5-2019

Debiasing the Human-Recommender System Feedback Loop in Collaborative Filtering

Wenlong Sun
University of Louisville

Sami Khenissi
University of Louisville

Olfa Nasraoui
University of Louisville, olfa.nasraoui@louisville.edu

Patrick Shafto
Rutgers University - Newark

Original Publication Information

Wenlong Sun, Sami Khenissi, Olfa Nasraoui, and Patrick Shafto. "Debiasing the Human-Recommender System Feedback Loop in Collaborative Filtering." 2019. *Companion Proceedings of The 2019 World Wide Web Conference* (*WWW '19*). Association for Computing Machinery, New York, NY, USA, 645–651.

Debiasing the Human-Recommender System Feedback Loop in Collaborative Filtering

Wenlong Sun*
wenlong.sun@louisville.edu
University of Louisville
Louisville, KY

Sami Khenissi*
sami.khenissi@louisville.edu
University of Louisville
Louisville, KY

Olfa Nasraoui*
olfa.nasraoui@louisville.edu
University of Louisville
Louisville, KY

Patrick Shafto*
patrick.shafto@rutgers.edu
Rutgers University - Newark
Newark, NJ

ABSTRACT

Recommender Systems (RSs) are widely used to help online users discover products, books, news, music, movies, courses, restaurants, etc. Because a traditional recommendation strategy always shows the most relevant items (thus with highest predicted rating), traditional RS's are expected to make popular items become even more popular and non-popular items become even less popular which in turn further divides the haves (popular) from the have-nots (unpopular). Therefore, a major problem with RSs is that they may introduce biases affecting the exposure of items, thus creating a popularity divide of items during the feedback loop that occurs with users, and this may lead the RS to make increasingly biased recommendations over time. In this paper, we view the RS environment as a chain of events that are the result of interactions between users and the RS. Based on that, we propose several debiasing algorithms during this chain of events, and evaluate how these algorithms impact the predictive behavior of the RS, as well as trends in the popularity distribution of items over time. We also propose a novel blind-spot-aware matrix factorization (MF) algorithm to debias the RS. Results show that propensity matrix factorization achieved a certain level of debiasing of the RS while active learning combined with the propensity MF achieved a higher debiasing effect on recommendations.

CCS CONCEPTS

• **Information systems → Users and interactive retrieval**; **Collaborative search**.

KEYWORDS

Recommender Systems (RSs), Matrix Factorization, Active Learning (AL), Propensity

ACM Reference Format:
Wenlong Sun, Sami Khenissi, Olfa Nasraoui, and Patrick Shafto. 2019. Debiasing the Human-Recommender System Feedback Loop in Collaborative Filtering. In *Companion Proceedings of the 2019 World Wide Web Conference (WWW '19 Companion), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308560.3317303

WWW '19 Companion, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6675-5/19/05.
https://doi.org/10.1145/3308560.3317303

1 INTRODUCTION

The goal of a RS is to, given the user's previous ratings, predict which items the user might like. Modern RSs generally aim to discover a constant relationship between users and items. This may lead to a situation in which users only see a narrow subset of the entire range of available recommendations, a phenomenon known as the 'filter bubble' [9]. The relationship between users and items is, however, a time dependent variable because the RS predicts that some items may not be of interest to the user, and therefore these items may never actually be seen by the user. This presents a significant problem for RSs: we might know why a user likes an item, but we do not know why an item is not-liked by the user. Is it not-liked by a user because the user does not like it, or is it simply because the user has not seen the item in the RS results? Furthermore, if we assume the RS will continue to recommend items to users based on biased ratings, and that users will respond to these recommendations, the RS will slowly learn to recommend increasingly similar items. In other words, the RS will begin to systematically limit the users' ability to discover more items [15]. In this paper, we propose to model how iterated biases evolve from the continuous user-RS feedback loop, develop a series of different debiasing strategies, and evaluate how these algorithms impact the predictive accuracy of the RS, as well as trends in the popularity distribution of items over time. We also propose a novel blind spot aware matrix factorization algorithm to debias the RS.

1.1 Objectives and Contributions

First, we argue that a RS is a chain of events in which users actively interact with the output of a RS. Next, we introduce several debiasing algorithms for RSs, particularly those based on MF, during this chain of events. To debias the RS interactions with users, we propose to use three algorithms:

1) a unified recommendation and active learning strategy (active recommendation) during the interaction between users and the RS algorithm, with the goal of reducing recommendation uncertainty, while at the same time ensuring the integrity of the algorithm's performance;

2) an exposure-based collaborative filtering recommendation model that is also combined with an active recommendation to further debias the RS;

3) a blind spot aware MF, which takes into account the blind spot inherent in the learning phase of the RS.

