On Measuring Popularity Bias in Collaborative Filtering Data

Rodrigo Borges
rodrigo.borges@tuni.fi
Tampere University
Tampere, Finland

Kostas Stefanidis
konstantinos.stefanidis@tuni.fi
Tampere University
Tampere, Finland

ABSTRACT

The effect of having few data items responsible for the majority of ratings in a Collaborative Filtering recommendation, and the complement of having majority of items responsible for few ratings given by the users, are usually referred as *popularity bias*. The effect is known as reflecting the preference of users for popular items, but also as a consequence of methods and metrics normally applied by these systems. Variational Autoencoders (VAE) are considered today the state-of-the-art for collaborative filtering recommenders, and can handle big and sparse data entries with robustness and high accuracy. A methodology is proposed here for characterizing the popularity bias in Movielens and Netflix datasets, and when applying VAE for generating recommendations based on them. As a first step, the long tail model is applied for segmenting items and users in three different classes (Short Head, Medium Tail and Long Tail), depending on the proportion of interactions they are associated with. In addition, a real recommendation scenario is presented for measuring the proportion of unpopular items appearing among the suggestions provided by VAE. We consider characterizing the popularity in details as a very first step for providing recommenders with the desired serendipity effect, and expanding the knowledge of these systems about new and unpopular items with few ratings.

KEYWORDS

Recommendations; Bias; Collaborative filtering; Variational Autoencoders

1 INTRODUCTION

A huge amount of data is produced, converted to digital format and published online each day, from scientific articles to potential romantic partners information. Nevertheless, the amount of time users have available to spend browsing in platforms is severely limited if compared to the size of most of these catalogs. This motivated the development of recommender systems (RS), proposed for presenting to users a subset of items he/she will most likely react positively.

The most popular approach for implementing RS is called collaborative filtering (CF), and relies in a premise that users who interacted with items similarly in the past (e.g., bought many common books) are similar to each other. Once they shared previous decisions, they are assumed as maintaining their behavior and sharing also future ones. CF solutions explore this assumption suggesting to each user the items his/her neighbors, i.e., users with similar behavior, consumed, but that he/she has not had contacted yet.

Let's assume a scenario in which a recommender operates through an algorithm trained according to an error-based metric (as most of them really do) [3]. By error-based metric, we mean that its success is measured by the number of right guesses it makes in an separated part of the data (test set) after adjusting its weights. Let's assume that after a big number of rounds of recommendations, 10% of the available items were responsible for more than 30% of users interactions registered by the platform. In its next train procedure, the algorithm will try to adjust its parameters to maximize its overall accuracy, which will certainly account mostly for those 10% items than for unpopular ones responsible, for example, for 0.5% of the play counts. We imagine this happening successively, and in each round the model is more adjusted according to popular items, and unaware of a great slice of items that could potentially found their niches of consumption.

We assume the *popularity bias* effect as a mixture of unbalanced preferences authentically expressed by the users, and a side effect of algorithms and metrics applied in the current systems. Apart from that, suggesting unpopular items has the desired effect of serendipity (providing users with novelty), and also expand the knowledge of the system about unpopular items with very sparse rating information [21].

We propose a methodology for identifying and retrieving the bias contained in collaborative filtering popular datasets. Interactions between user and items composing Movielens and Netflix datasets were represented as static data for identifying popularity. They were ordered by the number of occurrences and segmented in Short Head, Medium Tail and Long Tail. First gathered by items and then by users.

Variational Autoencoder (VAE) is a dimensionality reduction technique considered today as the state-of-the-art for the task of CF [15, 18]. It represents user interaction as normal distributions in a latent space, with great power of predicting unseen item ratings. We are here interested in tracking how prone to bias this solution is. We conduct standard procedures for training and testing it, and measure the proportion of each popularity categories presented in the results.

The rest of the paper is structured as follows. In Section 2, we present a detailed analysis of both datasets focusing on popularity bias. In Section 3, we provide a metric for retrieving how users are prone to each class of items, named *Mainstreaminess*. In Section 4, we simulate a real scenario of recommendations for checking how biased the state-of-the-art collaborative filtering approach is. We conclude and point our future works in Section 6.

