Unbiased Offline Recommender Evaluation for Missing-Not-At-Random Implicit Feedback

Longqi Yang
Cornell Tech, Cornell University
ly283@cornell.edu

Yin Cui
Cornell Tech, Cornell University
yc984@cornell.edu

Yuan Xuan
Cornell Tech, Cornell University
yx424@cornell.edu

Chenyang Wang
Cornell Tech, Cornell University
cw823@cornell.edu

Serge Belongie
Cornell Tech, Cornell University
sjb344@cornell.edu

Deborah Estrin
Cornell Tech, Cornell University
destrin@cornell.edu

ABSTRACT

Implicit-feedback Recommenders (ImplicitRec) leverage positive only user–item interactions, such as clicks, to learn personalized user preferences. Recommenders are often evaluated and compared offline using datasets collected from online platforms. These platforms are subject to popularity bias (i.e., popular items are more likely to be presented and interacted with), and therefore logged ground truth data are Missing-Not-At-Random (MNAR). As a result, the widely used Average-Over-All (AOA) evaluator is biased toward accurately recommending trendy items. In this paper, we (a) investigate evaluation bias of AOA and (b) develop an unbiased and practical offline evaluator for implicit MNAR datasets using the Inverse-Propensity-Scoring (IPS) technique. Through extensive experiments using four real–world datasets and four widely used algorithms, we show that (a) popularity bias is widely manifested in item presentation and interaction; (b) evaluation bias due to MNAR data pervasively exists in most cases where AOA is used to evaluate ImplicitRec; and (c) the unbiased estimator significantly reduces the AOA evaluation bias by more than 30% in the Yahoo! music dataset in terms of the Mean Absolute Error (MAE).

KEYWORDS

Recommendation; Evaluation; Bias; Implicit feedback; Propensity

ACM Reference Format:
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. In *Twelfth ACM Conference on Recommender Systems (RecSys '18), October 2–7, 2018, Vancouver, BC, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3240323.3240355

1 INTRODUCTION

Researchers often evaluate recommendation algorithms using offline datasets because online A/B testing can be very expensive, inefficient, and irreproducible. Unlike other machine learning applications, unbiased evaluation of recommendation performance offline is notoriously challenging because of the biased user feedback collected from online platforms that selectively recommend items. Prior work on **Explicit**-rating **Rec**ommenders (**ExplicitRec**) [12, 30] revealed that users give subjective ratings to items, which results in **M**issing-**N**ot-**A**t-**R**andom (**MNAR**) ground truth data. It has been widely recognized in the literature [12, 16, 18–20] that MNAR rating data can lead to biased conclusions. Therefore, many mechanisms are proposed to debias offline recommender evaluation of rating data [16, 18–20].

However, existing approaches are not directly applicable to implicit user–item interactions (e.g., click, watch, and listen) [6], which are much more prevalent and have been widely used by many state-of-the-art recommendation solutions [3, 5, 29]. Different from explicit ratings (e.g., those based on a Likert scale), implicit feedback signals are one-sided and positive only. In other words, an ideal recommender would never observe user interactions with *irrelevant*[1] items, whereas in ExplicitRec, complete observations assume that each user has a latent preference score for every item. As a result, for **Implicit**-feedback **Rec**ommenders (**ImplicitRec**), it is unclear whether a missing item in a user's history is not favored by the user or has simply not yet been observed.

Existing work simplifies the evaluation of ImplicitRec by assuming that positive signals are **M**issing-**A**t-**R**andom (**MAR**) [4, 5, 11], that is, each favored item is equal-likely to be clicked or viewed by an user. This assumption does not hold in real-world settings because online recommenders manifest popularity bias [2] (popular items are much more likely to be recommended and presented to users). Such a bias leads to the phenomenon that relevant and trendy items are more likely to be interacted with by users. Eventually, the **A**verage-**O**ver-**A**ll (**AOA**) evaluator implicitly places greater weights on the accuracy of serving popular items than on serving long-tail ones. This may overlook key limitations of recommendation algorithms, such as under-serving cold start groups [25], being dominated [2], and exacerbating unhealthful user behavior [24]

In this paper, we develop an unbiased offline recommendation evaluator for MNAR implicit feedback. Our framework is based on the **I**nverse-**P**ropensity-**S**coring (**IPS**) technique used in causal inference [7], which was recently applied to evaluate ExplicitRec [16]. Specifically, we (a) qualitatively and theoretically demonstrate that

RecSys '18, October 2–7, 2018, Vancouver, BC, Canada

ACM ISBN 978-1-4503-5901-6/18/10…$15.00
https://doi.org/10.1145/3240323.3240355

[1] An item is *relevant* to a user if the user is interested in interacting with it (e.g., clicking or viewing it). Otherwise, the item is regarded as *irrelevant.*

the existing evaluation protocol for ImplicitRec is biased; (b) derive unbiased performance estimators for major evaluation metrics, including AUC, DCG, DCG@K, and Recall@K; and (c) conduct extensive experiments using four real-world datasets (citeulike [26], Tradesy [4], Amazon book [13, 28], and Yahoo! music [1]) and four widely used algorithms (BPR [15], PMF [14], U-CML [5], and A-CML [5]). Our experimental results highlight three key contributions and implications of this work:

- The analysis of datasets and trained models (Section 4.2) reveals that popularity bias is widely manifested in item presentation (i.e., popular items are more likely to be presented than long-tail ones) and interaction (i.e., users tend to interact more with popular items). This implies that more attention is needed in considering the potentially negative social and economic impacts of the bias [2, 24].
- The comparisons of the classical AOA evaluator to the unbiased evaluator proposed herein (Sections 4.3 and 4.4) demonstrate that AOA is biased in evaluating most ImplicitRec. The bias may lead to inaccurate judgments of algorithmic improvements and sub-optimal decisions when it comes to model selection.
- The unbiased evaluator significantly reduces AOA evaluation error by more than 30% in the Yahoo! music dataset in terms of the mean absolute error (MAE) (Section 5).

Our code is available at https://github.com/ylongqi/unbiased-offline-recommender-evaluation.

2 RELATED WORK

Our work is inspired by three lines of research: (a) debiasing the evaluation of ExplicitRec; (b) ImplicitRec algorithms and evaluations; and (c) counterfactual evaluation. In this section, we discuss how our work builds upon existing ideas and contributes new knowledge to the field.