2 RELATED WORK

2.1 Debiasing Strategies

Work on debiasing RSs has been done from various perspectives. Hu et al. first proposed using an implicit feedback model to measure the level of confidence that a user will see an item [5]. This is similar to the modern notion of a propensity score [10], which is defined as the probability that an item will be seen by the user. Schnabel et al. [10] argued that recommending items in a RS is analogous to exposing a patient to a randomized treatment in a medical study, and proposed the introduction of a propensity-scored recommendation learning system. Liang et al. proposed that a propensity score matrix be calculated first, followed by a weighted MF based on the propensity score matrix [6, 7]. Abdollahpouri et al. proposed a fairness-aware regularization aiming to reduce popularity bias in recommender system [1]. Chanely et al. observed that algorithms increase homogeneity among users, thereby decreasing the utility gained by users of the system, and presented a simulation to illustrate how this effect occurs [2]. Yang et al. considered implicit feedback models instead of explicit feedback models in their study, and proposed ways to estimate a propensity matrix based exclusively on popularity, which essentially describes propensity as an estimator of the true probability distribution [17].

Singh et al. proposed a method to construct fair rankings among relevant items, positing that the problem originates at the ranking step (recommendation step) and not during the learning process [12]. Their methodology ranked the items based on a calculated utility, and added a fairness constraint based on the propensity score. Sinha et al. considered the RS as a feedback loop, assumed that user ratings are true prior to the feedback loop, and proposed a method to deconvolve this feedback mechanism, assuming that each feedback response follows a certain mathematic relationship to previous recommendations [13]. Nasraoui et al. considered the continuous interaction between learning algorithms and humans as a Markov Chain of event, and proposed several approaches to debias the learning algorithms [8]. Shafto and Nasraoui presented a cognitive foundation for the interaction between learning algorithms and humans, showing how recommendation algorithms may affect the behavior of humans within the *human-recommender* feedback loop, and proposed possible cognitive models to study and debias the interaction [11].

Unlike the above work, Our proposed strategies aim to reduce the iterated bias that occurs during the interactions between users and the RS without the introduction of strong assumptions. Moreover, in contrast to the aforementioned research, we propose algorithms that consider the RS to be a chain of events, and then focus on debiasing the iterated bias introduced by these interactions by using an estimated propensity score, with and without an active learning strategy. We also employ the Gini coefficient and the blind spot score to quantify how the interaction affects the users' ability to discover new items.

2.2 Propensity and Active Learning

We start by summarizing the notation used in this paper, which follows the notation in [7, 10]:

$R_{u,i}$: An integer which indicates the ratings of user u to item i.
$O_{u,i}$: A binary value which indicates that user u provided a rating for item i to the system, $[O_{u,i} = 1] \rightarrow [R_{u,i}\ is\ observed]$.
$P_{u,i}$: Propensity: The probability of observing an entry. $P_{u,i} = P(O_{u,i} = 1)$.
N_U: The total number of users.
N_i: The number of users who rated item i.

2.2.1 Propensity. Recommendations contain two sources of information: the items which the user can see, and the user's recorded preference toward those items. *Propensity* refers to the probability of observing $R_{u,i}$ [7, 10]. In a real-world application, what the user sees is highly subject to selection biases. For example, in a movie recommendation system, users watch and rate movies that they like, and rarely rate movies that they dislike since they may not have seen these movies. Another example would be an advertisement recommendation system that always shows ads that it believes to be of interest to the user. This bias is expected to deepen as a result of the iterative interaction between users and the recommendation outputs.

Recent research on RSs began to take into account the role of item propensity, where user exposure to an item in a RS is viewed as analogous to exposing a patient to a treatment in a medical study [7, 10]. As a result, propensity indicates how probable it is that a new treatment can or will be exposed to a patient. In both cases, the studies try to infer the effect based on current results (whether it be the effect of a new treatment, or a new item in a RS).