2 MEASURING POPULARITY BIAS

We assume here the *popularity bias* effect as a proxy of having few data items responsible for the majority of ratings in a dataset, and the complementary effect of having majority of items responsible for very few ratings given by the users.

In order to demonstrate the effect of popularity bias in a real world scenario, we selected two datasets widely used in recommender systems research field, both describing movies consumption: the first one is provided by Movielens[1], and the second one by Netflix [4]. A summary of the characteristics of the datasets

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

(events[2], users, items and sparsity) is presented in the first 5 columns of Table 1.

We follow demonstrating the bias effect in the datasets by applying the well known *long tail model* [17], in which items are ordered according to the number of events associated to them. The main point here is to visualize how events are concentrated in few popular items, entitled *short head* (SH), and how the remaining events are spread over the rest of them, known as *long tail* (LT). The long tail items can be even separated in two parts considering an intermediate proportion of them between popular, called *medium tail* (MT), and unpopular items. The events distribution is then segmented in three regions according to [1], who suggests 20% and 80% as the thresholds between SH and MT, and between MT and LT, respectively.

It is important to notice that the popularity bias can be addressed from two different perspectives, when considering items or users as responsible for the majority of samples. We conduct both analysis as a matter of comparison.

2.1 Item Bias

We start by applying the long tail model to each dataset by ordering items according to number of events associated to them, as presented in Figure 1. It is possible to see the decaying effect when moving from popular items to unpopular ones (from left to right), and the three regions, defined by vertical dashed lines, corresponding to SH, MT and LT . The x axis of the plots are maintained linear while the y one is converted to logarithmic for the sake of visibility.

The unbalanced effect of consumption becomes clear when analyzing the curves. Netflix data items distribution presents a wider MT region and a smoother decay than in the case of Movielens. Extremely unpopular items (the ones surrounding the 1 value in y axis) represent approximately 15% of the first dataset, and are not observed in the second one.

Fitting these distributions to power law is useful applying a general model [21]. The same data is also presented in the log-log format in Figure 2. According to [9], a quantity x obeys a power law if it is drawn from a probability distribution $p(x) \propto x^{-\alpha}$. Instead of a simple power law distribution, we have detected that a best overall fitting results occur when considering its shifted version, $p(x) \propto (a + bx)^{-\alpha}$. The Kolmogorov-Smirnov goodness of fit for Movielens items is 0.67 (α = 1.2), and for Netflix is 0.38 (α = 1.7).

The general comparison of three datasets is presented in Tables 1 and 2. The first relevant information to be mentioned is the number of interactions an item received in order to be considered in each class of the long tail model. In Columns 6 and 7 of Table 1, it is possible to notice that in order to be considered a popular movie in the Movielens dataset, a movie should sum more than 23,301 play counts, and in order to belong to LT it should have had been watched less than 2,140 times. When it comes to Netflix items the situation changes, a popular movie now accounts for more than 101,061, and an unpopular one for less than 9,635 interactions.

The next interesting information to be highlighted is the actual sizes of SH, MT and LT regions in data distributions. Table 2 shows the general information about this segmentation, indicating the proportion of items belonging to each popularity class. When considering extremely unpopular items, Movielens dataset

[2]We are here considering an event as one single line in the dataset, containing information of user id, movie id and timestamp.

