Bounding System-Induced Biases in Recommender Systems with a Randomized Dataset

DUGANG LIU, College of Computer Science and Software Engineering, Shenzhen University, China
PENGXIANG CHENG, Huawei Noah's Ark Lab, China
ZINAN LIN, College of Computer Science and Software Engineering, Shenzhen University, China
XIAOLIAN ZHANG and ZHENHUA DONG, Huawei 2012 Lab, China
RUI ZHANG, Tsinghua University, China
XIUQIANG HE, Tencent FIT, China
WEIKE PAN and ZHONG MING, College of Computer Science and Software Engineering, Shenzhen University, China

Debiased recommendation with a randomized dataset has shown very promising results in mitigating system-induced biases. However, it still lacks more theoretical insights or an ideal optimization objective function compared with the other more well-studied routes without a randomized dataset. To bridge this gap, we study the debiasing problem from a new perspective and propose to directly minimize the upper bound of an ideal objective function, which facilitates a better potential solution to system-induced biases. First, we formulate a new ideal optimization objective function with a randomized dataset. Second, according to the prior constraints that an adopted loss function may satisfy, we derive two different upper bounds of the objective function: a generalization error bound with triangle inequality and a generalization error bound with separability. Third, we show that most existing related methods can be regarded as the insufficient optimization of these two upper bounds. Fourth, we propose a novel method called *debiasing approximate upper bound* (*DUB*) with a randomized dataset, which achieves a more sufficient optimization of these upper bounds. Finally, we conduct extensive experiments on a public dataset and a real product dataset to verify the effectiveness of our DUB.

CCS Concepts: • **Information systems → Recommender systems;**

Additional Key Words and Phrases: System-induced bias, recommender systems, randomized dataset, upper bound minimization

Dugang Liu also with Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ).

We thank the National Natural Science Foundation of China for its support (grant nos. 61836005, 62272315, and 62172283).

Authors' addresses: D. Liu, Z. Lin, W. Pan (corresponding author), and Z. Ming (corresponding author), College of Computer Science and Software Engineering, Shenzhen University, 3688# Nanhai Avenue, Shenzhen, Guangdong, China, 518060; emails: {dugang.ldg, lzn87591}@gmail.com, {panweike, mingz}@szu.edu.cn; P. Cheng, Huawei Noah's Ark Lab, Bantian Street, Shenzhen, Guangdong, China, 518129; email: pengxiang.cpx@gmail.com; X. Zhang and Z. Dong, Huawei 2012 Lab, Bantian Street, Shenzhen, Guangdong, China, 518129; emails: {zhangxiaolian, dongzhenhua}@huawei.com; R. Zhang, Tsinghua University, Guangdong, China, 518055; email: rayteam@yeah.net; X. He, Tencent FIT, 33# Haitian Second Road, Shenzhen, Guangdong, China, 518057; email: xiuqianghe@tencent.com.

1046-8188/2023/04-ART108 $15.00
https://doi.org/10.1145/3582002

ACM Reference format:
Dugang Liu, Pengxiang Cheng, Zinan Lin, Xiaolian Zhang, Zhenhua Dong, Rui Zhang, Xiuqiang He, Weike Pan, and Zhong Ming. 2023. Bounding System-Induced Biases in Recommender Systems with a Randomized Dataset. *ACM Trans. Inf. Syst.* 41, 4, Article 108 (April 2023), 26 pages.
https://doi.org/10.1145/3582002

1 INTRODUCTION

Recently, the bias issue in recommender systems has received more attention from both the research communities and industries [27, 28, 36, 44, 45, 50]. Intuitively, as shown in Figure 1, a user will experience system-induced biases and user-induced biases when interacting with a recommender system. The *system-induced biases* are caused by the stochastic recommendation policy deployed on a recommender system and the selection and display order of each item is treated differently by the policy, including popularity bias [2, 6, 53], selection bias [17, 29, 35] and position bias [4, 39]. The *user-induced biases* depend on the user characteristics, such as trust bias and conformity bias [3, 24, 25, 52]. These specific biases will eventually be coupled into the *data bias* on the user feedback. In this article, we call this type of data a *non-randomized dataset*.