Existing approaches for RSs generally under-weight items that are not rated in the system. It is, however, difficult to determine whether an item is not rated by a user because the user does not like the item, or because the user has not seen the item as a result of the inherent selection bias in the RS [7]. The main idea behind a propensity-based MF is to under-weight the unrated item for recommendation by bringing into the objective function of the model an Inverse Propensity Score, as follows [10]:

$$\underset{V,M}{argmin} \sum_{O_{u,i}=1} \frac{||R - V^T M||^2}{P_{u,i}} + \lambda(||V||_F^2 + ||M||_F^2). \quad (1)$$

Here $P_{u,i}$ represents the probability that a user u will see item i, and is also referred to as propensity score. V and M are the two latent factors in the MF.

Estimating propensity. A simple way of estimating propensity is to use a popularity score [7]. This assumes that $O_{u,i}$ follows a Bernoulli distribution, i.e., $O_{u,i} \sim Bernoulli(\rho_i)$. Note that the propensity score is fixed across users in this case, i.e., $P_{u,i} = \hat{\rho}_i$. Given a rating matrix, the popularity of an item is the proportion of items exposed to certain users among all the users (essentially, the proportion of users who have rated the item relative to all users).

Another way of estimating propensity is to assume that $O_{u,i}$ follows a Poisson distribution [7], or $O_{u,i} \sim Pois(\pi_u^T \gamma_i)$. π_u and γ_i are the two latent factors of the Gamma prior. Given a ratings matrix, this method assigns a value of 1 to an item that was rated

by a user in the observational matrix O, and 0 if it is not rated. By factoring this observational matrix, we get the propensity scores of all user and item pairs (see Eq. 2)

$$P_{u,i} = 1 - P(O_{u,i} = 0|\pi_u, \gamma_i) \approx 1 - exp\left\{-E[\pi_u^T \gamma_i]\right\}. \quad (2)$$

2.2.2 Active Learning . Active Learning (AL) is a special case of semi-supervised learning in which the system has the ability to interactively prompt users to label (or rate) items in order to improve the accuracy of the model [3]. One of the advantages of AL is that the targeted knowledge the system acquires helps accelerates the speed in which the system learns the model. One way to implement AL is to actively prompt users to rate items that have been underweighted by the RS in order to improve the quality of the ratings of this subset of items. In a rating system with range from R_{min} to R_{max} , the active learning can be formalized directly as follows [18]: Select the next item x_{act} that satisfies

$$x_{act} = \underset{x_i}{argmin}[\theta - \hat{y}|x_i]. \quad (3)$$

Here, $\hat{y}$ is the rating predicted by the RS given an item x_i. θ controls the degree of active learning, ranging from the midrange rating of $0.5(R_{min} + R_{max})$ to the maximum rating R_{max}. It can be seen that $\theta = R_{max}$ recovers pure recommendation (select the most relevant item, hence the item with highest predicted rating), while $\theta = 0.5(R_{min} + R_{max})$ recovers pure active learning (select the item with most uncertain relevance to the user based on the predicted rating, hence an item that is far from both being very relevant ($\hat{y} = R_{max}$) and very non-relevant ($\hat{y} = R_{min}$)). Cognitive experiments have shown that an active recommendation system can cover a wider choice of items, while maintaining the accuracy of the results [18].

3 PROPOSED METHODS TO DEBIAS RECOMMENDER SYSTEMS

We first introduce our interactive recommender system framework, which considers a RS as a continuous chain of events. First the initial RS suggests items to each user based on the initial training ratings and it is assumed that the users have 100% agreement with the recommendation. We then retrieve the true ratings from our masked ratings and add to the new training ratings. After that, a new recommendation based on new training data will be issued. This interaction will continue until a maximal number of iterations is reached. Algorithm 1 shows the details of our interactive recommender system with human in the loop.

We propose several recommendation strategies based on well-known algorithms to simulate real-life user-recommendation system interaction.

Conventional MF:
This model is trained using conventional MF (same as Eq. 1 with $P_{u,i} = 1$), and the system always selects the top predicted item for each user, and adds it to the next (new) training set. In other words, there is no active learning.

Conventional MF + Active Learning:
This model is trained using conventional MF (Eq. 1 with $P_{u,i} = 1$), but the system selects the active recommendation items with $\theta = 4.5$ for each user in Eq. (3). $\theta = 4.5$ is chosen to be between $\theta = 0.5(R_{min} + R_{max}) = 3$ (pure AL) and $\theta = R_{max} = 5$ (pure recommendation).

Popularity Propensity MF:
This model is trained with propensity MF (Eq. 1) [10]. Here the propensity $P_{u,i}$ is estimated based on popularity. The system always selects the top predicted item for each user, and adds it to the next (new) training set. In other words, there is no active learning.