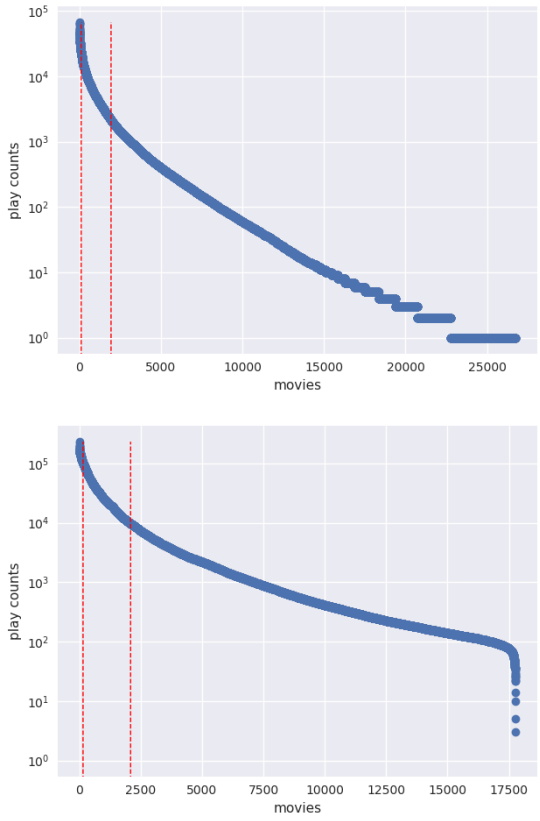

Figure 1: Lin-log plots for MovieLens (top) and Netflix (bottom) datasets interactions distribution grouped by items. The thresholds of 20% and 80% of the total summation are indicated in vertical dashed lines.

is more prone to bias, having 92.3% of its movies corresponding to less than 80% of its online activities. When considering the popular ones, it presents also the smallest SH, with just 0,4% of the items responsible for 20% of all user activity. Netflix data presents considerably larger MT, i.e., 10,9% of all movies watched in the platform.

2.2 User Bias

We analyze the bias of our datasets, considering the complementary effect of having few items concentrating the majority of interactions in an on-line service, which is the effect of having few and very active users along with very sparse ones who rarely provides feedback to the system. The same methodology is replicated here, but now considering the effect happening because of different reasons.

A similar decaying effect is observed provoked by the different behavior of users, as one can see in Figure 3. This time Movielens curve have smother decaying then in the case of items, and the proportions associated to the three consumption categories seems more homogeneous than before.

None of Movielens users watched nearly zero movies, and it is possible to observe sharp slopes in SH and LT regions in the Netflix distribution curve. But the reasons that may have provoked these discontinuities goes beyond the scope of this article.

Table 1: Datasets description.

Dataset	#Events	#Users	#Items	Sparsity	20% Items	80% Items	20% Users	80% Users
MovieLens	20,000,263	138,493	26,744	0.54%	23,301	2,140	775	100
Netflix	100,325,382	477,412	17,768	1.18%	101,061	9,635	966	178

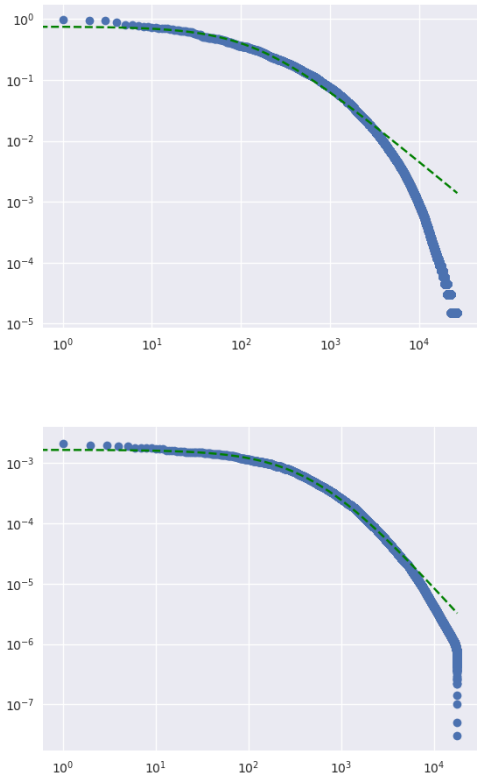

Figure 2: Log-log plots for MovieLens (top) and Netflix (bottom) datasets interactions grouped by items.

