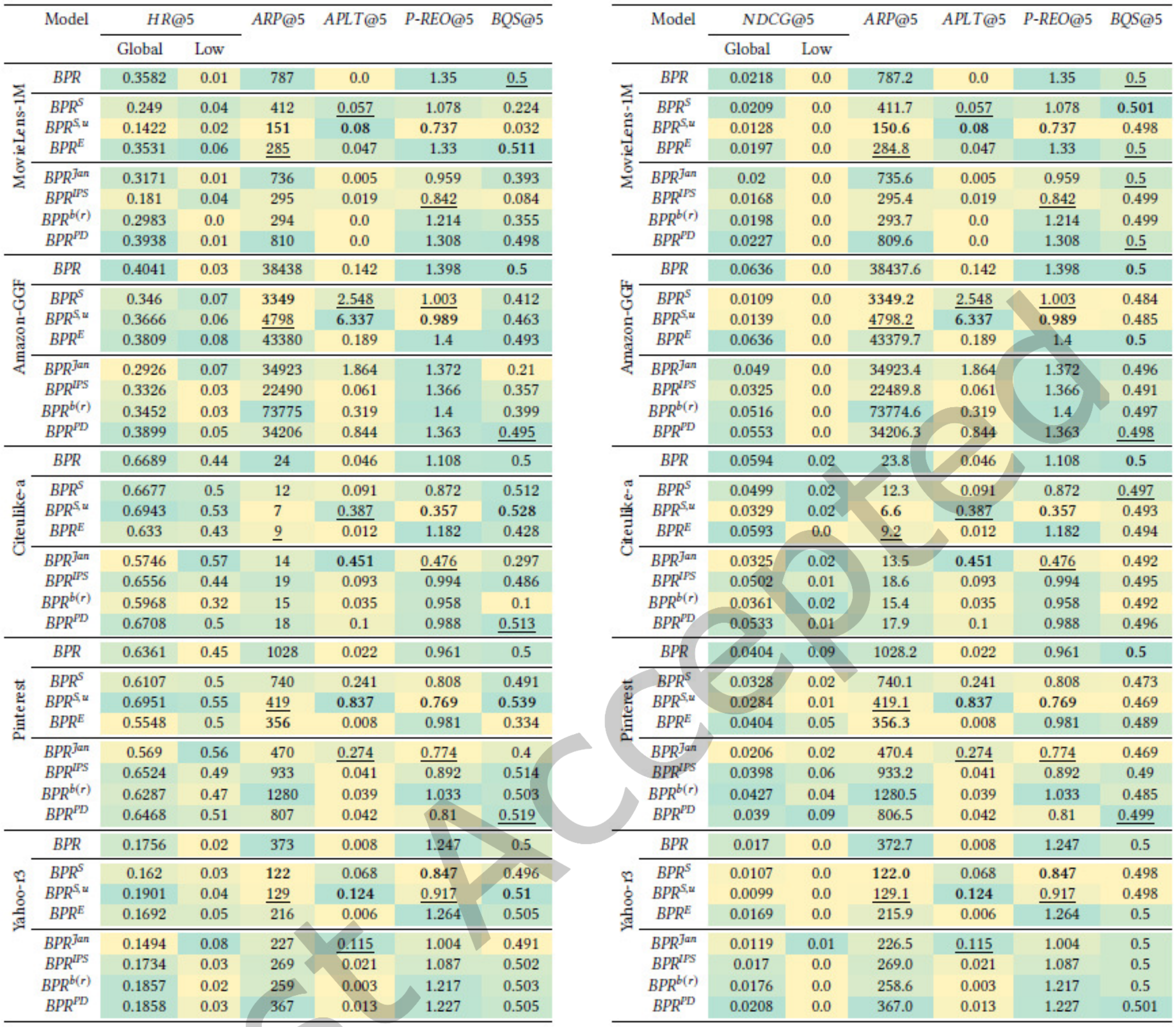
	Model	*HR@1*		*ARP@1*	*APLT@1*	*P-REO@1*	*BQS@1*
		Global	Low				
MovieLens-1M	*BPR*	0.0946	0.0	890	0.0	1.388	0.5
	BPR^{S}	0.0612	0.01	459	0.144	1.267	0.461
	$BPR^{S,u}$	0.0311	0.0	**125**	**0.184**	1.042	0.384
	BPR^{E}	0.0923	0.02	300	0.119	1.398	**0.504**
	BPR^{Jan}	0.0746	0.0	1002	0.023	1.321	0.479
	BPR^{IPS}	0.043	0.01	327	0.048	**1.036**	0.418
	$BPR^{b(r)}$	0.0683	0.0	296	0.0	1.275	0.472
	BPR^{FD}	0.1055	0.0	890	0.0	1.327	0.502
Amazon-GGF	*BPR*	0.2151	0.01	45895	0.036	1.407	**0.5**
	BPR^{S}	0.1404	0.01	**3080**	1.5	0.971	0.348
	$BPR^{S,u}$	0.1587	0.02	3498	**7.844**	**0.917**	0.41
	BPR^{E}	0.211	0.01	51811	0.03	1.408	**0.5**
	BPR^{Jan}	0.1586	0.01	45521	1.357	1.387	0.409
	BPR^{IPS}	0.1727	0.0	24040	0.01	1.373	0.44
	$BPR^{b(r)}$	0.1855	0.0	85835	0.171	1.404	0.467
	BPR^{FD}	0.2019	0.01	41872	0.491	1.397	0.493
Citeulike-a	*BPR*	0.3399	0.14	31	0.038	1.247	0.5
	BPR^{S}	0.3568	0.21	14	0.063	1.005	0.521
	$BPR^{S,u}$	0.3926	0.29	7	0.481	**0.53**	**0.549**
	BPR^{E}	0.332	0.14	12	0.01	1.308	0.494
	BPR^{Jan}	0.2732	0.26	23	**0.482**	0.897	0.403
	BPR^{IPS}	0.3386	0.16	23	0.097	1.168	0.504
	$BPR^{b(r)}$	0.2957	0.1	19	0.024	1.065	0.366
	BPR^{FD}	0.3357	0.18	23	0.093	1.094	0.507
Pinterest	*BPR*	0.2464	0.14	1169	0.016	1.069	0.5
	BPR^{S}	0.2298	0.22	775	0.206	0.939	0.51
	$BPR^{S,u}$	0.2643	0.27	**365**	**1.465**	0.727	**0.537**
	BPR^{E}	0.2405	0.15	405	0.004	1.08	0.501
	BPR^{Jan}	0.19	0.22	503	0.199	**0.702**	0.426
	BPR^{IPS}	0.2539	0.16	1034	0.031	0.992	0.507
	$BPR^{b(r)}$	0.2479	0.15	1453	0.031	1.092	0.502
	BPR^{FD}	0.2432	0.17	904	0.034	0.927	0.507
Yahoo-r3	*BPR*	0.0514	0.0	437	0.004	1.343	0.5
	BPR^{S}	0.0429	0.01	103	0.057	1.073	0.497
	$BPR^{S,u}$	0.0553	0.01	**92**	**0.215**	**1.02**	0.501
	BPR^{E}	0.051	0.0	254	0.002	1.357	0.499
	BPR^{Jan}	0.0451	0.02	275	0.084	1.071	0.501
	BPR^{IPS}	0.0496	0.01	295	0.009	1.224	0.501
	$BPR^{b(r)}$	0.0577	0.0	247	0.0	1.374	0.501
	BPR^{FD}	0.0557	0.01	426	0.008	1.247	0.502

	Model	*NDCG@1*		*ARP@1*	*APLT@1*	*P-REO@1*	*BQS@1*
		Global	Low				
MovieLens-1M	*BPR*	0.0199	0.0	889.6	0.0	1.388	0.5
	BPR^{S}	0.0235	0.0	458.5	0.144	1.267	**0.501**
	$BPR^{S,u}$	0.0135	0.0	**125.3**	**0.184**	1.042	0.498
	BPR^{E}	0.0175	0.0	300.0	0.119	1.398	0.499
	BPR^{Jan}	0.0217	0.0	1002.5	0.023	1.321	0.5
	BPR^{IPS}	0.0181	0.0	326.8	0.048	**1.036**	0.5
	$BPR^{b(r)}$	0.0166	0.0	296.0	0.0	1.275	0.499
	BPR^{FD}	0.0186	0.0	889.9	0.0	1.327	0.5
Amazon-GGF	*BPR*	0.049	0.0	45895.3	0.036	1.407	**0.5**
	BPR^{S}	0.0071	0.0	**3080.3**	1.5	0.971	0.488
	$BPR^{S,u}$	0.0065	0.0	3498.2	**7.844**	**0.917**	0.488
	BPR^{E}	0.049	0.0	51811.4	0.03	1.408	**0.5**
	BPR^{Jan}	0.0354	0.0	45520.5	1.357	1.387	0.496
	BPR^{IPS}	0.0205	0.0	24040.1	0.01	1.373	0.492
	$BPR^{b(r)}$	0.0378	0.0	85835.2	0.171	1.404	0.497
	BPR^{FD}	0.0418	0.0	41871.7	0.491	1.397	0.498
Citeulike-a	*BPR*	0.0675	0.02	30.8	0.038	1.247	**0.5**
	BPR^{S}	0.0538	0.02	13.9	0.063	1.005	0.495
	$BPR^{S,u}$	0.0291	0.01	**6.7**	0.481	**0.53**	0.486
	BPR^{E}	0.0675	0.0	11.9	0.01	1.308	0.495
	BPR^{Jan}	0.037	0.02	22.6	**0.482**	0.897	0.491
	BPR^{IPS}	0.0505	0.0	23.3	0.097	1.168	0.49
	$BPR^{b(r)}$	0.0332	0.02	19.1	0.024	1.065	0.49
	BPR^{FD}	0.0594	0.01	23.2	0.093	1.094	0.497
Pinterest	*BPR*	0.0446	0.1	1169.1	0.016	1.069	0.5
	BPR^{S}	0.0348	0.03	774.8	0.206	0.939	0.472
	$BPR^{S,u}$	0.0257	0.01	**364.7**	**1.465**	0.727	0.464
	BPR^{E}	0.0446	0.02	405.3	0.004	1.08	0.472
	BPR^{Jan}	0.0209	0.02	503.1	0.199	**0.702**	0.465
	BPR^{IPS}	0.0438	0.15	1034.2	0.031	0.992	**0.509**
	$BPR^{b(r)}$	0.0473	0.05	1452.6	0.031	1.092	0.481
	BPR^{FD}	0.0426	0.12	904.0	0.034	0.927	0.502
Yahoo-r3	*BPR*	0.0092	0.0	437.0	0.004	1.343	0.5
	BPR^{S}	0.0038	0.0	102.8	0.057	1.073	0.499
	$BPR^{S,u}$	0.0029	0.0	**91.6**	**0.215**	**1.02**	0.498
	BPR^{E}	0.0092	0.0	253.9	0.002	1.357	0.5
	BPR^{Jan}	0.009	0.0	274.8	0.084	1.071	0.5
	BPR^{IPS}	0.0086	0.0	295.3	0.009	1.224	0.5
	$BPR^{b(r)}$	0.0092	0.0	246.7	0.0	1.374	0.5
	BPR^{FD}	0.0109	0.0	426.2	0.008	1.247	0.5