Popularity Propensity MF + Active Learning:
This model is trained with propensity MF (Eq. 1) [10]. Here the propensity $P_{u,i}$ is estimated using popularity. The system selects the active recommendation items with $\theta = 4.5$ for each user in Eq. (3).

Poisson Propensity MF:
The model is trained with propensity MF (Eq. 1). Here the propensity $P_{u,i}$ is estimated based on Poisson MF (2) [7] on the exposure matrix. The system always selects the top predicted item for each user, and adds it to the next (new) training set. In other words, there is no active learning.

Poisson Propensity MF + Active Learning:
The model is trained with propensity MF (Eq. 1). Here the propensity $P_{u,i}$ is estimated using Poisson MF (Eq. 2) [7] on the exposure matrix and the system selects the active recommendation items using (Eq. 3) with $\theta = 4.5$ for each user (again chosen to be between $\theta = 0.5(R_{min} + R_{max}) = 3$ (pure AL) and $\theta = R_{max} = 5$ (pure recommendation)).

Algorithm 1: Interactive Recommendation System with the Human-Recommender System Feedback Loop Debiasing Mechanism

Data: Rating matrix R'_{ui}, λ, Learning rate η, $MAX_{interation}$, Iterations=0, Size of selection
Result: MAE, RMSE, Gini Coefficient
The system trains the initial Matrix Factorization model and computes predictions $\hat{R}$;
while *Iterations < Max Feedback Loop Iterations* **do**
  1 for all users u in the system {
    1.1. The system selects top-N items to recommend from the predicted ratings $\hat{R}$ based on a specialized recommending strategy;
    1.2. User u picks the selected top-N items and gives rating $R^{new}_{u,i}$ (from ground-truth complete data) for each item i;
  }
  2. The system records the popularity $P_i = N_i/N_U$ after the new ratings are taken in.
  3. The system records the metrics such as the RMSE and the Gini index of the popularity given the current rating matrix;
  4. The system retrains the model with the new rating matrix using *steps* gradient descent updates (Eq. 4 or Eq. 8 depending on the recommendation strategy chosen, and recomputes the predictions).
  5. Iterations++
end

In this preliminary work, we use both popularity and Poisson matrix factorization (PMF) to estimate the propensity. To minimize the objective function, we use stochastic gradient descent, which has been used successfully to solve MF with big datasets. For a given training rating r_{ij}, the updates for V_u and M_i can be shown to be:

$$\begin{aligned} V_u &\leftarrow V_u + \eta(2e_{ui}M_i - \lambda V_u) \\ M_i &\leftarrow M_i + \eta(2e_{ui}V_u - \lambda M_i). \end{aligned} \tag{4}$$

Here $e_{ui} = \frac{(\hat{r}_{ui} - V_u^T M_i)}{P_{u,i}}$, $\hat{r}_{ui}$ is the predicted rating and η is the learning rate for gradient descent. With a proper choice of step size, gradient descent converges to a local minimum. Propensity $P_{u,i}$ is computed as follows:

$$P_{u,i} = \begin{cases} 1.0 & for\ Conventional\ MF \\ N_i/N_U & for\ Popularity\ Propensity\ MF \\ 1 - P(O_{u,i} = 0 | \pi_u, \gamma_i) & for\ Poisson\ Propensity\ MF \end{cases} \tag{5}$$

This means that $P_{u,i} = 1$ for the first two (Conventional MF, with or without AL) recommendation strategies. $P_{u,i}$ is estimated using popularity for Popularity Propensity MF (with or without AL) strategies and it is estimated using Eq. 2 for Poisson Propensity MF (with or without AL) strategies.

In addition to the above proposed debiasing strategies above, a seventh algorithm called **Blind Spot Aware Matrix Factorization** is introduced. In this paper, we define the blind spot size as the number of item with a predicted ratings $\hat{R_{u,i}}$ that is smaller than a threshold δ , i.e., $\mathbf{D_\delta^u} = \{\mathbf{i} \in I \mid \hat{R}_{u,i} < \delta\}$. Note that because each user has their own blind spot, we define a threshold for each user. The threshold is used to set a percentile cut-off for each user, 95% in our experiments. Therefore, We define the blind spot for user u as

$$\mathbf{D_\epsilon^u} = \{\mathbf{i} \in I \mid \hat{R}_{u,i} < max_{u,i}(\hat{R}_{u,i}) * \epsilon\}. \tag{6}$$

Here ϵ is a cut-off which controls the threshold.