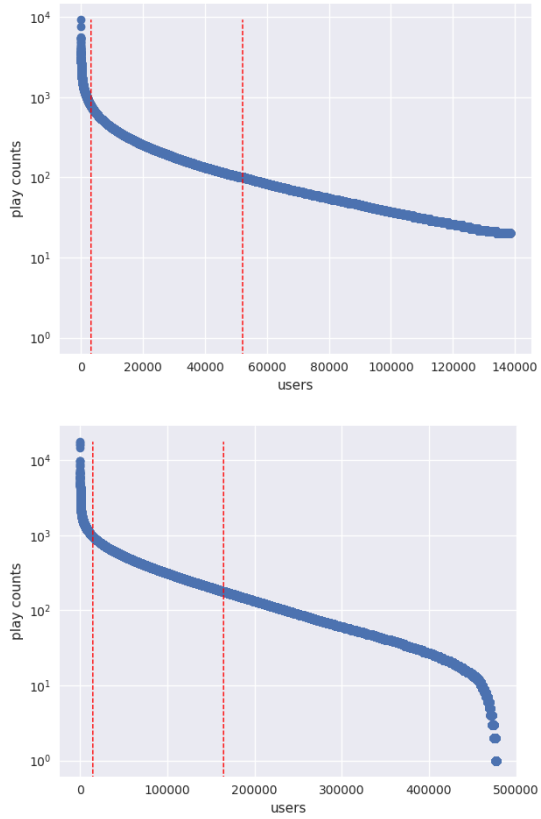

Figure 3: Lin-log plots for MovieLens (top) and Netflix (bottom) datasets interactions distribution grouped by users. The thresholds of 20% and 80% of the total summation are indicated in vertical dashed lines.

Table 2: Item Bias Summary

Dataset	Short Head	Medium Tail	Long Tail
MovieLens	118 (0,4%)	1,831 (6,8%)	24,677 (92,3%)
Netflix	154 (0.8%)	1,938 (10,9%)	15,522 (87,4%)

The log-log graphs for the user-oriented analysis is presented in Figure 4. The fitting indexes to the theoretical distribution are 0.42 (α = 0.7), 0.37 (α = 1.0) for Movielens and Netflix respectively, indicating stronger evidences that these data can be modeled by the shifted power law distribution.

The description of the user-based analysis is presented in Tables 1 and 3. In order to be considered a heavy user of Movielens, and consequently belonging to its SH proportion of the user distribution, someone should have watched more than 775 movies, and for the case of sparse users, less than 100 movies. The person who watched more than 966 movies in Netflix is considered a frequent user, and the one who made less than 178 contributions to the data is considered in the long tail proportion of the distribution, located in the right area in Figure 3.

A general overview of the analysis of how unbalanced users interactions occur is presented in Table 3. SH and MT proportions are generally bigger when compared to the previous case. And as a direct consequence, less users are considered sparse in both cases.

Table 3: User Bias Summary

Dataset	Short Head	Medium Tail	Long Tail
MovieLens	3,261 (2,4%)	48,909 (35,3%)	83,062 (60%)
Netflix	14,444 (3%)	149,905 (31,4%)	298,619 (62,5%)

3 MEASURING MAINSTREAMINESS

To conclude our analysis of users for the datasets, we introduce a metric named *mainstreaminess*, for retrieving the information of how users are prone to each class of items. A similar approach is conducted by [2], when three different types of users are defined according to their interest in popular items. Here, we are

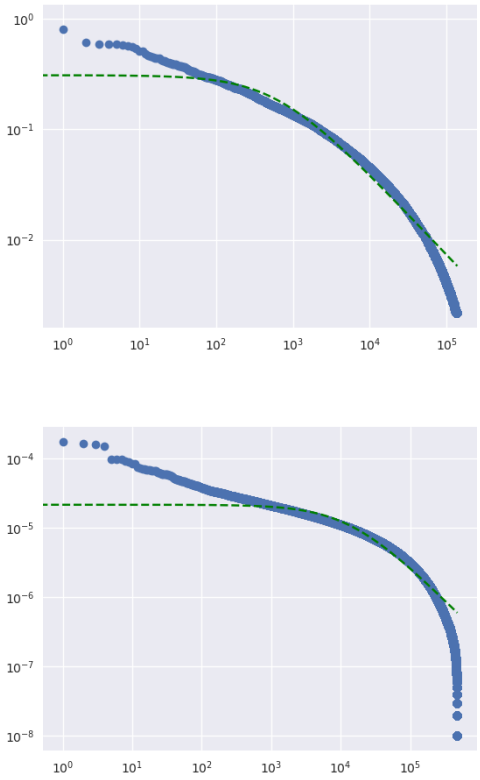

Figure 4: Log-log plots for MovieLens (top) and Netflix (bottom) datasets interactions grouped by users.

interested in giving a general overview of the dataset by taking an average of normalized profiles considering the categories of items.