Table 6. Results obtained with *BPR*, by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off *k*@1. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.

6 ACKNOWLEDGMENTS

This work was supported by the Departmental Strategic Plan (PSD) of the University of Udine, Interdepartmental Project on Artificial Intelligence (2020-25). It was also partially supported by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU.



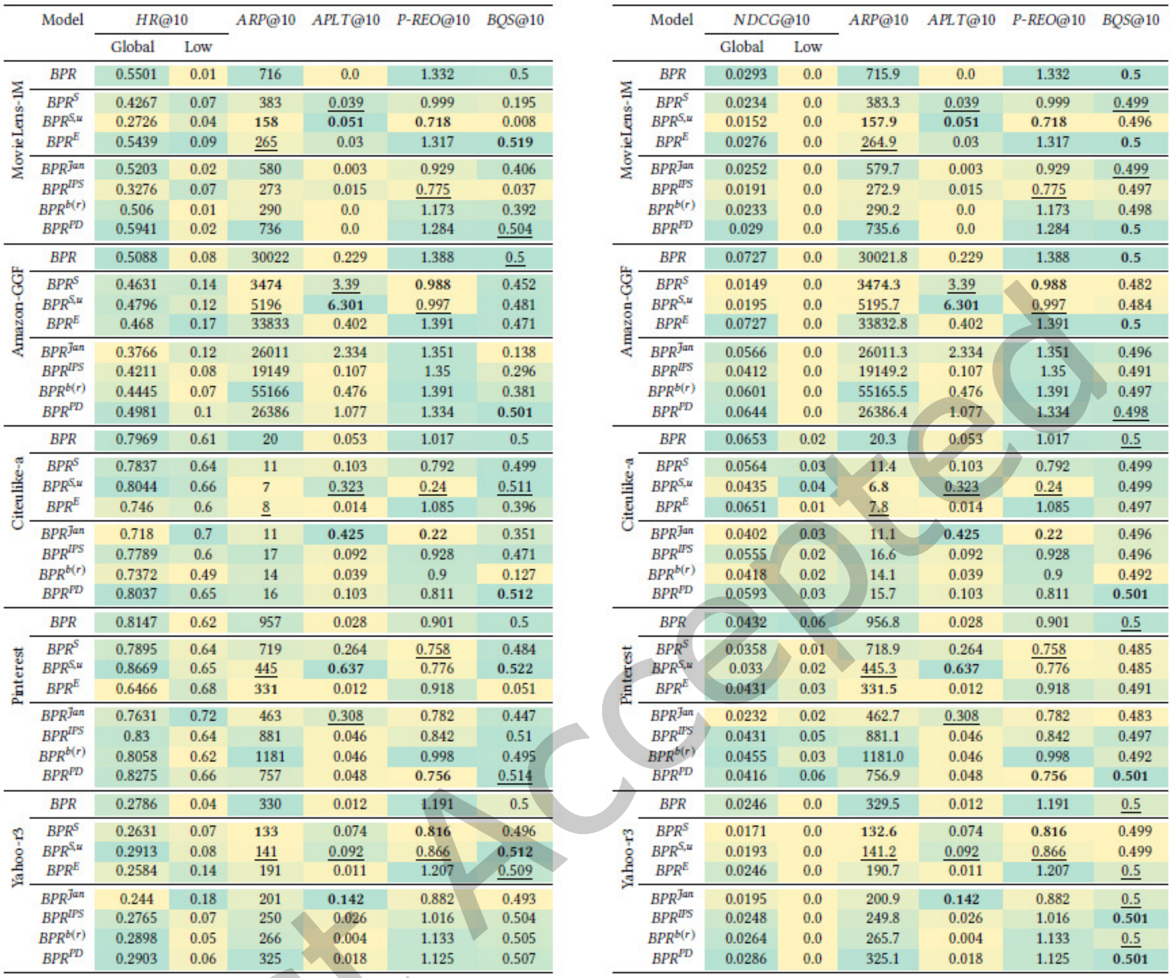
	Model	*HR@5*		*ARP@5*	*APLT@5*	*P-REO@5*	*BQS@5*
		Global	Low				
MovieLens-1M	BPR	0.3582	0.01	787	0.0	1.35	<u>0.5</u>
	BPR^{S}	0.249	0.04	412	<u>0.057</u>	1.078	0.224
	$BPR^{S,u}$	0.1422	0.02	**151**	**0.08**	**0.737**	0.032
	BPR^{E}	0.3531	0.06	<u>285</u>	0.047	1.33	**0.511**
	BPR^{Jan}	0.3171	0.01	736	0.005	0.959	0.393
	BPR^{IPS}	0.181	0.04	295	0.019	<u>0.842</u>	0.084
	$BPR^{b(r)}$	0.2983	0.0	294	0.0	1.214	0.355
	BPR^{PD}	0.3938	0.01	810	0.0	1.308	0.498
Amazon-GGF	BPR	0.4041	0.03	38438	0.142	1.398	**0.5**
	BPR^{S}	0.346	0.07	**3349**	<u>2.548</u>	<u>1.003</u>	0.412
	$BPR^{S,u}$	0.3666	0.06	<u>4798</u>	**6.337**	**0.989**	0.463
	BPR^{E}	0.3809	0.08	43380	0.189	1.4	0.493
	BPR^{Jan}	0.2926	0.07	34923	1.864	1.372	0.21
	BPR^{IPS}	0.3326	0.03	22490	0.061	1.366	0.357
	$BPR^{b(r)}$	0.3452	0.03	73775	0.319	1.4	0.399
	BPR^{PD}	0.3899	0.05	34206	0.844	1.363	<u>0.495</u>
Citeulike-a	BPR	0.6689	0.44	24	0.046	1.108	0.5
	BPR^{S}	0.6677	0.5	12	0.091	0.872	0.512
	$BPR^{S,u}$	0.6943	0.53	**7**	<u>0.387</u>	**0.357**	**0.528**
	BPR^{E}	0.633	0.43	<u>9</u>	0.012	1.182	0.428
	BPR^{Jan}	0.5746	0.57	14	**0.451**	<u>0.476</u>	0.297
	BPR^{IPS}	0.6556	0.44	19	0.093	0.994	0.486
	$BPR^{b(r)}$	0.5968	0.32	15	0.035	0.958	0.1
	BPR^{PD}	0.6708	0.5	18	0.1	0.988	<u>0.513</u>
Pinterest	BPR	0.6361	0.45	1028	0.022	0.961	0.5
	BPR^{S}	0.6107	0.5	740	0.241	0.808	0.491
	$BPR^{S,u}$	0.6951	0.55	<u>419</u>	**0.837**	**0.769**	**0.539**
	BPR^{E}	0.5548	0.5	**356**	0.008	0.981	0.334
	BPR^{Jan}	0.569	0.56	470	<u>0.274</u>	<u>0.774</u>	0.4
	BPR^{IPS}	0.6524	0.49	933	0.041	0.892	0.514
	$BPR^{b(r)}$	0.6287	0.47	1280	0.039	1.033	0.503
	BPR^{PD}	0.6468	0.51	807	0.042	0.81	<u>0.519</u>
Yahoo-r3	BPR	0.1756	0.02	373	0.008	1.247	0.5
	BPR^{S}	0.162	0.03	**122**	0.068	**0.847**	0.496
	$BPR^{S,u}$	0.1901	0.04	<u>129</u>	**0.124**	<u>0.917</u>	**0.51**
	BPR^{E}	0.1692	0.05	216	0.006	1.264	0.505
	BPR^{Jan}	0.1494	0.08	227	<u>0.115</u>	1.004	0.491
	BPR^{IPS}	0.1734	0.03	269	0.021	1.087	0.502
	$BPR^{b(r)}$	0.1857	0.02	259	0.003	1.217	0.503
	BPR^{PD}	0.1858	0.03	367	0.013	1.227	0.505