Our proposed Blind Spot Aware MF tries to limit the blind spot when trying to optimize the cost function of the conventional matrix factorization. The cost function for blind spot aware MF is as follows:

$$\begin{aligned} J = \sum_{u,i \in R} ||r_{u,i} - V_u^T M_i||^2 + \frac{\lambda}{2}(||V_u||^2 + ||M_i||^2) \\ + \underbrace{\frac{\beta}{2}||V_u - M_i||^2}_{Blind\ \ Spot\ \ Aware\ \ Term}. \end{aligned} \tag{7}$$

To minimize the objective function, we use stochastic gradient descent, which has been used successfully to solve MF for CF with big datasets. For a given training rating r_{ij}, the updates for V_u and M_i can be shown to be:

$$\begin{aligned} V_u &\leftarrow V_u + \eta(2e_{ui}M_i - \lambda V_u - \beta(V_u - M_i)) \\ M_i &\leftarrow M_i + \eta(2e_{ui}V_u - \lambda M_i + \beta(V_u - M_i)). \end{aligned} \tag{8}$$

Here $e_{ui} = \hat{r}_{ui} - V_u^T M_i$, $\hat{r}_{ui}$ is the predicted rating, and η is the learning rate for gradient descent. With a proper choice of step size, gradient descent converges to a local minimum.

Intuition Behind Blind Spot Aware MF: Conventional Matrix Factorization aims to predict ratings by approximating the existing ratings by the dot product between user and item. The RS tries to differentiate different items for each user as much as possible (see Figure 1). Given the same proportional range of similarity, blind spot aware MF aims to recommend more items to explore (with high relevance scores). Conventional MF has more limited choices to recommend to users given the predicted ratings range $[0.8\hat{R}_{max}, \hat{R}_{max}]$, where $\hat{R}_{max}$ is the maximum predicted rating for a user.

On the other hand, Blind Spot Aware MF tries to match the existing rating, while bringing items close to each other so that a user has a higher chance to explore more items, so that the overall blind spot size is decreased (see figure 1).

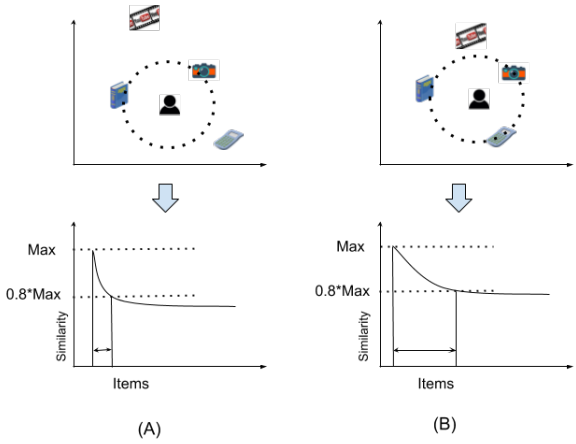

Figure 1: Conventional Matrix Factorization vs. Blind Spot Aware Matrix Factorization. Figure (A) indicates Conventional Matrix Factorization, in which the algorithm aims to find items that are close to users through differentiation. The top of Figure (A) indicates how items are distributed around a user in the latent space under Conventional MF, the bottom of (A) shows the similarity between the user and all items in latent space. Figure (B) shows how the proposed Bias-aware Matrix Factorization finds items close to users while keeping the items that are close to each other in the latent space. The top of Figure (B) indicates how items are distributed around a user in the latent space , the bottom of (B) shows the similarity between the user and all items in latent space.

4 EXPERIMENTS

4.1 Data Set

We use item response theory to generate a sparse rating matrix $\boldsymbol{R}_{u,i}$ using the model proposed in [4]. Assume a_u to be the center of user u's rating scale, and b_u to be the rating sensitivity of user u. Finally let t_i be the intrinsic score of item i. We generate a user-item rating matrix as follows,

$$\boldsymbol{R}_{u,i} = L[a_u + b_u t_i + \eta_{u,i}], \tag{9}$$

where $L[\omega]$ is the discrete level function, assigning a score in the range $R_{min} = 1$ to $R_{max} = 5 : L[\omega] = \max(\min(\text{round}(\omega),$

5), 1) and $\eta_{u,i}$ is a noise parameter. In our experiments, we draw $a_u \sim N(3.4, 1)$, $b_u \sim N(0.5, 0.5)$, $t_u \sim N(0.1, 1)$, and $\eta_{u,i} \sim \epsilon N(0, 1)$; where N is a standard normal density, and ϵ is a noise parameter, we set up $\epsilon = 0.5$. We generate a rating matrix R with 500 users and 500 items, therefore we have 250,000 ratings in total.