2.1 Debiasing the evaluation of ExplicitRec

Previous research has shown that for explicit-feedback recommenders, users' ratings are MNAR [12, 16, 18–20]. This is because people tend to subjectively choose the items they rate, and the selection reflects biases of personal preferences [16] and opinions [12, 20]. To handle MNAR data and conduct unbiased evaluation, previous work assumed that users have latent ratings for every item, and then use popularity [19] or other predictive models [16] to estimate the probability that any given rating is observed. However, such a paradigm is not applicable to implicit feedback because of two fundamental differences: Implicit feedback (a) is available only for the subset of items preferred by users, and (b) is often recorded passively and thus is unlikely to be intentionally controlled.

Our work addresses the unique missing patterns of implicit feedback by extending the IPS framework [16].

2.2 ImplicitRec and evaluation

Recently, there has been a trend toward development of recommenders using implicit feedback signals [6], such as click [5, 26], watch [3], and view [29]. These signals are much richer than ratings. Classical offline evaluation approaches [4, 5, 11, 26, 29] randomly hold out one interacted item per user as a testing set and then report the average performance. Such a paradigm has been shown to be unbiased under MAR feedback [11]. However, MAR signals rarely exist in the real world, because it is very unlikely that a content platform would present items completely at random. In fact, item presentation is usually mediated by recommendation engines, which are subject to popularity bias [2].

Our work points out that under MNAR user feedback, the existing evaluation paradigm is biased. In light of this, we develop a practical and effective technique to address the bias.

2.3 Counterfactual evaluation

Our unbiased evaluator is based on the techniques developed for counterfactual evaluation [7, 21, 23], which aim to evaluate ranking policies offline based on the logs collected from online interactive systems. It has been successfully applied to interactive search [8] and recommendation [10, 23]. Our debiasing framework is built on the **S**elf-**N**ormalized **I**nverse-**P**ropensity-**S**coring (**SNIPS**) estimator proposed by Swaminathan et al. [22].

However, classical counterfactual reasoning operates on interactive logs, for example, (user$_1$, article$_1$, reward$_1$), ..., (user$_n$, article$_n$, reward$_n$), which are different from the implicit feedback-based matrix completion task that we consider. To the best of our knowledge, there has been little research on applying counterfactual estimators to debias ImplicitRec evaluations.

3 UNBIASED RECOMMENDER EVALUATION FOR IMPLICIT FEEDBACK

Recommenders built on implicit feedback receive only users' one-sided (positive) preference signals, such as clicks and watches. Under complete observations, user u has a set of preferred items $\mathcal{S}_u$ among the entire set of items, $\mathcal{I}$ (i.e., $\mathcal{S}_u \subseteq \mathcal{I}$). An ideal recommendation evaluator calculates the following reward $R(\hat{Z})$ for the predicted item ranking $\hat{Z}$.

$$R(\hat{Z}) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{|\mathcal{S}_u|} \sum_{i \in \mathcal{S}_u} c(\hat{Z}_{u,i}), \tag{1}$$

where $\hat{Z}_{u,i}$ is the predicted ranking of item i (among all the items in $\mathcal{I}$) for user u, and the function c denotes any top-N scoring metric, such as Area Under Curve (AUC), Discounted Cumulative Gain (DCG), DCG@K, or Recall@K. These functions are defined as follows:

$$\text{AUC: } c(\hat{Z}_{u,i}) = 1 - \frac{\hat{Z}_{u,i}}{|\mathcal{I}|} \tag{2}$$

$$\text{DCG: } c(\hat{Z}_{u,i}) = \frac{1}{\log_2(\hat{Z}_{u,i} + 1)} \tag{3}$$

$$\text{DCG@K: } c(\hat{Z}_{u,i}) = \frac{\mathbf{1}\{\hat{Z}_{u,i} \le K\}}{\log_2(\hat{Z}_{u,i} + 1)} \tag{4}$$

$$\text{Recall@K: } c(\hat{Z}_{u,i}) = \mathbf{1}\{\hat{Z}_{u,i} \le K\} \tag{5}$$

Eqn. 1 measures idealistic recommendation performance, which assumes that users would go through all items in the system and interact with every one that appeals to them. From a practical standpoint, it is impossible to browse and judge millions or billions of items. As a result, recommenders have access to only a partial view of $\mathcal{S}_u$, denoted by $\mathcal{S}_u^*$. For each positive signal $(u, i), i \in \mathcal{S}_u$, we use $O_{u,i}$ to indicate whether (u, i) is observed ($O_{u,i} = 1$ if (u, i) is observed, and $O_{u,i} = 0$ otherwise). In addition, inspired by [16], we

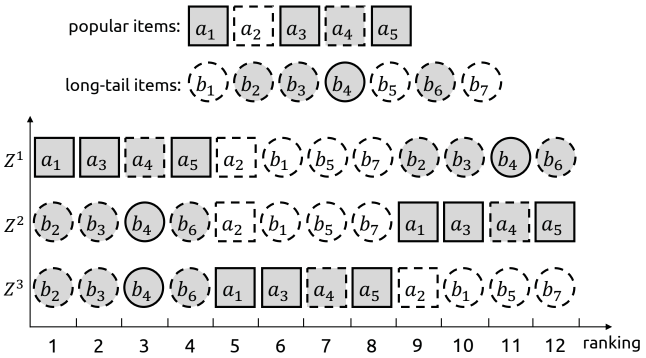
popular items:
a_1 a_2 a_3 a_4 a_5
long-tail items:
b_1 b_2 b_3 b_4 b_5 b_6 b_7
Z^1
a_1 a_3 a_4 a_5 a_2 b_1 b_5 b_7 b_2 b_3 b_4 b_6
Z^2
b_2 b_3 b_4 b_6 a_2 b_1 b_5 b_7 a_1 a_3 a_4 a_5
Z^3
b_2 b_3 b_4 b_6 a_1 a_3 a_4 a_5 a_2 b_1 b_5 b_7
1 2 3 4 5 6 7 8 9 10 11 12
ranking

Figure 1: A hypothetical example to illustrate the evaluation bias that results from use of the AOA evaluator. Three recommenders generated distinct lists of recommendations, Z^1, Z^2 and Z^3, for the same user. Among the shaded items that were preferred by the user, the ones with a solid border were observed by recommenders. The performance was measured by DCG, and the results are presented in Table 1.

Table 1: The true and estimated DCG values for three recommenders in Fig. 1. $R(\hat{Z})$ denotes the ground truth, and $\hat{R}_{\text{AOA}}(\hat{Z})$ denotes the AOA estimations. The AOA estimator outputs larger values when popular items are ranked higher.