	Model	*NDCG@5*		*ARP@5*	*APLT@5*	*P-REO@5*	*BQS@5*
		Global	Low				
MovieLens-1M	BPR	0.0218	0.0	787.2	0.0	1.35	<u>0.5</u>
	BPR^{S}	0.0209	0.0	411.7	<u>0.057</u>	1.078	**0.501**
	$BPR^{S,u}$	0.0128	0.0	**150.6**	**0.08**	**0.737**	0.498
	BPR^{E}	0.0197	0.0	<u>284.8</u>	0.047	1.33	<u>0.5</u>
	BPR^{Jan}	0.02	0.0	735.6	0.005	0.959	<u>0.5</u>
	BPR^{IPS}	0.0168	0.0	295.4	0.019	<u>0.842</u>	0.499
	$BPR^{b(r)}$	0.0198	0.0	293.7	0.0	1.214	0.499
	BPR^{PD}	0.0227	0.0	809.6	0.0	1.308	<u>0.5</u>
Amazon-GGF	BPR	0.0636	0.0	38437.6	0.142	1.398	**0.5**
	BPR^{S}	0.0109	0.0	**3349.2**	<u>2.548</u>	<u>1.003</u>	0.484
	$BPR^{S,u}$	0.0139	0.0	<u>4798.2</u>	**6.337**	**0.989**	0.485
	BPR^{E}	0.0636	0.0	43379.7	0.189	1.4	**0.5**
	BPR^{Jan}	0.049	0.0	34923.4	1.864	1.372	0.496
	BPR^{IPS}	0.0325	0.0	22489.8	0.061	1.366	0.491
	$BPR^{b(r)}$	0.0516	0.0	73774.6	0.319	1.4	0.497
	BPR^{PD}	0.0553	0.0	34206.3	0.844	1.363	<u>0.498</u>
Citeulike-a	BPR	0.0594	0.02	23.8	0.046	1.108	**0.5**
	BPR^{S}	0.0499	0.02	12.3	0.091	0.872	<u>0.497</u>
	$BPR^{S,u}$	0.0329	0.02	**6.6**	<u>0.387</u>	**0.357**	0.493
	BPR^{E}	0.0593	0.0	<u>9.2</u>	0.012	1.182	0.494
	BPR^{Jan}	0.0325	0.02	13.5	**0.451**	<u>0.476</u>	0.492
	BPR^{IPS}	0.0502	0.01	18.6	0.093	0.994	0.495
	$BPR^{b(r)}$	0.0361	0.02	15.4	0.035	0.958	0.492
	BPR^{PD}	0.0533	0.01	17.9	0.1	0.988	0.496
Pinterest	BPR	0.0404	0.09	1028.2	0.022	0.961	**0.5**
	BPR^{S}	0.0328	0.02	740.1	0.241	0.808	0.473
	$BPR^{S,u}$	0.0284	0.01	<u>419.1</u>	**0.837**	**0.769**	0.469
	BPR^{E}	0.0404	0.05	**356.3**	0.008	0.981	0.489
	BPR^{Jan}	0.0206	0.02	470.4	<u>0.274</u>	<u>0.774</u>	0.469
	BPR^{IPS}	0.0398	0.06	933.2	0.041	0.892	0.49
	$BPR^{b(r)}$	0.0427	0.04	1280.5	0.039	1.033	0.485
	BPR^{PD}	0.039	0.09	806.5	0.042	0.81	<u>0.499</u>
Yahoo-r3	BPR	0.017	0.0	372.7	0.008	1.247	0.5
	BPR^{S}	0.0107	0.0	**122.0**	0.068	**0.847**	0.498
	$BPR^{S,u}$	0.0099	0.0	<u>129.1</u>	**0.124**	<u>0.917</u>	0.498
	BPR^{E}	0.0169	0.0	215.9	0.006	1.264	0.5
	BPR^{Jan}	0.0119	0.01	226.5	<u>0.115</u>	1.004	0.5
	BPR^{IPS}	0.017	0.0	269.0	0.021	1.087	0.5
	$BPR^{b(r)}$	0.0176	0.0	258.6	0.003	1.217	0.5
	BPR^{PD}	0.0208	0.0	367.0	0.013	1.227	0.501

Table 7. Results obtained with *BPR*, by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off *k*@5. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.

7 AUTHOR CONTRIBUTIONS

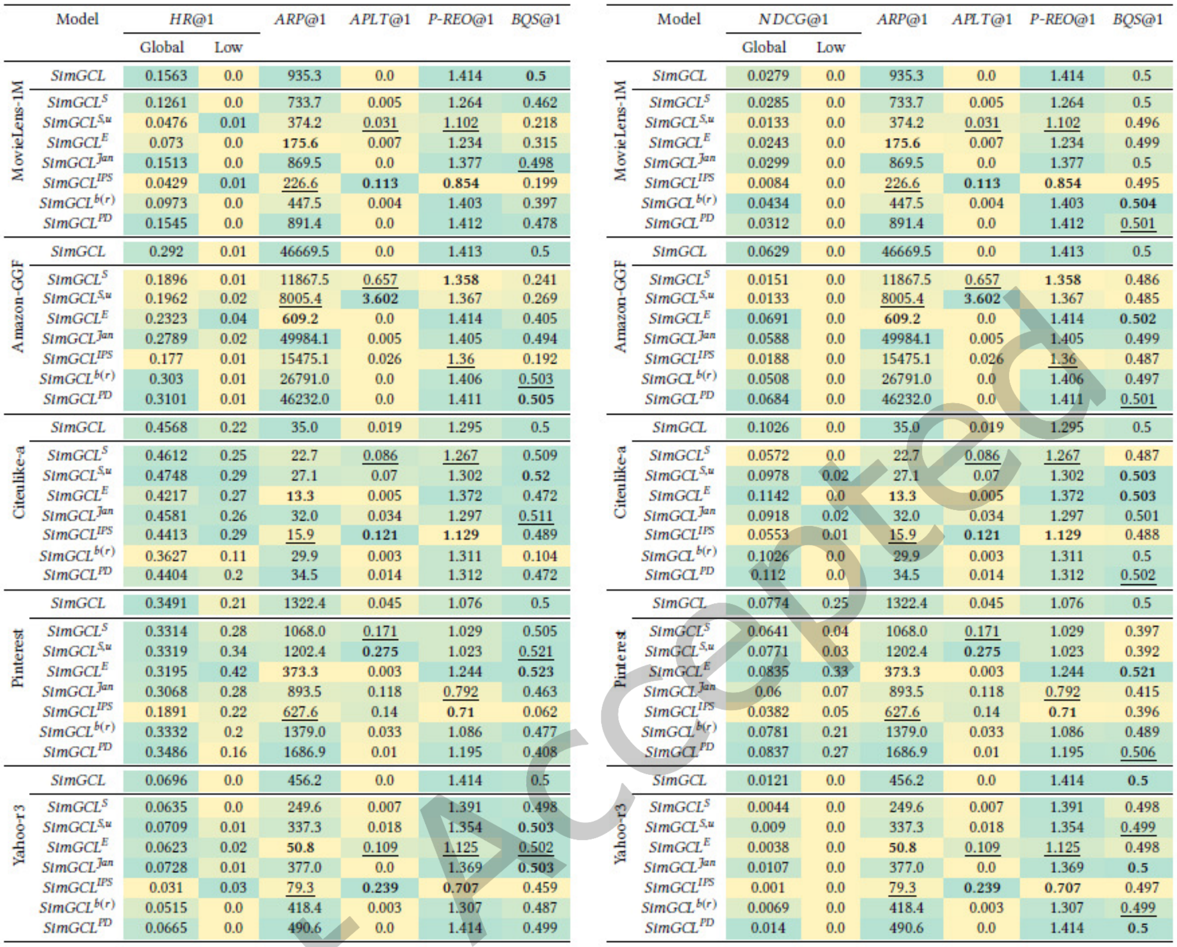
GM and ER conceived the idea underlying the article. EC, MM, LC and FSP developed the software. GM, ER, EC and MM designed the experiments. EC, MM and LC performed the experiments using the provided software. All authors contributed to write the manuscript. EC and MM have to be considered as co-first authors.