4.2 Evaluation

In order to assess the accuracy of the prediction of the RS during the interactive recommendation, we compute the Root Mean Square Error (RMSE). To check the impact of the debiasing mechanism, we compute the Gini coefficient of the popularity scores of all items. The Gini coefficient is used to measure the inequality of a distribution [14]. Let P_i be the popularity of each item after training the model. For a population with n values P_i, $i = 1$ to n, that are indexed in non-decreasing order ($P_{(i)} \leq P_{(i+1)}$), the Gini coefficient can be calculated as follows [14]:

$$G = \left(\frac{\sum_{i=1}^{n}(2i - n - 1)P_{(i)}}{n\sum_{i=1}^{n} P_{(i)}}\right). \tag{10}$$

The higher the Gini coefficient, the more unequal are the values in the dataset. The Gini coefficient of the popularity of items shows how the recommendation system's output affects the exposure distribution of items. Traditional RS's are expected to make popular items become even more popular and non-popular items become even less popular because a traditional recommendation strategy always shows the most relevant items (thus with highest predicted rating), which further divides the haves (popular) from the have-nots (unpopular).

4.3 Method

We first randomly selected 25 items for each of the 500 users from the completed rating matrix and started our initial training. We then select 20% of the ratings as the test set. Note that the testing set is later fixed, and will not be changed. All other ratings are considered as candidate ratings: they are used to simulate the feedback loop of RS and human interaction, and will be added based on the selection mechanism of the recommendation strategy in each iteration/loop. Figure 2 shows the approach for splitting our data. We record the RMSE and the Gini coefficient of predicted item popularity of each testing set.

In each simulation of a feedback loop, we use one of the recommendation strategies listed in Section 3 to recommend items to each user. After that, we simulate the users' response by assuming that they responded to items that are recommended and provide the true ratings for the top-N ratings (top-N=10). After new ratings are taken in, we update the recommender system. We then simulate runs of *Max Feedback Loops* = 20 iterations in Algorithm 1 where a single iteration (or loop) consists of the algorithm providing a recommendation, the user labeling the recommendation, and the algorithm updating its model of the user's preferences.

For each of the recommendation strategies in section 3, we set the number of items (top-N) selected after each recommendation to 10. For the blind spot aware MF and conventional MF with and without AL models, we use stochastic gradient descent to optimize the objective function. For all the Matrix Factorization methods, we set the dimension of the latent space to 20. For all the four propensity based MF algorithms, we optimize the objective function following coordinate gradient descent update rules to learn the model [16]. For all gradient descent optimization updates, we set the number of learning iterations *steps* = 200, regularization weight $\lambda = 0.02$ and learning rate $\eta = 0.002$ for training the matrix factorization model. All the results are from the average of 10 independent runs.

We also report the results for random selection as baseline. This means that the items are selected randomly by the user to rate after the recommendations are made by using the Conventional MF model (meaning essentially an *open loop*).

For the Blind Spot Aware MF 7, We determine the blind spot aware term weight coefficient β by grid search, i.e., running the matrix factorization (Eq. 7) with different $\beta = [0.002, 0.02, 0.05, 0.1, 0.5, 0.8]$ on the initial rating data (before the completion). Finally, we choose β as 0.2 for the Blind Spot Aware term.

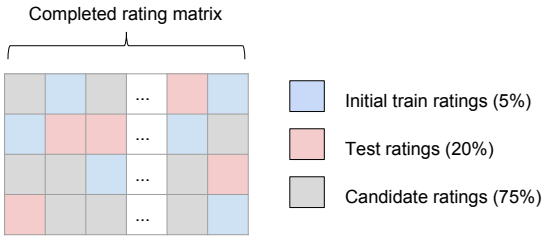

Figure 2: The splitting of the completed rating matrix. The completed rating matrix is split into 3 parts: 1) Initial ratings; 2) Test ratings; and 3) Candidate ratings. The candidate ratings will be added to the training ratings when queried by the system.