Estimator	Z^1	Z^2	Z^3
$R(\hat{Z})$	0.463	0.463	0.494
$\hat{R}_{\text{AOA}}(\hat{Z})$	0.585	0.340	0.390

assume the observations of every signal to be Bernoulli distributed, that is, $O_{u,i} \sim \mathcal{B}(1, P_{u,i})$, where with probability $P_{u,i} = P(O_{u,i} = 1)$, (u, i) is observed by a recommender.

In reality, the partial view $\mathcal{S}_u^*$ is mostly biased and the implicit feedback is MNAR. In Section 3.1, we show that the AOA evaluator, which is widely used in the existing literature, is biased, and in Section 3.2 we propose an unbiased evaluator based on the inverse-propensity-scoring (IPS) technique [16].

3.1 Average-over-all (AOA) evaluator

In prior literature, $R(\hat{Z})$ was estimated by taking the average over all observed user feedback $\mathcal{S}_u^*$:

$$\begin{aligned}\hat{R}_{\text{AOA}}(\hat{Z}) &= \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{|\mathcal{S}_u^*|}\sum_{i\in\mathcal{S}_u^*} c(\hat{Z}_{u,i}) \\ &= \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{\sum_{i\in\mathcal{S}_u} O_{u,i}}\sum_{i\in\mathcal{S}_u} c(\hat{Z}_{u,i})\cdot O_{u,i}\end{aligned} \tag{6}$$

To intuitively illustrate the bias of the AOA evaluator, we considered a hypothetical platform that served 12 items, as shown in Fig. 1. We divided the items into two groups based on the number of interactions they received: popular items ($a_1, ..., a_5$) and long-tail items ($b_1, ..., b_7$). For a specific user, three different recommenders generated distinct ranked lists, Z^1, Z^2, and Z^3, based on the predicted user preferences. Each item on the platform was either relevant (shaded) or irrelevant (blank) to the user. Among all the relevant items, only feedback for a partial set was observed (solid border). To encode the popularity bias manifested in ImplicitRec (i.e., user interactions with popular items are more likely to be observed), we assumed that among the relevant items, 75% of the popular items and 25% of the long-tail items were interacted with. In addition, three ranked lists were strategically designed: The Z^1 and Z^2 ranked lists had the same *true performance* on the ranking of *relevant* items but differed on the serving of the popular and long-tail groups. The Z^3 ranked list achieved the best *true performance*.

We calculated the DCG scores (eqn. 3) for three recommenders using the AOA evaluator (eqn. 6) and compared the scores to the true performances (eqn. 1). According to the results presented in Table 1, Z^1 was evaluated as much more accurate than Z^2 and Z^3, despite the fact that, in reality, Z^2 had the same performance as Z^1, and Z^3 performed much better. This demonstrates that the AOA evaluator is significantly biased toward the accuracy of serving trendy items; that is, the estimated $\hat{R}_{\text{AOA}}(\hat{Z})$ is larger if popular items are ranked higher. The conclusions made based on such empirical evidence result in incorrect and even opposite judgments of the relative utilities of recommenders.

Basically, the expected outcome of the AOA evaluator does not conform to the true performance, that is, $\mathbb{E}_O\left[\hat{R}_{\text{AOA}}(\hat{Z})\right] \neq R(\hat{Z})$. We prove this inequivalence by a counterexample. Suppose that for any user u, among all relevant items ($\mathcal{S}_u$), only one item $k^u \in \mathcal{S}_u$ has an observation probability close to 1, so that $P(O_{u,k^u}) = 1 - \epsilon$; whereas for the other items, $P(O_{u,i}) = \epsilon, i \in \mathcal{S}_u \backslash \{k^u\}$. In this case, $\mathbb{E}_O\left[\hat{R}_{\text{AOA}}(\hat{Z})\right] \approx_{\epsilon \ll 1} \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} c(\hat{Z}_{u,k^u}) \neq R(\hat{Z})$. Next, we present our proposed unbiased performance evaluator as an alternative to the existing AOA evaluator.

3.2 Unbiased evaluator

To conduct unbiased evaluation of biased observations, we leverage the IPS framework [16, 22] that weights each observation with the inverse of its propensity, where the term *propensity* refers to the tendency or the likelihood of an event happening. The intuition is to down-weight the commonly observed interactions, while up-weighting the rare ones. In the context of this paper, the probability $P_{u,i}$ is treated as the pointwise propensity score. Therefore, the IPS unbiased evaluator is defined as follows:

$$\begin{aligned}\hat{R}_{\text{IPS}}(\hat{Z}|P) &= \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{|\mathcal{S}_u|}\sum_{i\in\mathcal{S}_u^*}\frac{c(\hat{Z}_{u,i})}{P_{u,i}} \\ &= \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{|\mathcal{S}_u|}\sum_{i\in\mathcal{S}_u}\frac{c(\hat{Z}_{u,i})}{P_{u,i}}\cdot O_{u,i}\end{aligned} \tag{7}$$

We prove that given any propensity assignment P, $\hat{R}_{\text{IPS}}(\hat{Z}|P)$ is an unbiased estimator.

$$\begin{aligned}\mathbb{E}_O\left[\hat{R}_{\text{IPS}}(\hat{Z}|P)\right] &= \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{|\mathcal{S}_u|}\sum_{i\in\mathcal{S}_u}\frac{c(\hat{Z}_{u,i})}{P_{u,i}}\cdot\mathbb{E}_O\left[O_{u,i}\right] \\ &= \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{|\mathcal{S}_u|}\sum_{i\in\mathcal{S}_u} c(\hat{Z}_{u,i}) = R(\hat{Z})\end{aligned} \tag{8}$$

Furthermore, to estimate $|\mathcal{S}_u|$ and control the variability of the IPS evaluator, we leverage the control variates [16, 22] to derive a Self-Normalized Inverse-Propensity-Scoring (SNIPS) evaluator. According to the theory of Monte Carlo approximation [22], the estimation $\hat{W}$ of the expectation $\mathbb{E}_X[W(X)]$ has a lower variance if a multiplicative control variate $V(X)$ with known expectation $\mathbb{E}_X[V(X)] = v \neq 0$ is introduced, that is, if $\hat{W}$ is calculated as: $\hat{W} = \frac{\sum_{j=1}^{n} W(X_j)}{\sum_{j=1}^{n} V(X_j)} v$. While $\hat{W}$ is not a completely unbiased estimator, it strongly converges to the true expectation for large n [22].