	Model	*HR@10*		*ARP@10*	*APLT@10*	*P-REO@10*	*BQS@10*
		Global	Low				
MovieLens-1M	*BPR*	0.5501	0.01	716	0.0	1.332	0.5
	BPR^{S}	0.4267	0.07	383	0.039	0.999	0.195
	$BPR^{S,u}$	0.2726	0.04	**158**	**0.051**	**0.718**	0.008
	BPR^{E}	0.5439	0.09	265	0.03	1.317	**0.519**
	BPR^{Jan}	0.5203	0.02	580	0.003	0.929	0.406
	BPR^{IPS}	0.3276	0.07	273	0.015	0.775	0.037
	$BPR^{b(r)}$	0.506	0.01	290	0.0	1.173	0.392
	BPR^{FD}	0.5941	0.02	736	0.0	1.284	0.504
Amazon-GGF	*BPR*	0.5088	0.08	30022	0.229	1.388	0.5
	BPR^{S}	0.4631	0.14	**3474**	3.39	**0.988**	0.452
	$BPR^{S,u}$	0.4796	0.12	5196	**6.301**	0.997	0.481
	BPR^{E}	0.468	0.17	33833	0.402	1.391	0.471
	BPR^{Jan}	0.3766	0.12	26011	2.334	1.351	0.138
	BPR^{IPS}	0.4211	0.08	19149	0.107	1.35	0.296
	$BPR^{b(r)}$	0.4445	0.07	55166	0.476	1.391	0.381
	BPR^{FD}	0.4981	0.1	26386	1.077	1.334	**0.501**
Citeulike-a	*BPR*	0.7969	0.61	20	0.053	1.017	0.5
	BPR^{S}	0.7837	0.64	11	0.103	0.792	0.499
	$BPR^{S,u}$	0.8044	0.66	**7**	0.323	0.24	0.511
	BPR^{E}	0.746	0.6	8	0.014	1.085	0.396
	BPR^{Jan}	0.718	0.7	11	**0.425**	**0.22**	0.351
	BPR^{IPS}	0.7789	0.6	17	0.092	0.928	0.471
	$BPR^{b(r)}$	0.7372	0.49	14	0.039	0.9	0.127
	BPR^{FD}	0.8037	0.65	16	0.103	0.811	**0.512**
Pinterest	*BPR*	0.8147	0.62	957	0.028	0.901	0.5
	BPR^{S}	0.7895	0.64	719	0.264	0.758	0.484
	$BPR^{S,u}$	0.8669	0.65	445	**0.637**	0.776	**0.522**
	BPR^{E}	0.6466	0.68	**331**	0.012	0.918	0.051
	BPR^{Jan}	0.7631	0.72	463	0.308	0.782	0.447
	BPR^{IPS}	0.83	0.64	881	0.046	0.842	0.51
	$BPR^{b(r)}$	0.8058	0.62	1181	0.046	0.998	0.495
	BPR^{FD}	0.8275	0.66	757	0.048	**0.756**	0.514
Yahoo-r3	*BPR*	0.2786	0.04	330	0.012	1.191	0.5
	BPR^{S}	0.2631	0.07	**133**	0.074	**0.816**	0.496
	$BPR^{S,u}$	0.2913	0.08	141	0.092	0.866	**0.512**
	BPR^{E}	0.2584	0.14	191	0.011	1.207	0.509
	BPR^{Jan}	0.244	0.18	201	**0.142**	0.882	0.493
	BPR^{IPS}	0.2765	0.07	250	0.026	1.016	0.504
	$BPR^{b(r)}$	0.2898	0.05	266	0.004	1.133	0.505
	BPR^{FD}	0.2903	0.06	325	0.018	1.125	0.507

	Model	*NDCG@10*		*ARP@10*	*APLT@10*	*P-REO@10*	*BQS@10*
		Global	Low				
MovieLens-1M	*BPR*	0.0293	0.0	715.9	0.0	1.332	**0.5**
	BPR^{S}	0.0234	0.0	383.3	0.039	0.999	0.499
	$BPR^{S,u}$	0.0152	0.0	**157.9**	**0.051**	**0.718**	0.496
	BPR^{E}	0.0276	0.0	264.9	0.03	1.317	**0.5**
	BPR^{Jan}	0.0252	0.0	579.7	0.003	0.929	0.499
	BPR^{IPS}	0.0191	0.0	272.9	0.015	0.775	0.497
	$BPR^{b(r)}$	0.0233	0.0	290.2	0.0	1.173	0.498
	BPR^{FD}	0.029	0.0	735.6	0.0	1.284	**0.5**
Amazon-GGF	*BPR*	0.0727	0.0	30021.8	0.229	1.388	**0.5**
	BPR^{S}	0.0149	0.0	**3474.3**	3.39	**0.988**	0.482
	$BPR^{S,u}$	0.0195	0.0	5195.7	**6.301**	0.997	0.484
	BPR^{E}	0.0727	0.0	33832.8	0.402	1.391	**0.5**
	BPR^{Jan}	0.0566	0.0	26011.3	2.334	1.351	0.496
	BPR^{IPS}	0.0412	0.0	19149.2	0.107	1.35	0.491
	$BPR^{b(r)}$	0.0601	0.0	55165.5	0.476	1.391	0.497
	BPR^{FD}	0.0644	0.0	26386.4	1.077	1.334	0.498
Citeulike-a	*BPR*	0.0653	0.02	20.3	0.053	1.017	0.5
	BPR^{S}	0.0564	0.03	11.4	0.103	0.792	0.499
	$BPR^{S,u}$	0.0435	0.04	**6.8**	0.323	0.24	0.499
	BPR^{E}	0.0651	0.01	7.8	0.014	1.085	0.497
	BPR^{Jan}	0.0402	0.03	11.1	**0.425**	**0.22**	0.496
	BPR^{IPS}	0.0555	0.02	16.6	0.092	0.928	0.496
	$BPR^{b(r)}$	0.0418	0.02	14.1	0.039	0.9	0.492
	BPR^{FD}	0.0593	0.03	15.7	0.103	0.811	**0.501**
Pinterest	*BPR*	0.0432	0.06	956.8	0.028	0.901	0.5
	BPR^{S}	0.0358	0.01	718.9	0.264	0.758	0.485
	$BPR^{S,u}$	0.033	0.02	445.3	**0.637**	0.776	0.485
	BPR^{E}	0.0431	0.03	**331.5**	0.012	0.918	0.491
	BPR^{Jan}	0.0232	0.02	462.7	0.308	0.782	0.483
	BPR^{IPS}	0.0431	0.05	881.1	0.046	0.842	0.497
	$BPR^{b(r)}$	0.0455	0.03	1181.0	0.046	0.998	0.492
	BPR^{FD}	0.0416	0.06	756.9	0.048	**0.756**	**0.501**
Yahoo-r3	*BPR*	0.0246	0.0	329.5	0.012	1.191	0.5
	BPR^{S}	0.0171	0.0	**132.6**	0.074	**0.816**	0.499
	$BPR^{S,u}$	0.0193	0.0	141.2	0.092	0.866	0.499
	BPR^{E}	0.0246	0.0	190.7	0.011	1.207	0.5
	BPR^{Jan}	0.0195	0.0	200.9	**0.142**	0.882	0.5
	BPR^{IPS}	0.0248	0.0	249.8	0.026	1.016	**0.501**
	$BPR^{b(r)}$	0.0264	0.0	265.7	0.004	1.133	0.5
	BPR^{FD}	0.0286	0.0	325.1	0.018	1.125	**0.501**

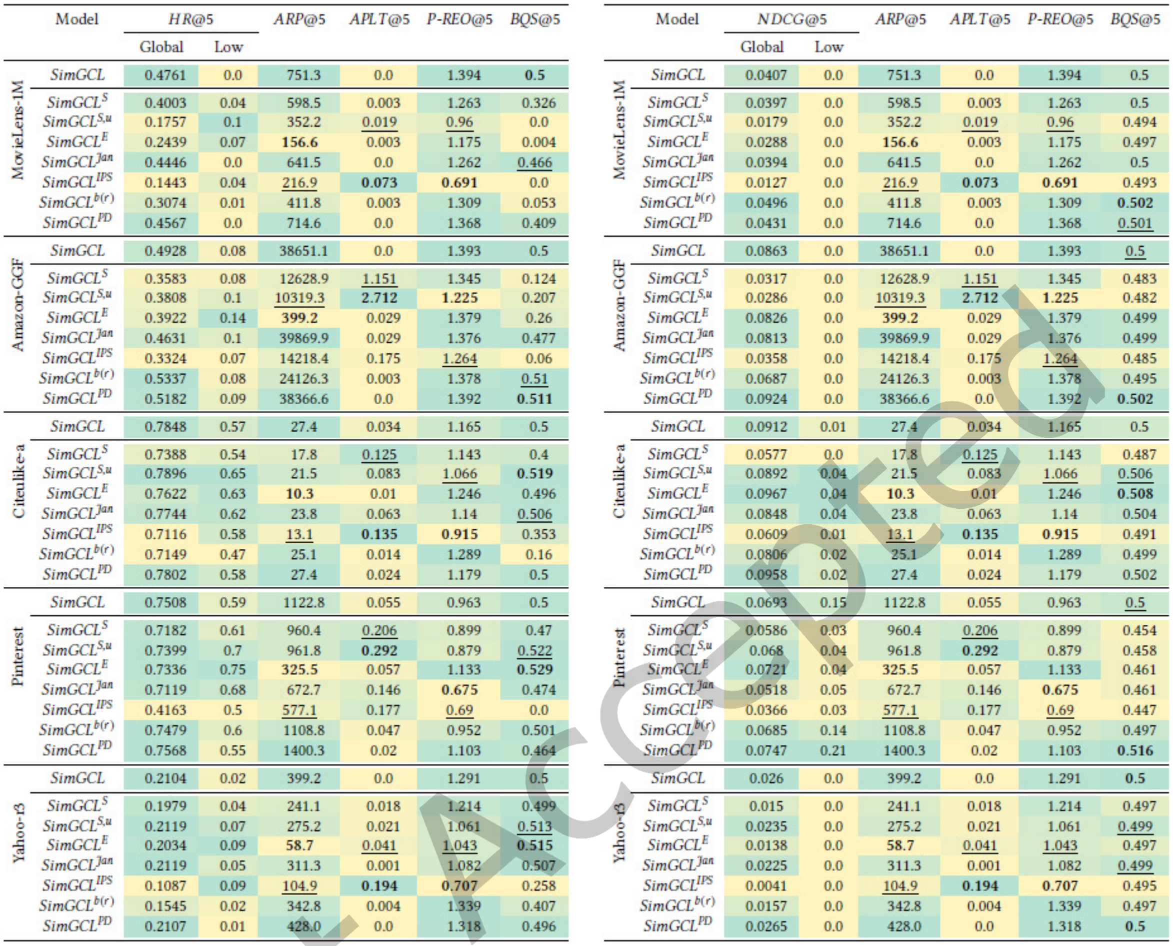
Table 8. Results obtained with *BPR*, by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off *k*@10. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.