4.4 Results

Figure 3 shows the RMSE of each recommendation strategy during the iterative recommender system for the training ratings. The conventional MF and the conventional MF with AL show similar trends in the RMSE (identical on the plot). However, the conventional MF with random selection starts with a high RMSE, but this decreases with each iteration when more ratings are collected. All propensity based MF methods, with or without AL, show a similar trend for the training RMSE as well as the blind spot aware MF. At iteration 1, all six algorithms have the same initial training ratings. The reason why they have large differences on training RMSE lies in the fact that all the propensity MF strategies do not minimize the RMSE; instead they minimize the primary loss term consisting of the square error loss between the prediction $\hat{r}_{ui}$ and the true rating r_{ui}, inversely weighted by $\frac{1}{P_{u,i}}$ (see Eq. 1). On the other hand, conventional MF directly minimizes RMSE, i.e. does not weight this error in the primary loss term (which is equivalent to setting $P_{u,i} = 1$). Note that random baseline means that the items are selected randomly after the recommendations (meaning essentially an open loop).

We also record the RMSE on the testing ratings, as shown in Figure 4. The RMSE for the testing set increases dramatically in the early stages for all propensity based MFs, but then drops to a low level. On the other hand, random selection and conventional MF approaches have a strictly decreasing trend in RMSE with more

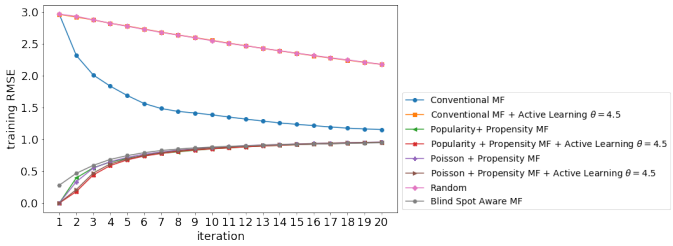

Figure 3: Training primary loss term with different debiasing mechanisms for each iteration on synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the RMSE for the training set. The conventional MF and the conventional MF with AL show a similar trend during each iteration. All four propensity based MFs also show a similar trend as well as the blind spot aware MF. Note that random baseline means that the items are selected randomly after the recommendations (meaning essentially an open loop).

iterations, however with a higher value compared to other algorithms. The blind spot aware MF algorithm has the lowest testing error compared to other algorithms.

Gini Coefficient. As stated in section 2, we computed the Gini coefficient of the item popularity after each feedback loop iteration to assess how different debiasing mechanisms affect the distribution of popularity. A higher Gini index indicates more heterogeneity of the popularity, essentially meaning a bigger popularity divide between the items, leading to a wider gap between the haves and the have-nots. Figure 5 shows the Gini coefficient distribution. Conventional MF increases the inequality of the popularity of items, which indicates that the conventional MF will boost some popular items. On the other hand, the pure Propensity based MF has a high Gini coefficient at an early stage which then decreases with each feedback loop iteration. Propensity based MF with AL results in a low Gini coefficient across all feedback loop iterations. Random selection appears to have the lowest Gini coefficient since all items

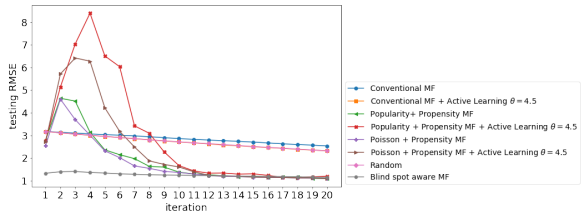

Figure 4: Testing RMSE with different debiasing mechanisms for each iteration on synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the RMSE for the testing set. Conventional MF, with and without AL, show a similar trend during each iteration, and they are similar to random selection. All four propensity based MFs also show a similar trend with lower RMSE, as well as the blind spot aware MF.

have the same probability of being explored. Blind Spot Aware MF has a lower Gini coefficient compared with Conventional MF.

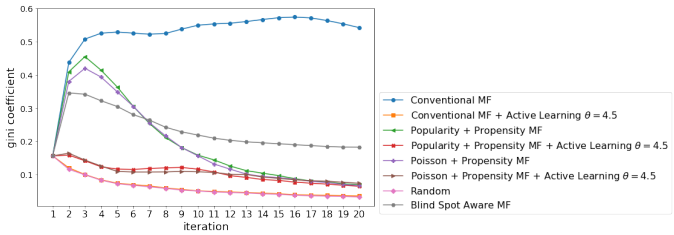

Figure 5: Gini coefficient vs. feedback loop iteration on the synthetic data. The Gini coefficient increases for the Popularity Propensity MF without AL, but then quickly decreases. On the other hand, the Gini coefficient score of the Propensity MF with AL continues to decrease. Note that a higher Gini index indicates more heterogeneity of the popularity, essentially meaning a bigger popularity divide between the items.