The main idea is to iterate through each user: (i) building a profile with how many items belong to each region of the item distribution (SH, MT and LT), (ii) normalizing the profile by the number of items consumed, and (iii) taking an average of these values for characterizing the dataset as a whole. For doing so, we adopt the Average Percentage Tail [1]:

$$APT = \frac{1}{|U_t|} \sum_{u \in U_t} \frac{|\{i, i \in (L(u) \cap \Phi)\}|}{|L(u)|} \tag{1}$$

where Φ corresponds to one of the three categories of items (short-head, medium-tail or long-tail), $L(u)$ to the profile of users u, and U_t to the set of users. For ease of representation we define APT-SH, APT-MT and APT-LT as the proportion of each category respectively.

The mainstreaminess measurements for the Movielens and Netflix datasets appear in Figure 5. These results indicate Netflix users more prone to popular items than Movielens ones. The highest proportion of MT consumption is observed in the case of Movielens, together with the smallest proportion of LT.

4 POPULARITY BIAS IN VARIATIONAL AUTOENCODERS

In order to verify the proposition in a recommendation situation, we count the proportion of long tail, medium tail and short head

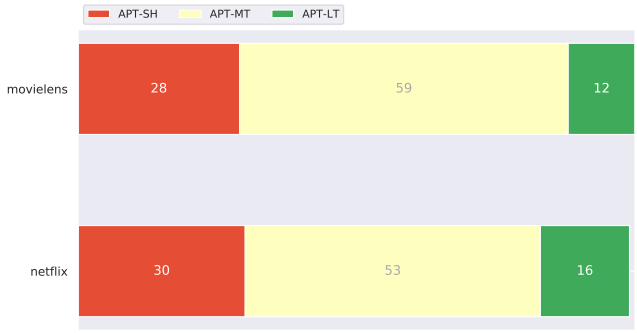

Figure 5: Mainstreaminess, or how users are prone to each item class, for Movielens and Netflix datasets.

items presented in each round of suggestions, in a regular operation of a state-of-the-art algorithm for Collaborative Filtering named Variational Autoencoders [15].

Variational Autoencoders (VAE) can be interpreted as a model whose aim is to find the probability distribution responsible for generating their input data. Lets suppose a set of input data $x \in \mathbb{R}^{d_x}$ following an unknown probability distribution $p(x)$. And a set of latent variables defined in a low dimensional space $z \in \mathbb{R}^{d_z}$ ($d_z \ll d_x$). The final model can be summarized as $p(x, z) = p(x|z)p(z)$, from where one could marginalize z and find $p(x)$. But the situation is that for most cases this integral can not be found in closed form [14].

Variational Inference (VI) [13] was recently proposed to address this problem thorught optimization, assuming that the distribution can be approximated by a simpler one that still models the data. VI specifies Q as a family of densities where members $q(z|x) \in Q$ is a candidate to the conditional $p(z|x)$. The inference occurs by minimizing the Kullback-Leibler (KL) between approximated and original density. After re-arranging terms, we have

$$\begin{aligned} \log p(x) - D_{KL}[q(z|x)||p(z|x)] = \\ \mathbb{E}_z[\log p(x|z)] - D_{KL}[q(z|x)||p(z)] \end{aligned} \tag{2}$$

We now want to maximize $\log p(x)$ minus the approximation error, and as an alternative we can optimize the second term. In order to do this, we rely on parametric distributions q_ϕ and p_θ. The optimization process will correspond to optimize parameters ϕ and θ of these distributions with:

$$\mathcal{L}_{\theta,\phi} = \mathbb{E}_z[\log p_\theta(x|z)] - \beta \cdot D_{KL}[q_\phi(z|x)||p_\theta(z)] \tag{3}$$

Where $\mathcal{L}$ is the Evidence Lower Bound (ELBO), $p_\theta(x|z)$ corresponds to the estimation of z space departing from input data, named Encoder, and $q_\phi(z|x)$ corresponds to estimating the original data departing from the latent space, named Decoder (Figure 6). And this defines Variational Autoencoder. The first term addresses the reconstruction error, and the second term the error of distribution approximation. The parameter β controls the strength of regularization [15].

We start from the implementation published by the authors[3] and include an item mapper to it so the model can refer each item to its category in the Long Tail model. The proportion of the items belonging to each category in the top-k recommendation list is measured with equation 1.