	Model	*HR@1*		*ARP@1*	*APLT@1*	*P-REO@1*	*BQS@1*
		Global	Low				
MovieLens-1M	*SimGCL*	0.1563	0.0	935.3	0.0	1.414	**0.5**
	$SimGCL^{S}$	0.1261	0.0	733.7	0.005	1.264	0.462
	$SimGCL^{S,u}$	0.0476	0.01	374.2	0.031	1.102	0.218
	$SimGCL^{E}$	0.073	0.0	**175.6**	0.007	1.234	0.315
	$SimGCL^{Jan}$	0.1513	0.0	869.5	0.0	1.377	0.498
	$SimGCL^{IPS}$	0.0429	0.01	226.6	**0.113**	**0.854**	0.199
	$SimGCL^{b(r)}$	0.0973	0.0	447.5	0.004	1.403	0.397
	$SimGCL^{PD}$	0.1545	0.0	891.4	0.0	1.412	0.478
Amazon-GGF	*SimGCL*	0.292	0.01	46669.5	0.0	1.413	0.5
	$SimGCL^{S}$	0.1896	0.01	11867.5	0.657	**1.358**	0.241
	$SimGCL^{S,u}$	0.1962	0.02	8005.4	**3.602**	1.367	0.269
	$SimGCL^{E}$	0.2323	0.04	**609.2**	0.0	1.414	0.405
	$SimGCL^{Jan}$	0.2789	0.02	49984.1	0.005	1.405	0.494
	$SimGCL^{IPS}$	0.177	0.01	15475.1	0.026	1.36	0.192
	$SimGCL^{b(r)}$	0.303	0.01	26791.0	0.0	1.406	0.503
	$SimGCL^{PD}$	0.3101	0.01	46232.0	0.0	1.411	**0.505**
Citeulike-a	*SimGCL*	0.4568	0.22	35.0	0.019	1.295	0.5
	$SimGCL^{S}$	0.4612	0.25	22.7	0.086	1.267	0.509
	$SimGCL^{S,u}$	0.4748	0.29	27.1	0.07	1.302	**0.52**
	$SimGCL^{E}$	0.4217	0.27	**13.3**	0.005	1.372	0.472
	$SimGCL^{Jan}$	0.4581	0.26	32.0	0.034	1.297	0.511
	$SimGCL^{IPS}$	0.4413	0.29	15.9	**0.121**	**1.129**	0.489
	$SimGCL^{b(r)}$	0.3627	0.11	29.9	0.003	1.311	0.104
	$SimGCL^{PD}$	0.4404	0.2	34.5	0.014	1.312	0.472
Pinterest	*SimGCL*	0.3491	0.21	1322.4	0.045	1.076	0.5
	$SimGCL^{S}$	0.3314	0.28	1068.0	0.171	1.029	0.505
	$SimGCL^{S,u}$	0.3319	0.34	1202.4	**0.275**	1.023	0.521
	$SimGCL^{E}$	0.3195	0.42	**373.3**	0.003	1.244	**0.523**
	$SimGCL^{Jan}$	0.3068	0.28	893.5	0.118	0.792	0.463
	$SimGCL^{IPS}$	0.1891	0.22	627.6	0.14	**0.71**	0.062
	$SimGCL^{b(r)}$	0.3332	0.2	1379.0	0.033	1.086	0.477
	$SimGCL^{PD}$	0.3486	0.16	1686.9	0.01	1.195	0.408
Yahoo-r3	*SimGCL*	0.0696	0.0	456.2	0.0	1.414	0.5
	$SimGCL^{S}$	0.0635	0.0	249.6	0.007	1.391	0.498
	$SimGCL^{S,u}$	0.0709	0.01	337.3	0.018	1.354	**0.503**
	$SimGCL^{E}$	0.0623	0.02	**50.8**	0.109	1.125	0.502
	$SimGCL^{Jan}$	0.0728	0.01	377.0	0.0	1.369	**0.503**
	$SimGCL^{IPS}$	0.031	0.03	79.3	**0.239**	**0.707**	0.459
	$SimGCL^{b(r)}$	0.0515	0.0	418.4	0.003	1.307	0.487
	$SimGCL^{PD}$	0.0665	0.0	490.6	0.0	1.414	0.499

	Model	*NDCG@1*		*ARP@1*	*APLT@1*	*P-REO@1*	*BQS@1*
		Global	Low				
MovieLens-1M	*SimGCL*	0.0279	0.0	935.3	0.0	1.414	0.5
	$SimGCL^{S}$	0.0285	0.0	733.7	0.005	1.264	0.5
	$SimGCL^{S,u}$	0.0133	0.0	374.2	0.031	1.102	0.496
	$SimGCL^{E}$	0.0243	0.0	**175.6**	0.007	1.234	0.499
	$SimGCL^{Jan}$	0.0299	0.0	869.5	0.0	1.377	0.5
	$SimGCL^{IPS}$	0.0084	0.0	226.6	**0.113**	**0.854**	0.495
	$SimGCL^{b(r)}$	0.0434	0.0	447.5	0.004	1.403	**0.504**
	$SimGCL^{PD}$	0.0312	0.0	891.4	0.0	1.412	0.501
Amazon-GGF	*SimGCL*	0.0629	0.0	46669.5	0.0	1.413	0.5
	$SimGCL^{S}$	0.0151	0.0	11867.5	0.657	**1.358**	0.486
	$SimGCL^{S,u}$	0.0133	0.0	8005.4	**3.602**	1.367	0.485
	$SimGCL^{E}$	0.0691	0.0	**609.2**	0.0	1.414	**0.502**
	$SimGCL^{Jan}$	0.0588	0.0	49984.1	0.005	1.405	0.499
	$SimGCL^{IPS}$	0.0188	0.0	15475.1	0.026	1.36	0.487
	$SimGCL^{b(r)}$	0.0508	0.0	26791.0	0.0	1.406	0.497
	$SimGCL^{PD}$	0.0684	0.0	46232.0	0.0	1.411	0.501
Citeulike-a	*SimGCL*	0.1026	0.0	35.0	0.019	1.295	0.5
	$SimGCL^{S}$	0.0572	0.0	22.7	0.086	1.267	0.487
	$SimGCL^{S,u}$	0.0978	0.02	27.1	0.07	1.302	**0.503**
	$SimGCL^{E}$	0.1142	0.0	**13.3**	0.005	1.372	**0.503**
	$SimGCL^{Jan}$	0.0918	0.02	32.0	0.034	1.297	0.501
	$SimGCL^{IPS}$	0.0553	0.01	15.9	**0.121**	**1.129**	0.488
	$SimGCL^{b(r)}$	0.1026	0.0	29.9	0.003	1.311	0.5
	$SimGCL^{PD}$	0.112	0.0	34.5	0.014	1.312	0.502
Pinterest	*SimGCL*	0.0774	0.25	1322.4	0.045	1.076	0.5
	$SimGCL^{S}$	0.0641	0.04	1068.0	0.171	1.029	0.397
	$SimGCL^{S,u}$	0.0771	0.03	1202.4	**0.275**	1.023	0.392
	$SimGCL^{E}$	0.0835	0.33	**373.3**	0.003	1.244	**0.521**
	$SimGCL^{Jan}$	0.06	0.07	893.5	0.118	0.792	0.415
	$SimGCL^{IPS}$	0.0382	0.05	627.6	0.14	**0.71**	0.396
	$SimGCL^{b(r)}$	0.0781	0.21	1379.0	0.033	1.086	0.489
	$SimGCL^{PD}$	0.0837	0.27	1686.9	0.01	1.195	0.506
Yahoo-r3	*SimGCL*	0.0121	0.0	456.2	0.0	1.414	**0.5**
	$SimGCL^{S}$	0.0044	0.0	249.6	0.007	1.391	0.498
	$SimGCL^{S,u}$	0.009	0.0	337.3	0.018	1.354	0.499
	$SimGCL^{E}$	0.0038	0.0	**50.8**	0.109	1.125	0.498
	$SimGCL^{Jan}$	0.0107	0.0	377.0	0.0	1.369	**0.5**
	$SimGCL^{IPS}$	0.001	0.0	79.3	**0.239**	**0.707**	0.497
	$SimGCL^{b(r)}$	0.0069	0.0	418.4	0.003	1.307	0.499
	$SimGCL^{PD}$	0.014	0.0	490.6	0.0	1.414	**0.5**

Table 9. Results obtained with *SimGCL* by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off *k*@1. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.