Based on the observed trends of the RMSE and the Gini coefficient analysis, we conclude that: 1) Propensity MF achieves a significant level of debiasing on the recommendations in terms of Gini coefficient ($p_{value} < 0.05$); 2) Active Learning combined with Propensity MF results in low testing RMSE and a high debiasing effect on recommendations; 3) Conventional MF increases the popularity bias which eventually affects the ability to discover new items, but decreases the bias when combined with an Active Learning strategy; 4) Blind Spot Aware MF achieves one of the lowest testing RMSE and decreases popularity bias compared to Conventional MF.

5 CONCLUSIONS

Recommender systems introduce bias during the interactive feedback loop with users over time. In this paper, we introduce an interactive framework to simulate the feedback loop that is created by the chain of events generated when a recommender system interacts with users over time. Based on this framework, we proposed several debiasing algorithms based on existing and new techniques to use during this chain of events. We also proposed a blind spot aware matrix factorization algorithm which takes into account the blind spot score when trying to learn the recommendation model.

Note that in this paper, we do not focus on improving collaborative filtering algorithms (Matrix Factorization) for recommender systems by studying user feedback. Instead, our goal is to simulate the interaction between users and the recommender system and to debias the recommender system during the interaction. Our results showed that propensity based MF achieved a certain level of debiasing of the RS while active learning combined with the propensity MF achieved a higher debiasing effect on recommendations. Our proposed blind spot aware matrix factorization also achieved a certain level of debiasing of the RS. One limitation of this study is that we assume that users totally agree with the recommendations in each iteration, and provide feedback. In real-life, users might not agree with recommendations. Thus future study could use a more realistic user reaction model.

ACKNOWLEDGEMENT

This work was supported by National Science Foundation grant NSF-1549981.

REFERENCES

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling Popularity Bias in Learning-to-Rank Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 42–46.

[2] Allison JB Chaney, Brandon M Stewart, and Barbara E Engelhardt. 2017. How Algorithmic Confounding in Recommendation Systems Increases Homogeneity and Decreases Utility. *arXiv preprint arXiv:1710.11214* (2017).

[3] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. 1996. Active learning with statistical models. *Journal of artificial intelligence research* 4, 1 (1996), 129–145.

[4] David F Gleich and Lek-heng Lim. 2011. Rank aggregation via nuclear norm minimization. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 60–68.

[5] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 263–272.

[6] Dawen Liang, Laurent Charlin, and David M Blei. [n. d.]. Causal Inference for Recommendation. ([n. d.]).

[7] Dawen Liang, Laurent Charlin, James McInerney, and David M Blei. 2016. Modeling user exposure in recommendation. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 951–961.

[8] Olfa Nasraoui and Patrick Shafto. 2016. Human-algorithm interaction biases in the big data cycle: A markov chain iterated learning framework. *arXiv preprint arXiv:1608.07895* (2016).

[9] Tien T Nguyen, Pik-Mai Hui, F Maxwell Harper, Loren Terveen, and Joseph A Konstan. 2014. Exploring the filter bubble: the effect of using recommender systems on content diversity. In *Proceedings of the 23rd international conference on World wide web*. ACM, 677–686.

[10] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. *arXiv preprint arXiv:1602.05352* (2016).

[11] Patrick Shafto and Olfa Nasraoui. 2016. Human-recommender systems: From benchmark data to benchmark cognitive models. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 127–130.

[12] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of Exposure in Rankings. *arXiv preprint arXiv:1802.07281* (2018).

[13] Ayan Sinha, David F Gleich, and Karthik Ramani. 2016. Deconvolving feedback loops in recommender systems. In *Advances in Neural Information Processing Systems*. 3243–3251.

[14] Alan Stuart, J Keith Ord, and Sir Maurice Kendall. 1994. *Distribution theory*. Edward Arnold; New York.

[15] Wenlong Sun, Olfa Nasraoui, and Patrick Shafto. 2018. Iterated Algorithmic Bias in the Interactive Machine Learning Process of Information Filtering. (2018).

[16] Stephen J Wright. 2015. Coordinate descent algorithms. *Mathematical Programming* 151, 1 (2015), 3–34.

[17] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin. 2018. Unbiased Offline Recommender Evaluation for Missing-Not-At-Random Implicit Feedback. (2018).

[18] Scott Cheng-Hsin Yang, Jake Alden Whritner, Olfa Nasraoui, and Patrick Shafto. [n. d.]. Unifying recommendation and active learning for human-algorithm interactions. ([n. d.]).