Since different biases may be coupled, mitigating a set of biases from a data perspective is an important research route. In addition, it is easier to reduce the system-induced biases by controlling the recommendation policy than by interacting with the user to reduce the user-induced biases. For these reasons, previous works propose using a special uniform policy to replace the stochastic recommendation policy [5, 7, 20]. Using a uniform policy means that for each user's request, instead of using a recommendation model for item delivery, the system randomly selects some items from all of the candidates and ranks the selected items with a uniform distribution. The users' feedback collected under such a uniform policy is called a *randomized dataset*. A randomized dataset can be regarded as a good unbiased agent because it largely avoids the sources of the system-induced biases. However, because the uniform logging policy does not take into account each user's preferences and tends to show the users a collection of the items that they are not interested in, it will hurt the users' experiences and the revenue of the platform. This means that it is necessary to constrain a randomized dataset collection within a particularly limited network traffic.

To utilize such a scarce and precious randomized dataset to help model training on a non-randomized dataset, the existing methods can be divided into three categories: (1) Use a randomized dataset to re-weight the samples in a non-randomized dataset [35, 47] or to train an imputation model for data augmentation of a non-randomized dataset [20, 48, 49]. In addition, the two can be integrated as a doubly robust framework [7, 40]. (2) Design a multi-stage training framework to alternately use a non-randomized dataset and a randomized dataset to learn debiased parameters [7, 41]. (3) Use a randomized dataset and a non-randomized log dataset to train two models jointly and constrain them to be close in some way so that the model trained on a non-randomized dataset can benefit from the model trained on a randomized dataset [5, 20]. Although these existing works have shown promising results in mitigating system-induced bias, it is still weak in theoretical insights or an ideal optimization objective function compared with the other more well-studied route, that is, debiased recommendation without a randomized dataset [22, 33, 34, 42]. This prevents theoretical analysis of the existing methods and systematic guidance of this research route.

To bridge this gap, we extend previous theoretical insights on debiased recommendation without a randomized dataset [33]. We first formulate a new ideal optimization objective function considering a randomized dataset and propose a new debiased perspective to facilitate the introduction of some theoretical insights and a more sufficient solution to the system-induced biases, that is, the debiasing issue is equivalent to directly optimizing the upper bound of this objective function.

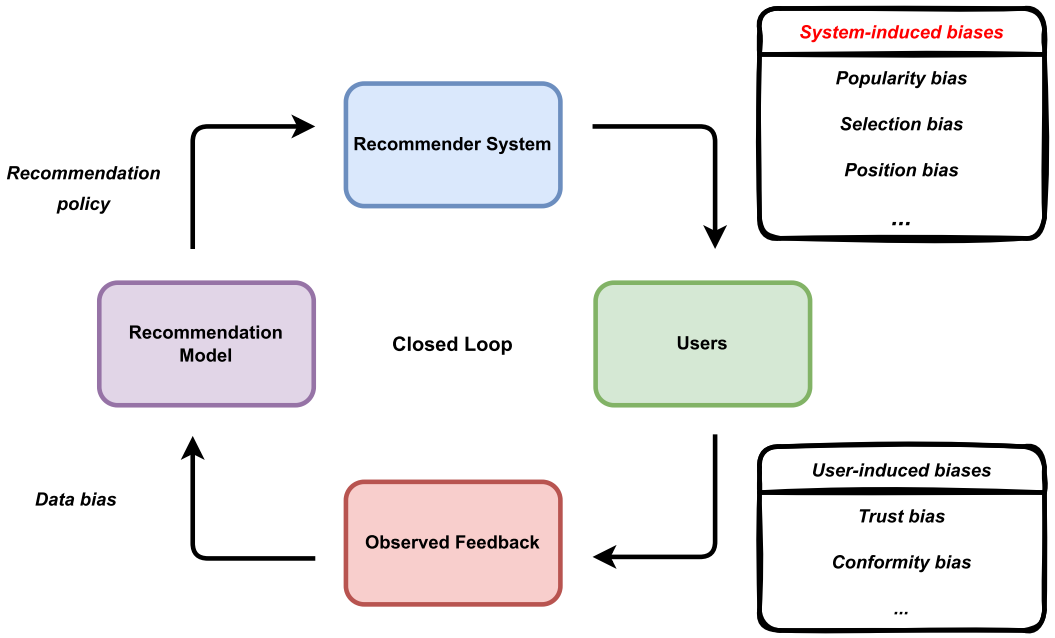


Fig. 1. The feedback loop in a recommender system, where the observed feedback contains the data bias coupled by the system-induced biases and the user-induced biases. The former is caused by the stochastic recommendation policy deployed on a recommender system, and the latter depends on the user characteristics.