In the context of the IPS evaluator, because $\mathbb{E}_O\left[\sum_{i \in \mathcal{S}_u^*} \frac{1}{P_{u,i}}\right] = \mathbb{E}_O\left[\sum_{i \in \mathcal{S}_u} \frac{1}{P_{u,i}} \cdot O_{u,i}\right] = |\mathcal{S}_u|$, we can write the SNIPS evaluation as follows:

$$\begin{aligned}\hat{R}_{\text{SNIPS}}(\hat{Z}|P) &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{|\mathcal{S}_u|} \frac{\mathbb{E}_O\left[\sum_{i \in \mathcal{S}_u^*} \frac{1}{P_{u,i}}\right]}{\sum_{i \in \mathcal{S}_u^*} \frac{1}{P_{u,i}}} \sum_{i \in \mathcal{S}_u^*} \frac{c(\hat{Z}_{u,i})}{P_{u,i}} \\ &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{\sum_{i \in \mathcal{S}_u^*} \frac{1}{P_{u,i}}} \sum_{i \in \mathcal{S}_u^*} \frac{c(\hat{Z}_{u,i})}{P_{u,i}}\end{aligned} \tag{9}$$

A key challenge in computing $\hat{R}_{\text{SNIPS}}(\hat{Z}|P)$ is to predict the propensity scores $P_{u,i}$. Next, we demonstrate our method, which estimates the propensity scores based solely on raw observations, without requiring any auxiliary user or item information.

3.3 Estimating propensity scores

We assume that the propensity score $P_{u,i}$ is user independent, that is, $P_{u,i} = P(O_{u,i} = 1) = P(O_{*,i} = 1) = P_{*,i}$. This simplified assumption is made to address the lack of auxiliary user information in many user–item interaction records.[2] We derive $P_{*,i}$ by constructing a **two-step generative process** of user–item interactions: (1) **Select**, where a recommender system selects a set of items to present to a user; and (2) **Interact**, where the user browses the recommended items and interacts with the ones she likes. Therefore, $P_{*,i}$ can be calculated as follows:

$$P_{*,i} = P_{*,i}^{\text{select}} \cdot P_{*,i}^{\text{interact}|\text{select}}, \tag{10}$$

where $P_{*,i}^{\text{select}}$ is the probability that item i is recommended and $P_{*,i}^{\text{interact}|\text{select}}$ is the conditional probability that the user interacts with item i given that it is recommended.

Since implicit feedback is passively recorded and is less likely to be subjectively manipulated, we assume that $P_{*,i}^{\text{interact}|\text{select}} = P_{*,i}^{\text{interact}}$, that is, the user interacts with all the items she likes in the recommended set, and the user's preferences are not affected by recommendations.[3] Also, because $P_{*,i}^{\text{interact}}$ is user independent, it is proportional to only the item's *true popularity* n_i (the number of occurrences in the *complete observation*):

$$\hat{P}_{*,i}^{\text{interact}} \propto n_i \tag{11}$$

Because items that are frequently interacted with are more likely to be recommended in ImplicitRec [2], the probability $P_{*,i}^{\text{select}}$ is modeled using n_i^* (the number of times item i is interacted with) as a covariate. Specifically, we follow a common template that accurately captures the popularity bias [19], which assumes that $P_{*,i}^{\text{select}}$ conforms to a power-law distribution parameterized by γ:

$$\hat{P}_{*,i}^{\text{select}} \propto (n_i^*)^{\gamma} \tag{12}$$

Therefore, according to the constructed generation process, $\hat{P}_{*,i}$ depends on only two variates, n_i^* and n_i:

$$\hat{P}_{*,i} \propto (n_i^*)^{\gamma} \cdot n_i, \tag{13}$$

where $n_i = \sum_{u \in \mathcal{U}} \mathbf{1}[i \in \mathcal{S}_u]$ and $n_i^* = \sum_{u \in \mathcal{U}, i \in \mathcal{S}_u^*} O_{*,i}$.

However, empirically, n_i is not directly observable. To address this problem, we observe that n_i^* is sampled from a binomial distribution[4] parameterized by n_i, that is, $n_i^* \sim \mathcal{B}(n_i, P_{*,i})$. Therefore, a relationship between n_i and n_i^* can be built by bridging the generative model (eqn. 13) with the following unbiased estimator:

$$\hat{P}_{*,i} = \frac{n_i^*}{n_i} \propto (n_i^*)^{\gamma} \cdot n_i \tag{14}$$

Therefore, $n_i \propto (n_i^*)^{\frac{1-\gamma}{2}}$. We use this as a replacement for the unobserved n_i in eqn. 13, which results in an unbiased $\hat{P}_{*,i}$ estimator that is determined by only the empirical counts of items:

$$\hat{P}_{*,i} \propto \left(n_i^*\right)^{\left(\frac{\gamma+1}{2}\right)} \tag{15}$$

Different values of the power-law exponent γ affect the propensity distributions over items with different observed popularity levels. A larger γ leads to lower propensity scores for long-tail items and higher scores for popular ones. In deployed systems, the exponent can be empirically predicted (as shown in Section 4.3).

4 EXPERIMENTS WITH BIASED FEEDBACK AND THE UNBIASED EVALUATOR

To more thoroughly understand the nature of MNAR implicit feedback and the proposed unbiased evaluator, we studied three large-scale real-world datasets and four recommendation algorithms. Our experiments are comprised of three parts: (a) investigating how popularity bias is manifested in real-world platforms, (b) exploring properties of the power-law exponent, and (c) understanding debiasing effects of the unbiased evaluator.

4.1 Experimental setup

To describe the stup of the experiments, we review the datasets and algorithms, describe the recommendation model implementations with OpenRec [28], and present the details of model training.

4.1.1 Datasets. We used three datasets of varied size and sparsity ($\frac{\text{\# interactions}}{\text{\# users} \times \text{\# items}}$). For each dataset, we randomly and independently hold out 15% of user–item interactions for validation and 15% for testing, and we used the remaining 70% of records for training. During testing, we excluded cold-start users and items that have no record in the training set.

- **citeulike** [26]. citeulike is a reference management service, where scholars curate article collections based on their preferences and professional needs. We used the dataset collected by Wang et

[2] This assumption may be relaxed in cases where auxiliary user information is available. We discuss this issue in Section 6.

[3] In reality, user–item interactions may be affected by the order of presentation of the items, and users' preferences may be shaped by recommendations in the long term. Modeling these effects may further improve the evaluator's performance (as discussed in Section 6).

[4] $O_{*,i}$ satisfies the Bernoulli distribution.

al. [26] and treated "saving an article" as a positive implicit feedback signal. The dataset contains 204,986 interactions between 5,551 users and 16,980 items (sparsity: 2e-3).