	Model	*HR@5*		*ARP@5*	*APLT@5*	*P-REO@5*	*BQS@5*
		Global	Low				
MovieLens-1M	$SimGCL$	0.4761	0.0	751.3	0.0	1.394	**0.5**
	$SimGCL^{S}$	0.4003	0.04	598.5	0.003	1.263	0.326
	$SimGCL^{S,u}$	0.1757	0.1	352.2	0.019	0.96	0.0
	$SimGCL^{E}$	0.2439	0.07	**156.6**	0.003	1.175	0.004
	$SimGCL^{Jan}$	0.4446	0.0	641.5	0.0	1.262	0.466
	$SimGCL^{IPS}$	0.1443	0.04	216.9	**0.073**	**0.691**	0.0
	$SimGCL^{b(r)}$	0.3074	0.01	411.8	0.003	1.309	0.053
	$SimGCL^{PD}$	0.4567	0.0	714.6	0.0	1.368	0.409
Amazon-GGF	$SimGCL$	0.4928	0.08	38651.1	0.0	1.393	0.5
	$SimGCL^{S}$	0.3583	0.08	12628.9	1.151	1.345	0.124
	$SimGCL^{S,u}$	0.3808	0.1	10319.3	**2.712**	**1.225**	0.207
	$SimGCL^{E}$	0.3922	0.14	**399.2**	0.029	1.379	0.26
	$SimGCL^{Jan}$	0.4631	0.1	39869.9	0.029	1.376	0.477
	$SimGCL^{IPS}$	0.3324	0.07	14218.4	0.175	1.264	0.06
	$SimGCL^{b(r)}$	0.5337	0.08	24126.3	0.003	1.378	0.51
	$SimGCL^{PD}$	0.5182	0.09	38366.6	0.0	1.392	**0.511**
Citeulike-a	$SimGCL$	0.7848	0.57	27.4	0.034	1.165	0.5
	$SimGCL^{S}$	0.7388	0.54	17.8	0.125	1.143	0.4
	$SimGCL^{S,u}$	0.7896	0.65	21.5	0.083	1.066	**0.519**
	$SimGCL^{E}$	0.7622	0.63	**10.3**	0.01	1.246	0.496
	$SimGCL^{Jan}$	0.7744	0.62	23.8	0.063	1.14	0.506
	$SimGCL^{IPS}$	0.7116	0.58	13.1	**0.135**	**0.915**	0.353
	$SimGCL^{b(r)}$	0.7149	0.47	25.1	0.014	1.289	0.16
	$SimGCL^{PD}$	0.7802	0.58	27.4	0.024	1.179	0.5
Pinterest	$SimGCL$	0.7508	0.59	1122.8	0.055	0.963	0.5
	$SimGCL^{S}$	0.7182	0.61	960.4	0.206	0.899	0.47
	$SimGCL^{S,u}$	0.7399	0.7	961.8	**0.292**	0.879	0.522
	$SimGCL^{E}$	0.7336	0.75	**325.5**	0.057	1.133	**0.529**
	$SimGCL^{Jan}$	0.7119	0.68	672.7	0.146	**0.675**	0.474
	$SimGCL^{IPS}$	0.4163	0.5	577.1	0.177	0.69	0.0
	$SimGCL^{b(r)}$	0.7479	0.6	1108.8	0.047	0.952	0.501
	$SimGCL^{PD}$	0.7568	0.55	1400.3	0.02	1.103	0.464
Yahoo-r3	$SimGCL$	0.2104	0.02	399.2	0.0	1.291	0.5
	$SimGCL^{S}$	0.1979	0.04	241.1	0.018	1.214	0.499
	$SimGCL^{S,u}$	0.2119	0.07	275.2	0.021	1.061	0.513
	$SimGCL^{E}$	0.2034	0.09	**58.7**	0.041	1.043	**0.515**
	$SimGCL^{Jan}$	0.2119	0.05	311.3	0.001	1.082	0.507
	$SimGCL^{IPS}$	0.1087	0.09	104.9	**0.194**	**0.707**	0.258
	$SimGCL^{b(r)}$	0.1545	0.02	342.8	0.004	1.339	0.407
	$SimGCL^{PD}$	0.2107	0.01	428.0	0.0	1.318	0.496

	Model	*NDCG@5*		*ARP@5*	*APLT@5*	*P-REO@5*	*BQS@5*
		Global	Low				
MovieLens-1M	$SimGCL$	0.0407	0.0	751.3	0.0	1.394	0.5
	$SimGCL^{S}$	0.0397	0.0	598.5	0.003	1.263	0.5
	$SimGCL^{S,u}$	0.0179	0.0	352.2	0.019	0.96	0.494
	$SimGCL^{E}$	0.0288	0.0	**156.6**	0.003	1.175	0.497
	$SimGCL^{Jan}$	0.0394	0.0	641.5	0.0	1.262	0.5
	$SimGCL^{IPS}$	0.0127	0.0	216.9	**0.073**	**0.691**	0.493
	$SimGCL^{b(r)}$	0.0496	0.0	411.8	0.003	1.309	**0.502**
	$SimGCL^{PD}$	0.0431	0.0	714.6	0.0	1.368	0.501
Amazon-GGF	$SimGCL$	0.0863	0.0	38651.1	0.0	1.393	0.5
	$SimGCL^{S}$	0.0317	0.0	12628.9	1.151	1.345	0.483
	$SimGCL^{S,u}$	0.0286	0.0	10319.3	**2.712**	**1.225**	0.482
	$SimGCL^{E}$	0.0826	0.0	**399.2**	0.029	1.379	0.499
	$SimGCL^{Jan}$	0.0813	0.0	39869.9	0.029	1.376	0.499
	$SimGCL^{IPS}$	0.0358	0.0	14218.4	0.175	1.264	0.485
	$SimGCL^{b(r)}$	0.0687	0.0	24126.3	0.003	1.378	0.495
	$SimGCL^{PD}$	0.0924	0.0	38366.6	0.0	1.392	**0.502**
Citeulike-a	$SimGCL$	0.0912	0.01	27.4	0.034	1.165	0.5
	$SimGCL^{S}$	0.0577	0.0	17.8	0.125	1.143	0.487
	$SimGCL^{S,u}$	0.0892	0.04	21.5	0.083	1.066	0.506
	$SimGCL^{E}$	0.0967	0.04	**10.3**	0.01	1.246	**0.508**
	$SimGCL^{Jan}$	0.0848	0.04	23.8	0.063	1.14	0.504
	$SimGCL^{IPS}$	0.0609	0.01	13.1	**0.135**	**0.915**	0.491
	$SimGCL^{b(r)}$	0.0806	0.02	25.1	0.014	1.289	0.499
	$SimGCL^{PD}$	0.0958	0.02	27.4	0.024	1.179	0.502
Pinterest	$SimGCL$	0.0693	0.15	1122.8	0.055	0.963	0.5
	$SimGCL^{S}$	0.0586	0.03	960.4	0.206	0.899	0.454
	$SimGCL^{S,u}$	0.068	0.04	961.8	**0.292**	0.879	0.458
	$SimGCL^{E}$	0.0721	0.04	**325.5**	0.057	1.133	0.461
	$SimGCL^{Jan}$	0.0518	0.05	672.7	0.146	**0.675**	0.461
	$SimGCL^{IPS}$	0.0366	0.03	577.1	0.177	0.69	0.447
	$SimGCL^{b(r)}$	0.0685	0.14	1108.8	0.047	0.952	0.497
	$SimGCL^{PD}$	0.0747	0.21	1400.3	0.02	1.103	**0.516**
Yahoo-r3	$SimGCL$	0.026	0.0	399.2	0.0	1.291	**0.5**
	$SimGCL^{S}$	0.015	0.0	241.1	0.018	1.214	0.497
	$SimGCL^{S,u}$	0.0235	0.0	275.2	0.021	1.061	0.499
	$SimGCL^{E}$	0.0138	0.0	**58.7**	0.041	1.043	0.497
	$SimGCL^{Jan}$	0.0225	0.0	311.3	0.001	1.082	0.499
	$SimGCL^{IPS}$	0.0041	0.0	104.9	**0.194**	**0.707**	0.495
	$SimGCL^{b(r)}$	0.0157	0.0	342.8	0.004	1.339	0.497
	$SimGCL^{PD}$	0.0265	0.0	428.0	0.0	1.318	**0.5**

Table 10. Results obtained with *SimGCL* by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off *k*@5. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.

	Model	*HR*@10		*ARP*@10	*APLT*@10	*P-REO*@10	*BQS*@10
		Global	Low				
MovieLens-1M	*SimGCL*	0.6607	0.0	680.3	0.0	1.362	**0.5**
	SimGCL^{S}	0.5838	0.07	541.7	0.002	1.255	0.328
	SimGCL^{S,u}	0.2809	0.22	329.4	0.015	0.817	0.0
	SimGCL^{E}	0.3892	0.26	**145.0**	0.003	1.114	0.001
	SimGCL^{Jan}	0.6285	0.02	562.7	0.0	1.249	0.468
	SimGCL^{IPS}	0.2236	0.09	209.6	**0.068**	**0.689**	0.0
	SimGCL$^{b(r)}$	0.4527	0.03	376.1	0.003	1.283	0.017
	SimGCL^{PD}	0.6327	0.01	644.6	0.0	1.336	0.405
Amazon-GGF	*SimGCL*	0.5842	0.17	29591.3	0.007	1.366	0.5
	SimGCL^{S}	0.4521	0.16	11208.6	1.019	1.293	0.129
	SimGCL^{S,u}	0.4815	0.18	9725.0	**1.848**	**1.161**	0.241
	SimGCL^{E}	0.4836	0.22	**293.3**	0.03	1.322	0.256
	SimGCL^{Jan}	0.5532	0.21	29696.3	0.062	1.339	0.478
	SimGCL^{IPS}	0.422	0.14	12662.4	0.379	1.202	0.052
	SimGCL$^{b(r)}$	0.6321	0.16	21002.0	0.006	1.354	0.507
	SimGCL^{PD}	0.6083	0.19	29539.5	0.002	1.367	**0.511**
Citeulike-a	*SimGCL*	0.8832	0.75	23.0	0.044	1.026	0.5
	SimGCL^{S}	0.8318	0.69	15.6	0.129	0.94	0.323
	SimGCL^{S,u}	0.8771	0.77	18.3	0.09	0.921	**0.504**
	SimGCL^{E}	0.8726	0.78	**8.7**	0.013	1.117	0.503
	SimGCL^{Jan}	0.8706	0.77	19.7	0.077	0.956	0.499
	SimGCL^{IPS}	0.8027	0.71	11.9	**0.134**	**0.732**	0.279
	SimGCL$^{b(r)}$	0.8525	0.7	22.2	0.019	1.204	0.38
	SimGCL^{PD}	0.8812	0.75	23.0	0.033	1.043	0.501
Pinterest	*SimGCL*	0.8887	0.73	1016.4	0.062	0.878	0.5
	SimGCL^{S}	0.8599	0.74	899.2	0.216	0.831	0.474
	SimGCL^{S,u}	0.8885	0.8	850.2	**0.301**	0.778	0.518
	SimGCL^{E}	0.8852	0.84	**295.0**	0.091	1.043	**0.527**
	SimGCL^{Jan}	0.8656	0.8	604.8	0.166	**0.683**	0.5
	SimGCL^{IPS}	0.5383	0.64	555.6	0.2	0.689	0.0
	SimGCL$^{b(r)}$	0.8949	0.74	976.7	0.055	0.843	0.503
	SimGCL^{PD}	0.8906	0.72	1245.9	0.025	1.014	0.494
Yahoo-r3	*SimGCL*	0.3321	0.04	355.9	0.001	1.148	0.5
	SimGCL^{S}	0.3045	0.1	231.6	0.023	0.952	0.489
	SimGCL^{S,u}	0.3231	0.15	236.9	0.03	0.901	0.522
	SimGCL^{E}	0.3112	0.2	**61.3**	0.031	0.965	**0.524**
	SimGCL^{Jan}	0.3246	0.11	274.9	0.003	0.951	0.513
	SimGCL^{IPS}	0.1823	0.15	99.0	**0.234**	**0.597**	0.095
	SimGCL$^{b(r)}$	0.2405	0.07	278.7	0.008	1.195	0.289
	SimGCL^{PD}	0.3319	0.04	377.8	0.0	1.233	0.497