Then, we derive two upper bounds of the *unbiased ideal loss function* corresponding to this objective function in practice: one generalization error bound with triangle inequality (in Section 4.1.1) and the other with separability (in Section 4.1.2). The difference between the two depends on the different prior constraints satisfied by the adopted loss function. We show that most existing methods can be regarded as insufficient optimization of our upper bound and propose a novel debiasing method called *debiasing approximate upper bound* (*DUB*). Our method achieves a more sufficient optimization on the upper bound, which is expected to further improve performance. We then conduct extensive experiments on a public dataset and a real product dataset to verify the effectiveness of the proposed method from five different aspects: unbiased testing scenarios, biased general testing scenarios, the ablation experiments, the distribution of the recommendation lists, and some key factors that may affect the performance of the proposed method.

The structure of this article is as follows. We briefly introduce some related works in Section 2. We present some necessary preliminaries in Section 3. We give a detailed description of the proposed theoretical insights and method in Section 4 and discuss the relations to the existing debiasing methods in Section 5. We analyze and discuss extensive experimental results in Section 6. We present a conclusion and some future directions in Section 7. The contributions of this article are as follows.

- We propose a new debiased perspective and formulate a new ideal optimization objective function with a randomized dataset, based on which a novel solution to system-induced biases can be obtained by directly minimizing the upper bound of this ideal optimization objective function.
- We give some theoretical insights on the upper bound of this ideal optimization objective function, in which the adopted loss functions satisfy triangle inequality and separability.

- We show that most existing solutions can be viewed as insufficient optimization of the two proposed upper bounds and then propose a novel method called *debiasing approximate upper bound* (*DUB*) with a randomized dataset for a more sufficient optimization of the proposed upper bound.
- We conduct extensive experiments on a public dataset and a real product dataset to show the effectiveness of the proposed method, including unbiased evaluation, biased general evaluation, ablation experiments of the model and distribution of the recommendation lists, as well as some key factors that may affect the performance of our DUB.

2 RELATED WORK

In this section, we briefly review some related works on two research topics: debiased recommendation without a randomized dataset and debiased recommendation with a randomized dataset.

2.1 Debiased Recommendation Without a Randomized Dataset

Due to the lack of such unbiased guidance information similar to a randomized dataset, the existing works on debiased recommendation without a randomized dataset require making some prior assumptions about the biases or checking and guaranteeing the unbiasedness of the model based on some specific sophisticated techniques. The existing works on this research route can be further subdivided into three classes — heuristic-based methods, inverse propensity score-based methods [35, 48], and theoretical tools-based methods — depending on the different techniques employed. A heuristic-based method links a user's feedback with different specific factors to make some prior assumptions about the generation process of some specific biases. For example, for selection bias in the feedback data (also known as *missing not at random mechanism*), some previous works have assumed that users' feedback on an item is related to their rating of the item and that users will provide their own feedback only when they are particularly satisfied or dissatisfied with an item [26, 46]. In addition to linking with ratings, some subsequent works further consider the different contributions of the user features and the item features in a user's feedback [9, 14, 18]. For conformity bias, some previous works assume that users will refer to public opinion in the process of feedback decision-making, such as hiding or adjusting their own feedback [19, 24, 25, 51]. Based on such prior assumptions, these works usually construct a probabilistic graphical model or a polynomial mixture model containing feature information for a specific bias problem and then solve the model parameters based on a generalized expectation maximization algorithm. An inverse propensity score–based method balances the distribution of the items in the observed feedback data by the propensity score estimated based on some variable factors so that a recommendation model trained on the adjusted non-randomized dataset can avoid the interference of these variable factors as much as possible. For example, one of the variable factors most often considered in the existing works is the relative exposure frequency of each item in the feedback data. With the adjustment of the propensity score based on the relative exposure frequency, the exposure distribution of each item in the feedback data is close to uniform [5, 18]. Moreover, a theoretical tool-based method integrates some theoretical tools from other research fields with debiased recommendation. They usually derive an unbiased ideal loss function that can be directly optimized for a specific bias problem or, in a case in which this unbiased ideal loss function is intractable, further derive a generalization error upper bound for it as a tractable alternative optimization objective. The common theoretical tools in the existing works include information bottleneck [22, 23, 42], positive-unlabeled learning [34], upper bound minimization [33], disentangled representation learning [52], and causal inference techniques [38, 43]. Our DUB adopts a similar upper bound minimization idea to provide some new theoretical insights but is quite different from the previous work [33]. We propose a new ideal optimization objective function for