- **Tradesy** [4]. Tradesy is a large second-hand retail market for clothing and fashion. We used the dataset released by He et al. [4], and treated "want an item" and "bought an item" as positive signals. The final dataset includes 19,243 users, 165,906 wanted or bought items, and 394,421 interactions (sparsity: 1e-4).
- **Amazon book** [13, 28]. The Amazon book dataset was derived from the original Amazon review dataset [13] by Yang et al. [28]. The dataset records users' purchasing history under the Amazon book category. The dataset covers 99,473 users, 450,166 books, and 996,938 transactions (sparsity: 2e-5).

4.1.2 Algorithms. We considered recommendation models with different training procedures (pairwise and pointwise) and architectures (matrix-factorization based and metric-learning based).

- **Bayesian Personalized Ranking (BPR)** [15]. BPR is based on the general framework of matrix factorization that learns vector representations for users and items. Specifically, user u's preference toward item i is modeled as $\hat{x}_{u,i} = v_u^T v_i + \beta_i$, where v_* denote representations, and β_i denotes the item-specific bias. Built upon the scoring function $\hat{x}_{u,i}$, BPR trains the model parameters on (u, i, j) triplets (i and j represent interacted item and non-interacted item respectively) using a pairwise ranking based optimization framework that minimizes the following loss.

$$\min_{\Theta} \sum_{(u,i,j)\in\mathcal{D}} -\ln(\hat{x}_{u,i} - \hat{x}_{u,j}) + \lambda_\Theta \|\Theta\| \tag{16}$$

where D is the set of triplets that are *randomly* sampled from the training dataset and Θ is the set of model parameters.

- **Collaborative Metric Learning with Uniform Weights (U-CML)** [5]. U-CML is trained on the same (u, i, j) triplets as BPR, but instead of modeling user–item scores using dot products, U-CML leverages the Euclidean distance metric to regularize the embedding space, that is, $\hat{x}_{u,i} = \beta_i - \|v_u - v_i\|^2$, where all representations are bounded within a unit sphere. Another difference between U-CML and BPR is that U-CML minimizes the pairwise hinge loss:

$$\min_{\Theta} \sum_{(u,i,j)\in\mathcal{D}} \left[m + \hat{x}_{u,i} - \hat{x}_{u,j}\right]_+ + \lambda_\Theta \|\Theta\|^2 \tag{17}$$

- **CML with Approximate-Rank Weights (A-CML)**. U-CML model randomly samples the triplets from the training set, making most of them become trivial samples as the training proceeds. Therefore, as suggested by Hsieh et al. [5], we leveraged the approximate-rank weighting technique [27] to adjust the weight of each training instance:

$$\min_{\Theta} \sum_{(u,i,j)\in\mathcal{D}} w_{u,j} \left[m + \hat{x}_{u,i} - \hat{x}_{u,j}\right]_+ + \lambda_\Theta \|\Theta\|^2, \tag{18}$$

where $w_{u,j} = \log(\text{rank}(u, j) + 1)$ and rank(u, j) is the rank of item j in user u's recommendation list. The rank can be estimated by sequential [27] or parallel [5] sampling. To speed up the training, we sampled 10 negative items in parallel for each observed user–item interaction, as suggested by Hsieh et al. [5].

- **Probabilistic Matrix Factorization (PMF)** [14]. PMF is a pointwise trained recommendation model, that is, it is built upon pairs (u, i). The model is optimized to minimize the following regularized square error:

$$\min_{\Theta} \sum_{u,i} c_{u,i}(r_{u,i} - \hat{x}_{u,i})^2 + \lambda_\Theta \|\Theta\|^2, \tag{19}$$

where $r_{u,i} = 1$ if user u interacted with item i, and $r_{u,i} = 0$ otherwise. Because of the sparsity of the interactions, $c_{u,i}$ is set to a higher value for $r_{u,i} = 1$ than for $r_{u,i} = 0$. In our experiments, $c_{u,i}$ was set to 1 and 0.25, respectively, for those two cases.

4.1.3 Implementations and training. We implemented the algorithms based on the OpenRec framework [28]. The dimensionality of user and item representations was set to 50 for citeulike and to 100 for the other datasets. Each model was trained using the Adam optimizer [9] with a batch size of 8K. Because of differences in the sizes of the datasets, the models were trained for 50K, 120K, and 200K iterations[5] under citeulike, tradesy, and Amazon book, respectively. We conducted *model selection* [16] for each algorithm–metric pair by training recommenders with different regularization parameters, that is, $\lambda_\Theta \in \{0.1, 0.01, 0.001, 1e-4, 1e-5\}$. The optimal training iteration and λ_Θ value are determined by the evaluation on the validation set. The recommendation performances are finally reported on the held-out testing sets. Because of the large item space, it is computationally infeasible to compute rankings over all items. Therefore, for each user, we randomly and independently sample 200 items with which users have not interacted before and compute rankings over the sampled sets. This is a common approach adopted by recent literature [28].

4.2 Investigating popularity bias

We initially conducted an experiment to understand *to what extent popularity bias is manifested in real-world recommendation systems*. Specifically, we investigated two kinds of bias related to popularity: (a) **interaction bias** (i.e., that users tend to interact more often with popular items), and (b) **presentation bias** (i.e., that recommenders unfairly present more popular items than long-tail ones).

However, in existing datasets, interaction bias is barely separable from presentation bias [17], since a user can interact with an item only if it is presented. Therefore, we resorted to the joint effects of the two kinds of bias, which are manifested in the distribution of n_i^*, that is, the number of times users interact with each item. Intuitively, an unbiased platform should expect users to interact broadly. As a result, user attentions are likely to be evenly distributed. On the contrary, if a platform is highly biased, then user interactions tend to be more concentrated, which leads to dominance by a small set of items. We show the n_i^* distribution for all $i \in \mathcal{I}$ in Fig. 2. Given that the horizontal axis is log scaled, the n_i^* distribution is significantly skewed: Most of the items received very few user interactions. For example, on Amazon book, more than 99.9% of items received fewer than 100 interactions. In addition, the degree of bias varies across datasets: The Amazon book dataset is the most popularity biased, while the tradesy dataset is the least popularity biased.

For the presentation bias, we measured the average number of times that an item with the observed popularity $n^* \in [1, \max(n_i^*)]$ was recommended, denoted by $f(n^*)$. An unbiased system should

[5] An iteration is defined as a feed forward and a backward propagation using a batch (size=8K) of randomly sampled training data.