	Model	*NDCG*@10		*ARP*@10	*APLT*@10	*P-REO*@10	*BQS*@10
		Global	Low				
MovieLens-1M	*SimGCL*	0.0529	0.0	680.3	0.0	1.362	0.5
	SimGCL^{S}	0.0493	0.0	541.7	0.002	1.255	0.499
	SimGCL^{S,u}	0.0212	0.0	329.4	0.015	0.817	0.491
	SimGCL^{E}	0.0336	0.0	**145.0**	0.003	1.114	0.495
	SimGCL^{Jan}	0.0521	0.0	562.7	0.0	1.249	0.5
	SimGCL^{IPS}	0.018	0.0	209.6	**0.068**	**0.689**	0.491
	SimGCL$^{b(r)}$	0.0542	0.0	376.1	0.003	1.283	0.5
	SimGCL^{PD}	0.0565	0.0	644.6	0.0	1.336	**0.501**
Amazon-GGF	*SimGCL*	0.0992	0.0	29591.3	0.007	1.366	0.5
	SimGCL^{S}	0.0418	0.0	11208.6	1.019	1.293	0.482
	SimGCL^{S,u}	0.0391	0.0	9725.0	**1.848**	**1.161**	0.481
	SimGCL^{E}	0.0939	0.0	**293.3**	0.03	1.322	0.499
	SimGCL^{Jan}	0.0937	0.0	29696.3	0.062	1.339	0.499
	SimGCL^{IPS}	0.045	0.0	12662.4	0.379	1.202	0.484
	SimGCL$^{b(r)}$	0.082	0.0	21002.0	0.006	1.354	0.495
	SimGCL^{PD}	0.1056	0.0	29539.5	0.002	1.367	**0.502**
Citeulike-a	*SimGCL*	0.1021	0.05	23.0	0.044	1.026	0.5
	SimGCL^{S}	0.0724	0.02	15.6	0.129	0.94	0.484
	SimGCL^{S,u}	0.1024	0.06	18.3	0.09	0.921	0.503
	SimGCL^{E}	0.1036	0.05	**8.7**	0.013	1.117	0.501
	SimGCL^{Jan}	0.0965	0.06	19.7	0.077	0.956	0.501
	SimGCL^{IPS}	0.0768	0.03	11.9	**0.134**	**0.732**	0.489
	SimGCL$^{b(r)}$	0.0878	0.04	22.2	0.019	1.204	0.494
	SimGCL^{PD}	0.104	0.06	23.0	0.033	1.043	**0.504**
Pinterest	*SimGCL*	0.073	0.12	1016.4	0.062	0.878	0.5
	SimGCL^{S}	0.0629	0.03	899.2	0.216	0.831	0.464
	SimGCL^{S,u}	0.0712	0.03	850.2	**0.301**	0.778	0.468
	SimGCL^{E}	0.0741	0.03	**295.0**	0.091	1.043	0.467
	SimGCL^{Jan}	0.0553	0.05	604.8	0.166	**0.683**	0.469
	SimGCL^{IPS}	0.0404	0.03	555.6	0.2	0.689	0.457
	SimGCL$^{b(r)}$	0.0721	0.15	976.7	0.055	0.843	0.505
	SimGCL^{PD}	0.0774	0.25	1245.9	0.025	1.014	**0.532**
Yahoo-r3	*SimGCL*	0.0375	0.0	355.9	0.001	1.148	**0.5**
	SimGCL^{S}	0.0263	0.0	231.6	0.023	0.952	0.497
	SimGCL^{S,u}	0.0367	0.0	236.9	0.03	0.901	**0.5**
	SimGCL^{E}	0.0279	0.0	**61.3**	0.031	0.965	0.498
	SimGCL^{Jan}	0.0363	0.0	274.9	0.003	0.951	**0.5**
	SimGCL^{IPS}	0.0105	0.0	99.0	**0.234**	**0.597**	0.493
	SimGCL$^{b(r)}$	0.0234	0.0	278.7	0.008	1.195	0.496
	SimGCL^{PD}	0.0372	0.0	377.8	0.0	1.233	**0.5**

Table 11. Results obtained with *SimGCL* by comparing either *HR* and HR_L (left Table), or *NDCG* and $NDCG_L$ (right Table), as well as *ARP*, *APLT*, *P-REO* and *BQS* at cut-off *k*@10. Colors refer to the column values: the darker the cell, the higher the content. Bold and underline values show the best and second-best results, respectively. All the metrics have been computed by averaging five different runs and applying the ANOVA statistics test. No bold nor underlined values mean differences are not statistically relevant.

A APPENDIX

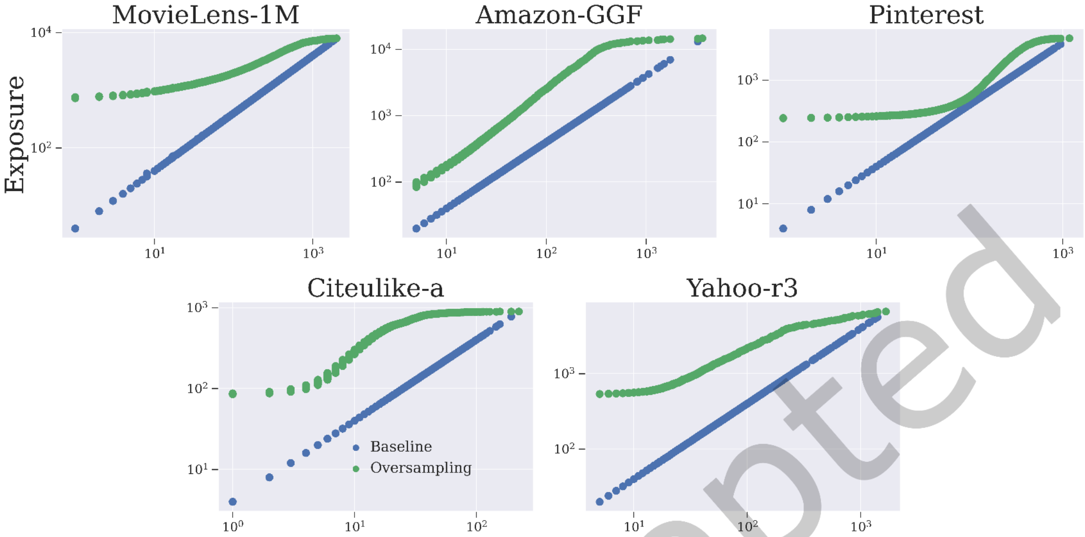


Fig. 4. Item exposure during training. X axis is the item popularity. Blue points are the exposures within the baseline, while green points represent the oversampling induced by Eq. 19. Both axes are on log-scale.

Dynamic Oversampling. The oversampling strategy consists in populating $\mathcal{D}_u$ by progressively increasing the exposure of positive items inversely to their popularity. Hence, rather than sampling, for each occurrence $x_{u,i} = 1$ in $\mathbf{X}$, a fixed number n of negative items, we can apply a stratified sampling scheme.

Let n'_i be a term that is inversely proportional to the popularity of the item, defined as:

$$n'_i = n_0 \frac{max(\boldsymbol{\rho})}{\rho_i \, d_i}\,, \tag{19}$$

where n_0 is a constant (we set it equal to 4), $\boldsymbol{\rho}$ is the popularity distribution of all the items in I, ρ_i is the popularity of the item i, and d_i is the discrete scaling factor that controls the sampling exposure. The latter is worthy of an in-depth discussion.

Consider the term $\frac{max(\rho)}{\rho_i}$: its approximation represents the under-exposure of an item i with respect to the most popular one(s). The adoption of this scaling factor could in principle rebalance the exposures. The adjusted cumulative exposure for item i, i.e. each time $x_{u,i} = 1$ for all $u \in U$ would become in fact:

$$\begin{aligned} exposure_i &= n_0 \sum_{u \in U} x_{u,i} \frac{max(\boldsymbol{\rho})}{\rho_i} = n_0 \frac{max(\boldsymbol{\rho})}{\rho_i} \sum_{u \in U} x_{u,i} \\ &= n_0 \frac{max(\boldsymbol{\rho})}{\rho_i} \rho_i = n_0 \; max(\boldsymbol{\rho})\,. \end{aligned} \tag{20}$$

This would result in all positive items associated with the same number of pairwise comparisons, regardless of their popularity (i.e., in the uniform oversampling strategy).