debiased recommendation with a randomized dataset. The existing works consider only the ideal optimization objective functions defined on a non-randomized dataset. As described in Section 3, this new ideal optimization objective function is more favorable for addressing system-induced biases. It can also be seen as an efficient extension of the existing theoretical insights based on upper bound minimization when a randomized dataset is available. We give more theoretical insights where the prior constraints beyond triangle inequality are employed to be compatible with more choices of loss functions in practice.

2.2 Debiased Recommendation with a Randomized Dataset

The research on this route also introduces a randomized dataset that can act as a proxy for the unbiased information. Most debiasing methods that fall into this route aim to mine the unbiased knowledge from a randomized dataset by formulating more sophisticated and efficient techniques and then use them to guide the training process of a recommendation model on a non-randomized dataset. The existing works on this research route can be further subdivided into three classes — inverse propensity score and imputation labels–based methods, multi-stage training-based methods, and joint training-based methods—depending on the different techniques employed. An inverse propensity score and imputation labels–based method utilizes an additional randomized dataset to estimate the propensity score for each feedback [35, 47] or to make the predictions of the imputation labels for unobserved feedback data [20, 21, 48, 49]. These obtained propensity scores or imputation labels will be integrated into the model's optimization objective, that is, transfer the unbiased knowledge into the model's training process. Propensity score recommendation learning is a representative work in this sub-route. Two methods are proposed for propensity score estimation based on a randomized dataset: a naïve Bayes estimator and a regression model estimator [35]. Note that the propensity scores are used in both debiased recommendation routes and they differ in whether the propensity score is estimated from a non-randomized dataset or a randomized dataset. Some works also consider estimating and using the propensity scores and imputation labels simultaneously to allow the model to benefit more in a doubly robust framework [7, 40]. A multi-stage training-based method designs effective multi-stage training frameworks in which a non-randomized dataset and a randomized dataset are used alternately based on the synergy with which it learns better unbiased parameters. AutoDebias [7] is one of the most representative methods in this research sub-line. Its main idea is to introduce a meta-learning strategy into a doubly robust debiasing framework to achieve better learning of the model. In each iteration of training, the parameters of the main network (i.e., the recommendation model) in the framework are first fixed and a randomized dataset is used to better estimate the propensity scores and imputation labels in the auxiliary meta-learning network. Then, the parameters of the auxiliary meta-learning network are fixed and a non-randomized dataset is used for unbiased model parameter learning in the main network. This multi-stage training mode is repeated until the recommendation model converges to a better feasible solution. Clearly, AutoDebias can be seen as an effective improvement on the training process towards a doubly robust debiasing framework, which is different from most existing debiasing methods that aim to improve the model's optimization objective. A joint training–based method trains a recommendation model and an auxiliary model for a non-randomized dataset and a randomized dataset, respectively, and uses custom alignment terms to directly constrain the two models for joint training. CausE [5] is a pioneering work of this sub-route that introduces an alignment term of model parameters to facilitate information fusion between the two models. Since the parameter alignment term will increase the difficulty of model training in a practical application, instead of aligning the two models on the model parameters, Bridge [20] constrains the predicted labels of the two models to be as close as possible on an auxiliary set sampled from the full set of feedback. In contrast to the existing works, we propose a