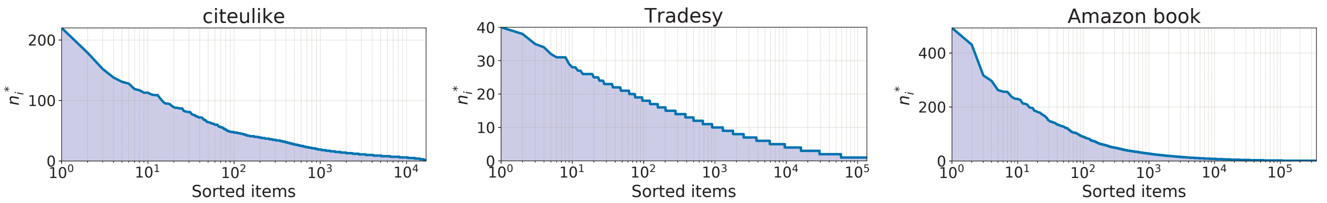

Figure 2: The distribution of n_i^* (the observed number of interactions with item i) in the three datasets. The items are presented in descending order of n_i^*. The horizontal axis is log scaled for better visualization. In all datasets, the n_i^* distribution is skewed and the user interactions are significantly biased.

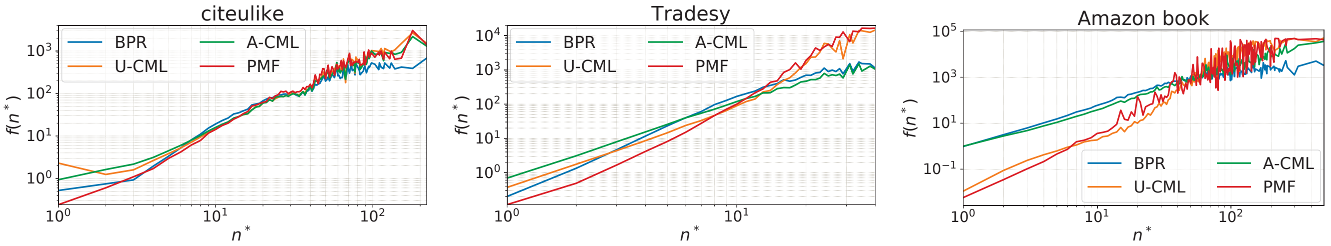

Figure 3: Empirically estimated $f(n^*)$ on the three datasets and the four recommendation algorithms. $f(n^*)$ denotes the average number of times that an item with observed popularity n^* was recommended. Both axes are log scaled. Therefore, exponential growth is linear in the figure. All settings manifest significant presentation bias.

expect a relatively flat $f(n^*)$ with a small slope, whereas a biased recommender may produce linearly or exponentially growing $f(n^*)$. We treated the top 50 recommendations that the trained recommenders made for every user as recommended items, and $f(n^*)$ was computed as follows:

$$f(n^*) = \frac{\sum_{i \in \mathcal{I}} \mathbf{1}(n_i^* = n^*) \cdot N_i}{\sum_{i \in \mathcal{I}} \mathbf{1}(n_i^* = n^*)}, \tag{20}$$

where N_i is the frequency of item i in all users' top 50 recommendations. For each user, the recommendation list was computed over the complete item set $\mathcal{I}$, excluding items that the user had already interacted with in the training set. In Fig. 3, we show the empirically estimated $f(n^*)$. All three $f(n^*)$ curves appear to be mostly monotonic, with small variations, which suggests that an item with small n_i^* is much less likely to be presented, compared to the ones with larger n_i^*. Also, different algorithms tend to manifest diverse patterns. For example, in Amazon book, BPR and A-CML are more likely to present long-tail items than PMF and U-CML.

To sum up the findings, we demonstrated that both forms of popularity bias pervasively exist on platforms that use the mainstream recommendation algorithms. Although the amount of bias varies across platforms and algorithms, it appears to be highly significant. In addition, the estimation of presentation bias provides a mechanism for gaining an empirical understanding of the properties of the power-law exponent (eqn. 15), which is discussed next.

Table 2: Estimated γ value for every dataset-algorithm pair. The algorithm that achieves the lowest γ in each dataset is bolded. The γ estimation is more sensitive to the choice of datasets than algorithms.

Dataset	BPR	U-CML	A-CML	PMF	Average
citeulike	1.67	1.64	**1.55**	1.89	1.69
Tradesy	2.96	2.40	**2.25**	3.07	2.67
Amazon book	1.85	2.11	**1.70**	1.80	1.87

4.3 Exploring the power-law exponent

To understand the properties of γ, we estimated its value by running simulations on offline datasets. The shape of the probability distribution $\hat{P}_{*,i}^{\text{select}}$, parameterized by γ, was most likely to be affected by two factors: the recommendation algorithm (which controls *what to select*) and the content platform (which determines *what is available*). Therefore, we predict a γ for each algorithm–platform pair. Due to the fact that $\hat{P}_{*,i}^{\text{select}}$ is only determined by an item's observed popularity n_i^*, the probability satisfies: $\hat{P}_{*,i}^{\text{select}} \propto (n_i^*)^\gamma \propto f(n^* = n_i^*)$. Estimating the value of γ is equivalent to minimizing the following square error:

$$\min_{\gamma} \sum_{(x,y) \in \mathcal{T}} \left(\log\left(\frac{f(y)}{f(x)}\right) - \gamma \cdot \log\left(\frac{y}{x}\right) \right)^2 \tag{21}$$

where $\mathcal{T}$ denotes all possible combinations of (x, y) where $x, y \in [1, \max(n_i^*)]$ and $x \neq y$. Because this is a quadratic optimization problem, γ can be analytically solved as:

$$\gamma = \frac{\sum_{(x,y)\in\mathcal{T}} \log\left(\frac{f(y)}{f(x)}\right)\cdot\log\left(\frac{y}{x}\right)}{\sum_{(x,y)\in\mathcal{T}}\left(\log\left(\frac{y}{x}\right)\right)^2} \tag{22}$$

We fit γ using the calculated $f(u^*)$ from Section 4.2. To make the estimation numerically more stable and robust to outliers, we exclude the top 0.5% of items that have the highest n^*. The final estimated γ values are presented in Table 2. We find that the power-law curve accurately fits $f(n^*)$ with small average square error (within the range $(0.001, 0.02)$). Also, among all algorithms, A-CML stands out (bolded in Table 2) as having the lowest estimated γ value in all datasets, which suggests that it manifests the least presentation bias. However, overall, the estimated γ value is relatively stable given a dataset (the value range is 0.34, 0.82, and 0.41 for the citeulike, Tradesy, and Amazon book datasets, respectively).