[3] https://github.com/dawenl/vae_cf

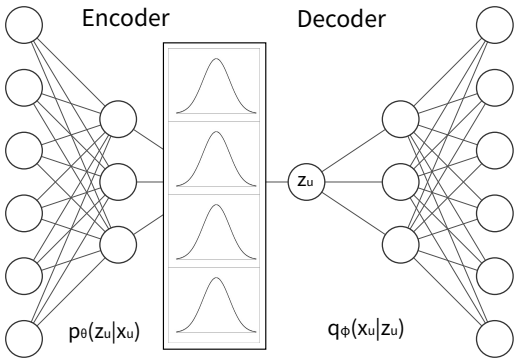

Figure 6: Variational Autoencoders

In the case of Movielens data, users who rated less than 5 items as well as ratings (fom 0 to 5) lower than 3.5 were removed, ending up with 9,990,682 watching events from 136,677 users and 20,720 movies. 10,000 users were separated and split for validation and test (5,000/ 5,000). VAE was trained with two hidden layers [20,720 -> 600 -> 200 -> 600 -> 20,720] for 200 epochs. The training batch size was set to 500 and the validation batch to 2,000. Weight initialization, activation functions, learning rate, and β regulation were inherited from [15].

The general results for the validation set achieved 0.33 for NDCG@10 and 0.34 for Recall@10 as the best results, and with reasonably stable values after hundred of epochs.

The proportion of LT items increases in the first epochs of the training, but stabilizes in an irrelevant proportions of recommendation results during the process (Figure 7). The proportion of SH items starts with extremely high proportion, for reaching an almost stationary state around 60% of items results in the first 10 higher scores provided by the recommender. A complementary effect is observed for MT items, representing approximately 40% of items after few epochs.

In the case of Netflix, ratings lower than 3.5 and users who rated less than 5 items were also removed, ending up with 56,785,778 watching events from 461,285 users and 17,767 movies (sparsity: 0.693%). The same parameter values were replicated, except for the size of input layer, which is smaller now [17,767 -> 600 -> 200 -> 600 -> 17,767]. 40,000 users were separated and divided equaly for validation and test. The model was trained for 200 epochs.

The results achieved for the validation data were 0.32 for both NDCG@10 and Recall@10 metrics, comparable to best scenarios provided in the original paper, 0.39 and 0.35, but for NDCG@100 and Recall@20 respectively.

A similar behavior is observed in the case of Netflix data, but in a slightly lower level than in the previous dataset. The proportion of extremely popular items corresponds now to approximately 55% of suggestions. The proportion of LT items is also irrelevant, as one can notice in Figure 7.

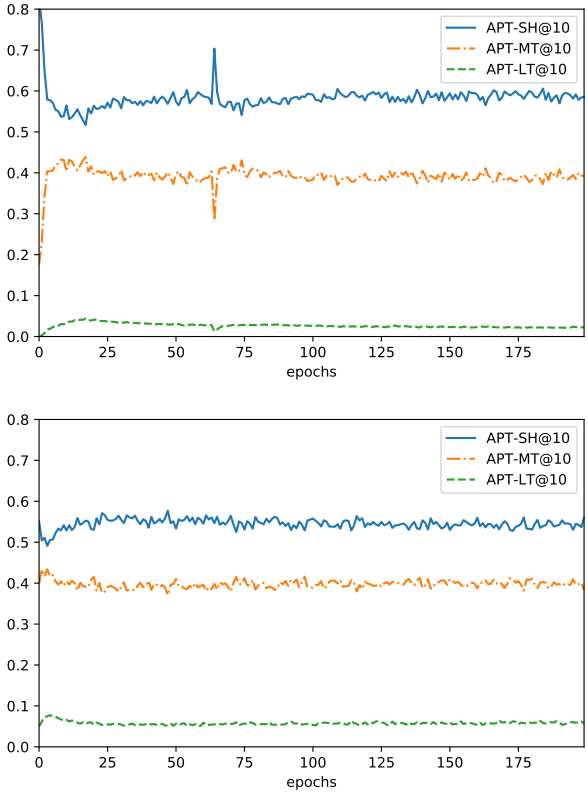

Figure 7: APT-SH, APT-MT and APT-LT proportions in the validation set during Variational Autoencoders for Collaborative Filtering training procedure for: (Top) Movielens and (Bottom) Netflix datasets.