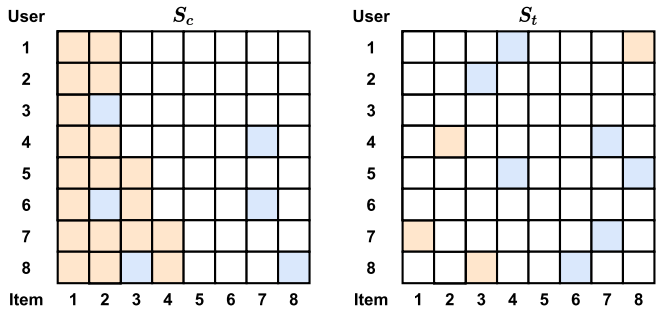


Fig. 2. An example of the difference between a non-randomized dataset S_c and a randomized dataset S_t.

new perspective on addressing system-induced biases from the upper bound of an unbiased ideal loss function and provide a theoretical objective function with a randomized dataset that can be optimized directly. This means that we convert the task of reducing the system-induced biases to an optimization problem that can be solved directly, which provides more guidance on the use of a randomized dataset and the analysis of debiasing methods.

3 PRELIMINARIES

3.1 Notations

A typical recommender system usually takes a user $u_i \in U$ as input and selects an attractive item $v_j \in V$ to be displayed to this user through a stochastic recommendation policy π_c deployed by the system, that is, $v_j \sim \pi_c(\cdot|u_i)$. Then, the system will collect the user's feedback on each displayed item $r^c_{ij} \sim R^c(\cdot|u_i, v_j) \in \{0, 1\}$, where $r^c_{ij} = 1$ denotes positive feedback, $r^c_{ij} = 0$ denotes negative feedback, and R^c is a complete feedback matrix under π_c. In this article, we call this type of data a *non-randomized dataset* S_c. Based on the collected data S_c, the system will retrain a recommendation model M_c and update the recommendation policy. Similarly, under a uniform policy π_t, we have that $v_j \sim \pi_t(\cdot|u_i)$ and $r^t_{ij} \sim R^t(\cdot|u_i, v_j)$. R^t is a complete feedback matrix under π_t, the feedback of users recorded under π_t is called *randomized dataset* S_t, and M_t is the auxiliary model trained on S_t.

To facilitate understanding of the difference between a non-randomized dataset S_c and a randomized dataset S_t, we include an example in Figure 2, in which the recommender system is assumed to contain 8 users and 8 items, and a yellow square and a blue square indicate that the corresponding user-item pair (u_i, v_j) is positive feedback and negative feedback, respectively. Due to the restricted collection process, the scale and scope of S_t are often much smaller than that of S_c, where scale refers to the amount of data and scope refers to the coverage of users and items. We can see from Figure 2 that in a randomized dataset S_t, the number of colored squares is smaller and there are some users who do not have colored squares. Due to the nature of a uniform policy π_t, a randomized dataset S_t suffers from less bias than a non-randomized dataset S_c, especially system-induced biases. From Figure 2, we can see that this relative unbiasedness may be reflected in the fact that each item has a similar probability of getting feedback from different users (i.e., each item has a similar number of colored squares) and each user has a preference distribution that is closer to the ideal state (i.e., due to limited preferences, a user should have far more negative feedback than positive feedback on all items [30]). We can also see from Figure 2 that a randomized dataset S_t may reveal interests for a user that are not perceived in a recommendation policy π_c, such as user 1 for item 8, and may correct for pseudo-negative feedback in a non-randomized dataset S_c subject to the system-induced biases, such as user 8 for item 3. Note that in order to ensure

non-overlapping between S_c and S_t, and because the feedback data in S_t is more unbiased and credible, we actually remove from S_c those feedback data that appear in S_t, such as user 8 for item 3.

Since a non-randomized dataset S_c and a randomized dataset S_t are part of the complete feedback matrix (i.e., R^c and R^t) under a recommendation policy π_c and a uniform policy π_t, respectively, we can intuitively conclude that R^c and R^t inherit this difference in bias between S_c and S_t, that is, R^t has better unbiasedness than R^c. In particular, each element in R^t can be thought of as a user's feedback result after an item has been displayed in all possible ways. This is a result that can be gradually achieved through the long-term deployment of a uniform policy π_t. Unlike R^t, even if we can obtain R^c, it can alleviate only some of the biases induced by the system and still inevitably suffers from the rest of these biases, especially pseudo-negative feedback.