These experimental results suggest that in practice, if the past recommendation algorithm is known, using the power-law function can accurately fit and reconstruct $\hat{P}^{\text{select}}_{*,i}$. Even if the accurate recommender is unknown, it is still plausible to roughly predict the γ value by experimenting classical algorithms in the given dataset. In the next experiment, we leverage the estimated γ value to understand debiasing effects of the unbiased evaluator.

4.4 Understanding the unbiased evaluator

We compare the outputs from the AOA and the unbiased evaluator under the same algorithm–platform settings. Specifically, for each dataset, we experiment on the *minimum*, *average* and *maximum* γ values from Table 2. We evaluate models against four metrics: AUC, DCG, DCG@5 and Recall@5, as defined from eqn. 2 to eqn. 5. The experimental results are presented in Fig. 4. Our main findings are discussed below.

- **The unbiased evaluator reports lower performance, regardless of the algorithm, dataset or evaluation metric**. As shown in Fig. 4, after applying the unbiased evaluation, the estimated recommendation performance significantly drops. This is because recommenders usually perform worse on long tail items than popular ones, and the unbiased evaluator corrects and reduces the biased weights that AOA places on popular items. This finding reveals that *the traditional evaluation method may over-estimate the performance of recommendation algorithms.*
- **The unbiased evaluator may amplify, diminish, or flip the relative differences reported by AOA.** In many cases, the unbiased estimator does not change the absolute performance difference between algorithms but amplifies the relative difference, e.g., BPR outperforms PMF by 22% and 26% in terms of the Recall reported by AOA and γ(min), respectively. Also, the unbiased evaluator may diminish (e.g., U-CML vs. BPR under Amazon book-DCG) or flip (e.g., PMF vs. U-CML under Tradesy-DCG) the relative differences. These observations highlight a caveat that *traditional evaluation may lead to inaccurate or mis-judgments of algorithms' relative utilities.*
- **The outputs of the unbiased estimator are stable for different γ values from the estimated range.** In all conditions, the outputs of the unbiased evaluator are stable for differnt γ values (min, avg., or max). In other words, as long as the γ value is from the estimated range, the unbiased evaluator is expected to produce robust evaluation results.

In summary, these results demonstrate that the unbiased evaluator is robust and has the potential to more objectively evaluate and compare different recommenders. Next, we empirically measure its debiasing performance.

5 EVALUATING DEBIASING PERFORMANCE

We leverage the Yahoo! music ratings dataset [1] to quantify debiasing performance of the unbiased evaluator. The dataset contains users' ratings towards a uniform-randomly selected sets of music, which can be used to measure recommenders' true performances.

5.1 Experimental setup

The original dataset includes a training set and a testing set. The training set contains 300K ratings given by 15.4K users against 1K songs through natural interactions, and the testing set is collected by asking a subset of 5.4K users to rate 10 randomly selected songs. To tailor this dataset for experimenting implicit feedback, we treat items rated greater than or equal to 4 as relevant, and others as irrelevant, as suggested by prior literature [5]. We filter the testing set by retaining users who have at least a relevant and an irrelevant song in the testing set and two relevant songs in the training set (2,296 users satisfy these requirements). We additionally held out a biased testing set (**biased-testing**) from the training set by randomly sampling 300 songs for each user.

We train models discussed in Section 4.1 using the same protocol but with fixed hyperparameters ($\lambda_\Theta = 0.001$, training iterations: 10K, latent factors: 50). For each model, different evaluators are used to evaluate its performance against the biased-testing set in terms of AUC and Recall.[6] The models' true performances were calculated by AOA over the unbiased testing set.

5.2 Results

Table 3 shows the mean absolute error (MAE) between different evaluators' outputs on the biased-testing set and the recommenders' true performances. For both AUC and Recall, the unbiased evaluator (UB) reduced more than 30% of the errors in AOA, and UB's debiasing performance was insensitive to the hyperparameter selections. Within the range of $[1.5, 3.0]$, UB consistently produced significantly lower errors than AOA. However, these results also demonstrate that UB is still imperfect, and that there is ample room for future improvements.

6 CONCLUSION AND DISCUSSION

We studied the problem of evaluating ImplicitRec using offline datasets and showed that the widely adopted AOA evaluation is biased toward popularity. Built upon the IPS technique from causal inference, we developed a theoretically grounded unbiased evaluator and empirically demonstrated its ability to significantly reduce recommender evaluation biases. However, the developed unbiased evaluator is limited in its two simplified assumptions, which points out promising future research directions:

[6] Recall@30 (biased-testing set) and Recall@1 (testing set) were compared since the biased-testing set is 10 times as large as the testing set.