5 RELATED WORK

Fairness: Typically, approaches for amplifying biases focus on how to strengthen fairness. In recommendations, such approaches can be distinguished as pre-processing, in-processing and post-processing. Pre-processing approaches target at transforming the data so that any underlying bias or discrimination is removed. Such approaches work on modifying the input to the recommender, for example, by appropriate sampling (e.g., [7]), by adding more data to the input (e.g., [22]), or by performing database repair [19]. In-processing approaches target at modifying existing or introducing new algorithms that result in fair recommendations, e.g., by removing bias. Existing approaches focus on fairness-aware matrix factorization [24], multi-armed bandits [11] and tensor factorization (e.g., [25]). When fairness with respect to both consumers and to item providers is important, variants of the well-known sparse linear method (SLIM) can be used to negotiate the trade-off between fairness and accuracy and improve the balance of user and item neighborhoods [6]. Alternatively, we can augment the learning objective in matrix factorization by adding a smoothed variation of a fairness metric [24]. As another example, [5] presents a method that mitigate bias to increase fairness by incorporating randomness in variational autoencoders recommenders. Post-processing approaches treat the algorithms for producing recommendations as black boxes, without changing their inner workings. To ensure fairness, they modify the output of the algorithm (e.g., [12]). Moving from individuals to groups, significant research efforts have been done recently (e.g., [16, 20, 23]), targeting at maximizing the satisfaction of each group member, while minimizing the unfairness between them.

Popularity Bias: The popularity bias in recommendation results may be inherited from the data used to train the models, or even from methods and metrics applied by them. [3] points for the limitations of error-based evaluation metrics widely used in the field of recommender systems. They argue that methods

trained to maximize the satisfaction of the majority of users will perform well on these metrics, but the problem relies on the fact that items with many training ratings will tend to have more positives test ratings, and will be liked by more users according to the test data.

It is important to differentiates the preferences expressed by user in historical data, from the *true* preference of a hypothetical scenario where all users would have rated all items, as pointed in [21]. The author proposes a nearly unbiased accuracy measurement for recommendation experiments, named Popularity-Stratified Recall, which favors items from the long tail with the aim of approximating observed and true preferences. The power-law modeling is proposed as a surrogate of the unobserved rating information, in the context where recommendations from the long tail present small bias, but also increase variance and reduce the accuracy. [1] addresses the popularity bias in matrix factorization solutions for recommendation by exploring the trade off between long tail coverage and ranking performance. The regularization factor is associated to the bias in results, to be adjusted in the experiments.

[10] tackles the specific situation of music recommendation platforms considering the bias presented in datasets available for training ML models as a possible reason for a situation where a group of artists are not suggested to users and therefore receive less compensation by streaming content providers. [8] has compared Collaborative Filtering, Content-Based and Expert-Based music recommendation engines for detecting popularity effect and the influence of the most popular artists in the network. They figured out that the collaborative algorithm is prone to popularity bias, and that the two other approaches are more efficient when exploring the long tail of the play count distributions.

6 CONCLUSION

When discussing about popularity bias, the first question that may come to someone is: if there are items more attractive than others, why promoting unpopular ones to a wider public? The first answer to this is commonly referred as the cold-start situation: when new items are introduced in the platforms, and need to be incorporated in the algorithm. We are here talking about potentially popular items with no historical data, that will need to enter the long tail before reaching the short head of the distribution. The challenge, in this case, is promoting relevant items among unpopular ones.

Mainstreaminess measurement have revealed users' preferences concentrated in MT items. In the case of Movielens it sums almost double the size of SH, and 6 times of the LT proportion. Even then, the recommenders suggests majority of SH items during the training phase. As discussed here before, the recommnender have probably learned with more information about popular items and this results in biased results.

We consider the popularity bias effect as being associated to the inner operation of the platforms, as much as to the social effect in which people interact with popular items. Experiments like the ones presented here are intended for measuring the overall bias, without distinguishing both sources. We argue that characterizing popularity bias in recommenders data and algorithms is a first step for addressing it, and addressing also, as a consequence, the problem of cold-start.