3.2 Problem Formulation

The optimization objective of most existing recommendation methods is the average loss function over the observed feedback under a policy π_c,

$$\mathcal{L}^{\ell}_{observed}(R^c, \hat{R}^c) = \frac{1}{|O|} \sum_{(i,j)\in O} \ell\left(R^c_{i,j}, \hat{R}^c_{i,j}\right), \tag{1}$$

where $O \in \{(i,j)\}$ denotes a set of observed feedback, $\hat{R}^c$ denotes the predicted label matrix of M_c, and $\ell(\cdot,\cdot)$ is an arbitrary loss function. Equation (1) can be regarded as the simplest estimator of the ideal optimization objective under policy π_c,

$$\mathcal{L}^{\ell}_{\pi_c-ideal}(R^c, \hat{R}^c) = \frac{1}{|\mathcal{D}|} \sum_{(i,j)\in \mathcal{D}} \ell\left(R^c_{i,j}, \hat{R}^c_{i,j}\right), \tag{2}$$

where $\mathcal{D}$ denotes the complete set of feedback. Due to system-induced biases, Equation (1) is not an unbiased estimation of Equation (2) [26, 37]. Instead, some previous works on debiased recommendation without a randomized dataset have shown that better performance can be obtained by optimizing an unbiased estimation or a generalization error bound of Equation (2) [33, 35].

However, as described in Section 3.1, even if we can obtain the complete feedback matrix R^c under a recommendation policy π_c, R^c can alleviate only some of the biases induced by the system. This means that an unbiased estimator for R^c is not necessarily equivalent to an ideal unbiased evaluation. To further eliminate system-induced biases, based on the analysis in Section 3.1, we argue that a better option is to use R^t instead of R^c. This is because R^t, consisting of a randomized data S_t, obviously contains better relative unbiasedness than R^c. Based on this idea, we formulate a new *ideal optimization objective function*,

$$\mathcal{L}^{\ell}_{\pi_t-ideal}(R^t, \hat{R}^c) = \frac{1}{|\mathcal{D}|} \sum_{(i,j)\in \mathcal{D}} \ell\left(R^t_{i,j}, \hat{R}^c_{i,j}\right). \tag{3}$$

This means that we can optimize Equation (3) as a better solution to the system-induced bias problem. Equation (3) can also be seen as an efficient extension of the existing ideal optimization objective functions when a randomized dataset is available. However, it is very difficult to directly optimize Equation (3). On one hand, we have only a small part of the real feedback of R^t, that is, S_t. On the other hand, although we have a non-randomized dataset S_c, we do not know the corresponding feedback in R^t for these feedback data, that is, the state of a non-randomized dataset S_c in R^t is unknown. In particular, we need to answer the following question: If the items in S_c are randomly displayed, what will the feedback be like? This involves the concept of counterfactual, which is recognized as a challenging problem [32]. To address this challenge, we will turn to deriving an upper bound of Equation (3) and propose a general debiasing framework based on upper

Table 1. The Main Notations and Explanations

Symbol	Meaning
S_t	a randomized dataset
S_c	a non-randomized dataset
S_u	the unobserved data
$\mathcal{D}$	the whole set of data, i.e., $\mathcal{D} = S_c \cup S_t \cup S_u$
M_c	the recommendation model trained on a non-randomized dataset S_c
M_t	the auxiliary model trained on a randomized dataset S_t
R^*	the complete feedback matrix under π_*, $\pi_* \in \{\pi_c, \pi_t\}$
$\hat{R}^*$	the predicted label matrix of M_*, $M_* \in \{M_c, M_t\}$
I_{S_*}	the set of user–item pair indices contained in the feedback data S_*, where $S_* \in \{S_c, S_t, S_u\}$
$\mathcal{L}(R^t, \hat{R}^c)$	the unbiased ideal loss function when a randomized dataset is available
$\mathcal{L}^{S_*}(\cdot,\cdot)$	the loss function defined on the set of user–item pair indices contained in the feedback data S_* with the size of the whole set as the denominator, i.e., $\mathcal{L}^{S_*}(\cdot,\cdot) = \frac{1}{\lvert\mathcal{D}\rvert}\sum_{(u,v)\in I_{S_*}} \ell(\cdot,\cdot)$
$\mathcal{L}^{S_*}_{\lvert S_*\rvert}(\cdot,\cdot)$	the average loss function defined on the set of user–item pair indices contained in the feedback data S_*, i.e., $\mathcal{L}^{S_*}_{\lvert S_*\rvert}(\cdot,\cdot) = \frac{1}{\lvert S_*\rvert}\sum_{(u,v)\in I_{S_*}} \ell(\cdot,\cdot)$

bound minimization in which the upper bound of Equation (3) will be taken as a new optimization objective function to drive a tractable solution.