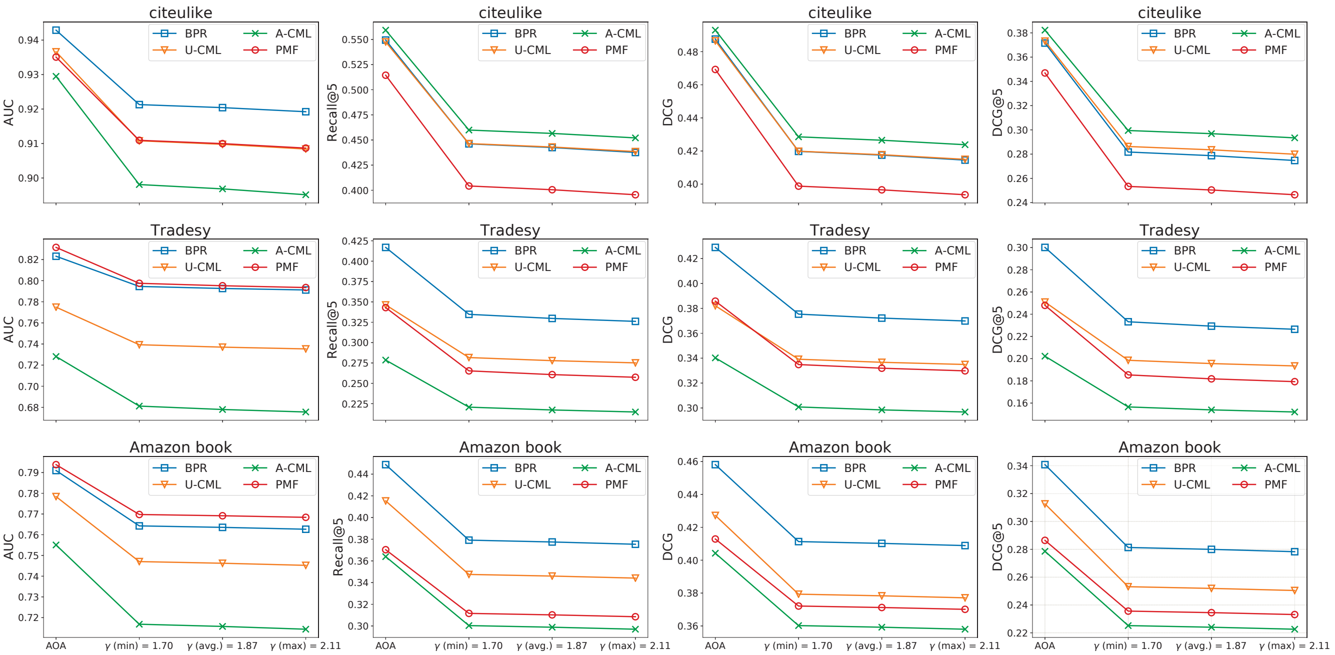

Figure 4: Comparison of the traditional and unbiased evaluators in measuring the performance of four recommendation algorithms. The evaluations were conducted over three datasets using four metrics. Each sub-figure represents a specific dataset–metric pair. For the unbiased evaluator, three estimated γ values from Section 4.3 were used in the experiments. The unbiased evaluator significantly reduces the biased weights that the AOA method places on the popular items and produces robust and consistent results for any γ from the estimated range.

Table 3: Mean absolute error (MAE) between evaluators’ outputs on the biased-testing set and recommenders’ true performances. Performance was measured against AUC and Recall. For the unbiased evaluator (UB), four γ values were used in the experiments (γ = 1.5, 2.0, 2.5, 3.0).

(a) Mean absolute error (MAE) on AUC

Model	AOA	UB(1.5)	UB(2.0)	UB(2.5)	UB(3.0)
U-CML	0.151	0.102	0.099	0.096	0.094
A-CML	0.152	0.103	0.099	0.097	0.094
BPR	0.147	0.109	0.106	0.104	0.103
PMF	0.148	0.103	0.100	0.097	0.095

(b) Mean absolute error (MAE) on Recall

Model	AOA	UB(1.5)	UB(2.0)	UB(2.5)	UB(3.0)
U-CML	0.401	0.270	0.260	0.253	0.248
A-CML	0.399	0.274	0.264	0.258	0.253
BPR	0.380	0.275	0.268	0.262	0.258
PMF	0.386	0.267	0.259	0.252	0.248

- **User-independent propensity.** In the absence of detailed meta-information about users, we assumed that the propensity was user independent and that the probability of an item being presented was determined by its observed popularity. In reality, the propensity may be affected by user-specific traits and preferences. Future research could investigate more sophisticated propensity estimation methods, such as building predictive models to take auxiliary user features into consideration.
- **Selection-independent interaction.** We assumed that the probability that a user interacts with an item is independent of the probability that the item is recommended. This does not capture the potential impact of recommendations and item presentation order on users’ preferences. Future research could conduct controlled user testing to model these nuanced effects.

In addition, our work has implications for the development of recommendation algorithms that are robust to popularity bias. This work shows that a recommender’s accuracy on popular items usually overestimates that recommender’s true performance. Algorithms that intend to be robust to popularity bias should explore ways to improve long-tail recommendations, not only through popularity under-weighting, but also via other techniques such as stratified sampling, data augmentation, and low-shot learning.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments and suggestions. This research was funded by the National Science Foundation (#1700832) and Oath (the Connected Experiences Laboratory at Cornell Tech). The work was further supported by the small data lab at Cornell Tech, which receives funding from NSF, NIH, RWJF, UnitedHealth Group, Google, and Adobe.

REFERENCES

[1] 2006. *Yahoo! Webscope dataset ydata-ymusic-rating-study-v1.* http://research.yahoo.com/Academic_Relations

[2] 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.

[3] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. 2010. The YouTube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 293–296.

[4] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback.. In *AAAI*. 144–150.

[5] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative metric learning. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 193–201.

[6] 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.

[7] T. Joachims and A. Swaminathan. 2016. Tutorial on Counterfactual Evaluation and Learning for Search, Recommendation and Ad Placement. In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*. 1199–1201.

[8] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 781–789.

[9] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[10] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 297–306.

[11] Daryl Lim, Julian McAuley, and Gert Lanckriet. 2015. Top-n recommendation with missing implicit feedback. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 309–312.

[12] Benjamin M Marlin and Richard S Zemel. 2009. Collaborative prediction and ranking with non-random missing data. In *Proceedings of the third ACM conference on Recommender systems*. ACM, 5–12.

[13] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 785–794.

[14] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[15] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[16] 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*. 1670–1679.

[17] 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.

[18] Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 713–722.

[19] Harald Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 125–132.

[20] Harald Steck. 2013. Evaluation of recommendations: rating-prediction and ranking. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 213–220.

[21] Adith Swaminathan and Thorsten Joachims. 2015. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*. 814–823.

[22] Adith Swaminathan and Thorsten Joachims. 2015. The self-normalized estimator for counterfactual learning. In *Advances in Neural Information Processing Systems*. 3231–3239.

[23] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miro Dudik, John Langford, Damien Jose, and Imed Zitouni. 2017. Off-policy evaluation for slate recommendation. In *Advances in Neural Information Processing Systems*. 3635–3645.

[24] Christoph Trattner and David Elsweiler. 2017. Investigating the healthiness of internet-sourced recipes: implications for meal planning and recommender systems. In *Proceedings of the 26th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 489–498.

[25] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in neural information processing systems*. 2643–2651.

[26] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[27] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, Vol. 11. 2764–2770.

[28] Longqi Yang, Eugene Bagdasaryan, Joshua Gruenstein, Cheng-Kang Hsieh, and Deborah Estrin. 2018. OpenRec: A Modular Framework for Extensible and Adaptable Recommendation Algorithms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 664–672. https://doi.org/10.1145/3159652.3159681

[29] Longqi Yang, Chen Fang, Hailin Jin, Matthew D Hoffman, and Deborah Estrin. 2017. Personalizing Software and Web Services by Integrating Unstructured Application Usage Traces. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 485–493.

[30] Xiaoying Zhang, Junzhou Zhao, and John Lui. 2017. Modeling the Assimilation-Contrast Effects in Online Product Rating Systems: Debiasing and Recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 98–106.