We can further investigate the rebalancing capability of the term d_i, by introducing a dynamic factor moderating the over-exposure of medium- and low-popular items. We hence define d_i as:

$$d_i = \frac{r_i}{max(1, h)} + 1\,, \tag{21}$$

where r_i is the rank of the item i (ranging from 0 for the most popular item, to $|I| - 1$ for the least popular one), and h represents the highest rank of the set of items for which we want to preserve a certain number of pairwise comparisons. Its value is the rank of the last high-popular item, i.e. $|I_H| - 1$. The term $\frac{r_i}{h}$ indicates how far the item i is from the top popular ones: the farther, the more d_i will penalize n'_i.

This dynamic sampling strategy consists in feeding the recommender with n_i pairwise comparisons for each positive occurrence of $i \in \mathbf{X}$, where:

$$n_i = \begin{cases} \lceil n'_i \rceil & \text{if } 0 < \epsilon \le n'_i - \lfloor n'_i \rfloor \\ \lfloor n'_i \rfloor & \text{otherwise} \end{cases}\,, \tag{22}$$

with $\epsilon \sim \mathcal{U}(0, 1)$ sampled from a uniform distribution. The random process mitigates the overexposure of popular items that are not maximally popular, which a ceiling process would produce.

By construction, the exposure of the top-popular item i_{top} coincides with the one induced by the baseline, as shown in the top-right corner of Figure 4, due to $d_{i_{top}} = 1$ and $\rho_{i_{top}} = max(\boldsymbol{\rho})$ (see Equations 19-21). From here, since the value of d_i progressively increases, the exposure of the items is adapted according to the popularity classes. Compared to the baseline, the exposure of high-popular items exhibits negligible changes, while medium- and especially low-popular items gain much more relevance, while the overall popularity relationships are kept coherent and smooth.

REFERENCES

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling Popularity Bias in Learning-to-Rank Recommendation *(RecSys '17)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3109859.3109912

[2] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2019. Managing Popularity Bias in Recommender Systems with Personalized Re-Ranking. In *International Florida Artificial Intelligence Research Society Conference (FLAIRS '19)*. 413–418.

[3] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2019. The Unfairness of Popularity Bias in Recommendation. In *Workshop on Recommendation in Multi-stakeholder Environments (CEUR Workshop Proceedings '19)*, Vol. 2440.

[4] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2020. The Connection Between Popularity Bias, Calibration, and Fairness in Recommendation. In *ACM Conference on Recommender Systems (RecSys '20)*. 726–731.

[5] Gediminas Adomavicius and YoungOk Kwon. 2012. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Transactions on Knowledge and Data Engineering* 24, 5 (2012), 896–911. https://doi.org/10.1109/TKDE.2011.15

[6] Charu C. Aggarwal. 2016. *Recommender Systems*. Springer.

[7] Ludovico Boratto, Gianni Fenu, and Mirko Marras. 2021. Connecting user and item perspectives in popularity debiasing for collaborative recommendation. *Information Processing & Management* 58, 1 (2021), 102387.

[8] Rodrigo Borges and Kostas Stefanidis. 2020. On Measuring Popularity Bias in Collaborative Filtering Data. In *EDBT Workshop on BigVis 2020: Big Data Visual Exploration and Analytics (EDBT/ICDT Workshops)*.

[9] Sushma Channamsetty and Michael D. Ekstrand. 2017. Recommender Response to Diversity and Popularity Bias in User Profiles. In *International Florida Artificial Intelligence Research Society Conference (FLAIRS '17)*. 657–660.

[10] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2023. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems* 41, 3 (2023), 1–39.

[11] Zhihong Chen, Jiawei Wu, Chenliang Li, Jingxu Chen, Rong Xiao, and Binqiang Zhao. 2022. Co-Training Disentangled Domain Adaptation Network for Leveraging Popularity Bias in Recommenders. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3477495.3531952

[12] Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. 2009. Power-Law Distributions in Empirical Data. *SIAM Rev.* 51, 4 (nov 2009), 661–703. https://doi.org/10.1137/070710111

[13] Sihao Ding, Fuli Feng, Xiangnan He, Jinqiu Jin, Wenjie Wang, Yong Liao, and Yongdong Zhang. 2022. Interpolative distillation for unifying biased and debiased recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 40–49.

[14] Travis Ebesu, Bin Shen, and Yi Fang. 2018. *Collaborative Memory Network for Recommendation Systems*. Association for Computing Machinery, 515–524.

[15] Michael D Ekstrand, Mucun Tian, Ion Madrazo Azpiazu, Jennifer D Ekstrand, Oghenemaro Anuyah, David McNeill, and Maria Soledad Pera. 2018. All The Cool Kids, How Do They Fit In?: Popularity and Demographic Biases in Recommender Evaluation and Effectiveness. In *Conference on Fairness, Accountability, and Transparency (PMLR '18)*. 172–186.

[16] Batya Friedman and Helen Nissenbaum. 1996. Bias in Computer Systems. *ACM Transactions on Information Systems* 14, 3 (1996), 330–347.

[17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. 173–182.

[18] Gary W. Heiman. 2001. *Understanding research methods and statistics: An integrated introduction for psychology*. Houghton, Mifflin and Company.

[19] Balázs Hidasi and Ádám Tibor Czapp. 2023. Widespread Flaws in Offline Evaluation of Recommender Systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 848–855.

[20] 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.

[21] Toshihiro Kamishima, Shotaro Akaho, and Hideki Asoh. 2014. Correcting Popularity Bias by Enhancing Recommendation Neutrality. In *ACM Conference on Recommender Systems (RecSys'14)*.

[22] Dominik Kowald, Markus Schedl, and Elisabeth Lex. 2020. The Unfairness of Popularity Bias in Music Recommendation: A Reproducibility Study. In *Advances in Information Retrieval*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.).

[23] Xiaopeng Li and James She. 2017. Collaborative Variational Autoencoder for Recommender Systems. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*. 305–314.

[24] Dawen Liang, Laurent Charlin, James McInerney, and David M. Blei. 2016. Modeling User Exposure in Recommendation. In *ACM Conference on World Wide Web (WWW '16)*. 951–961.

[25] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. 689–698.

[26] Yudan Liu, Kaikai Ge, Xu Zhang, and Leyu Lin. 2019. Real-time Attention Based Look-alike Model for Recommender System. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. 2765–2773.

[27] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. 2020. Feedback loop and bias amplification in recommender systems. In *ACM International Conference on Information and Knowledge Management (CIKM '20)*. 2145–2148.

[28] MEJ Newman. 2005. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics* 46, 5 (sep 2005), 323–351. https://doi.org/10.1080/00107510500052444

[29] M. E. J. Newman. 2003. The Structure and Function of Complex Networks. *SIAM Rev.* 45, 2 (jan 2003), 167–256. https://doi.org/10.1137/s003614450342480

[30] Steffen Rendle. 2012. Factorization Machines with LibFM. *ACM Trans. Intell. Syst. Technol.* 3, 3 (2012).

[31] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Conference on Uncertainty in Artificial Intelligence (UAI '09)*. 452–461.

[32] Noveen Sachdeva, Giuseppe Manco, Ettore Ritacco, and Vikram Pudi. 2019. Sequential Variational Autoencoders for Collaborative Filtering. In *ACM International Conference on Web Search and Data Mining (WSDM '19)*. 600–608.

[33] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. In *international conference on machine learning*. PMLR, 1670–1679.

[34] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. AutoRec: Autoencoders Meet Collaborative Filtering. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. 111–112.

[35] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I. Nikolenko. 2020. RecVAE: A New Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *International Conference on Web Search and Data Mining (WSDM '20)*. 528–536.

[36] Harald Steck. 2018. Calibrated Recommendations. In *ACM Conference on Recommender Systems (RecSys'18)*. 154–162.

[37] Virginia Tsintzou, Evaggelia Pitoura, and Panayiotis Tsaparas. 2019. Bias Disparity in Recommendation Systems. In *Workshop on Recommendation in Multi-stakeholder Environments (CEUR Workshop Proceedings '19)*, Vol. 2440.

[38] Hamed Valizadegan, Rong Jin, Ruofei Zhang, and Jianchang Mao. 2009. Learning to rank by optimizing ndcg measure. *Advances in neural information processing systems* 22 (2009).

[39] Wenjie Wang, Fuli Feng, Xiangnan He, Xiang Wang, and Tat-Seng Chua. 2021. Deconfounded recommendation for alleviating bias amplification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1717–1725.

[40] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19)*. 165–174.

[41] 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 *ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD '21)*. 1791–1800.

[42] Guipeng Xv, Chen Lin, Hui Li, Jinsong Su, Weiyao Ye, and Yewang Chen. 2022. Neutralizing popularity bias in recommendation models. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2623–2628.

[43] 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).

[44] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are Graph Augmentations Necessary? Simple Graph Contrastive Learning for Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 1294–1303. https://doi.org/10.1145/3477495.3531937

[45] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling User Interest and Conformity for Recommendation with Causal Embedding. In *Proceedings of the Web Conference 2021 (WWW '21)*. 2980–2991.

[46] Ziwei Zhu, Xia Hu, and James Caverlee. 2018. Fairness-Aware Tensor-Based Recommendation. In *ACM International Conference on Information and Knowledge Management (CIKM '18)*. 1153–1162.

[47] Ziwei Zhu, Jianling Wang, and James Caverlee. 2020. Measuring and Mitigating Item Under-Recommendation Bias in Personalized Ranking Systems. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 449–458.