Another objective of this study was to claim attention for the researchers working on recommender systems field, specially the ones working with Movielens and Netflix datasets, that popularity bias is present in most of the data available for experiments. This should help future studies when deciding thresholds for filtering items, when separating users in a test set, and also for considering the possibility of potential popular items with few ratings.

As future work, we aim to explore in detail how to address popularity bias in the specific case of Variational Autoencoders applied in collaborative filtering. This is a powerful and scalable solutions, for high performance recommendations, but also with space for improvements.

REFERENCES

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

[2] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2019. The Unfairness of Popularity Bias in Recommendation. In *RMSE*.

[3] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2017. Statistical biases in Information Retrieval metrics for recommender systems. *Information Retrieval Journal* 20, 6 (2017), 606–634.

[4] James Bennett, Stan Lanning, and Netflix Netflix. 2007. The Netflix Prize. In *In KDD Cup and Workshop in conjunction with KDD*.

[5] Rodrigo Borges and Kostas Stefanidis. 2019. Enhancing Long Term Fairness in Recommendations with Variational Autoencoders. In *MEDES*.

[6] Robin Burke. 2017. Multisided Fairness for Recommendation. *CoRR* abs/1707.00093 (2017).

[7] L. Elisa Celis, Amit Deshpande, Tarun Kathuria, and Nisheeth K. Vishnoi. 2016. How to be Fair and Diverse? *CoRR* abs/1610.07183 (2016).

[8] Ò. Celma and P. Cano. 2008. From hits to niches? or how popular artists can bias music recommendation and discovery. In *2nd Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition (ACM KDD)*.

[9] Aaron. Clauset, Cosma Rohilla. Shalizi, and M. E. J. Newman. 2009. Power-Law Distributions in Empirical Data. *SIAM Rev.* 51, 4 (2009), 661–703.

[10] Andre Holzapfel, Bob L. Sturm, and Mark Coeckelbergh. 2018. Ethical Dimensions of Music Information Retrieval Technology. *Transactions of the International Society for Music Information Retrieval* 1, 1 (2018), 44 – 55.

[11] Matthew Joseph, Michael J. Kearns, Jamie H. Morgenstern, and Aaron Roth. 2016. Fairness in Learning: Classic and Contextual Bandits. In *NIPS*.

[12] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2018. Recommendation Independence. In *FAT*.

[13] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

[14] Diederik P. Kingma and Max Welling. 2019. An Introduction to Variational Autoencoders. *CoRR* abs/1906.02691 (2019).

[15] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *WWW*.

[16] Xiao Lin, Min Zhang, Yongfeng Zhang, Zhaoquan Gu, Yiqun Liu, and Shaoping Ma. 2017. Fairness-Aware Group Recommendation with Pareto-Efficiency. In *RecSys*.

[17] Yoon-Joo Park and Alexander Tuzhilin. 2008. The Long Tail of Recommender Systems and How to Leverage It. In *RecSys*.

[18] Noveen Sachdeva, Giuseppe Manco, Ettore Ritacco, and Vikram Pudi. 2019. Sequential Variational Autoencoders for Collaborative Filtering. In *WSDM*.

[19] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. 2019. Interventional Fairness: Causal Database Repair for Algorithmic Fairness. In *SIGMOD*.

[20] Dimitris Serbos, Shuyao Qi, Nikos Mamoulis, Evaggelia Pitoura, and Panayiotis Tsaparas. 2017. Fairness in Package-to-Group Recommendations. In *WWW*.

[21] Harald Steck. 2011. Item Popularity and Recommendation Accuracy. In *RecSys*.

[22] Harald Steck. 2018. Calibrated recommendations. In *RecSys*.

[23] Maria Stratigi, Jyrki Nummenmaa, Evaggelia Pitoura, and Kostas Stefanidis. 2020. Fair Sequential Group Recommendations. In *SAC*.

[24] Sirui Yao and Bert Huang. 2017. Beyond Parity: Fairness Objectives for Collaborative Filtering. In *NIPS*.

[25] Ziwei Zhu, Xia Hu, and James Caverlee. 2018. Fairness-Aware Tensor-Based Recommendation. In *CIKM*.