4 THE PROPOSED METHOD

In this section, we first present some theoretical insights into debiased recommendation with a randomized dataset. Our goal is to derive an upper bound of the *ideal optimization objective function* in Equation (3) by extending the theory in [33] and use it as an alternative objective that can be optimized directly. Note that, in practice, we need to specify the type of loss function ℓ in this optimization objective, and we refer to the objective function having a specific form as the *unbiased ideal loss function* in the following. Different types of loss functions satisfy different prior constraints and have different effects on theoretical insights. Therefore, in order to be compatible with as many types of loss function as possible, we propose two corresponding upper bounds when the adopted loss functions ℓ satisfy triangular inequality (in Section 4.1.1) and separability (in Section 4.1.2), respectively. Then, we discuss the generalization error bounds to clarify the key factors. Finally, we give a detailed description of the proposed method, DUB. Note that unless otherwise specified, we abbreviate $\mathcal{L}^{\ell}_{\pi_t-ideal}(R^t, \hat{R}^c)$ as $\mathcal{L}(R^t, \hat{R}^c)$ in the following for brevity. For ease of reference, the main notations in theoretical analysis are listed in Table 1.

In order to emphasize a confusing notation $\mathcal{L}^{S_*}(\cdot,\cdot)$, we further describe the difference between $\mathcal{L}^{S_c}(R^c, \hat{R}^c)$, $\mathcal{L}^{S_c}(R^c, \hat{R}^t)$, and $\mathcal{L}^{S_c}(R^t, \hat{R}^t)$ as an example. By definition, $\mathcal{L}^{S_c}(R^c, \hat{R}^c)$ denotes a loss function defined on the set of user–item pair indices contained in the feedback data S_c. Therefore, the true labels used in this loss function are the corresponding part of R^c on the specific user–item pair index set I_{S_c}. Obviously, the true labels at this time are the feedback labels of a non-randomized dataset S_c. Similarly, the predicted labels used in the loss function are the predicted outputs of the recommendation model M_c for each sample in a non-randomized dataset S_c. For $\mathcal{L}^{S_c}(R^c, \hat{R}^t)$, the

true labels used are also the feedback labels of a non-randomized dataset S_c, but the predicted labels used are changed to the part of $\hat{R}^t$ on the specific user–item pair index set I_{S_c}, that is, the predicted outputs of the auxiliary model M_t for each sample in a non-randomized dataset S_c. In particular, for $\mathcal{L}^{S_c}(R^t, \hat{R}^t)$, the true labels used in the loss function are changed to the part of R^t on the specific user-item pair index set I_{S_c}. Obviously, as described in Section 3.1, we cannot know the true labels of this part of the feedback data in practice, that is, it cannot be optimized directly using the supervision information.

4.1 Theoretical Analysis

4.1.1 A Generalization Error Bound with Triangle Inequality. Similar to most works using the upper bound minimization framework [8, 33], we first consider the case in which the adopted loss function ℓ satisfies the triangle inequality, for example, the 0-1 loss and l_1-norm [13, 16]. In Proposition 4.1, we first derive a simple upper bound on Equation (3) based on this prior constraint.

Proposition 4.1. *Assume that the loss function ℓ obeys the triangle inequality. Then, for any given predicted label matrices $\hat{R}^t$ and $\hat{R}^c$, the following inequality holds.*

$$\mathcal{L}\left(R^t, \hat{R}^c\right) \leq \mathcal{L}^{S_t}\left(R^t, \hat{R}^c\right) + \mathcal{L}^{S_c}\left(R^t, R^c\right) + \mathcal{L}^{S_c}\left(R^c, \hat{R}^c\right) + \mathcal{L}^{S_u}\left(R^t, \hat{R}^t\right) + \mathcal{L}^{S_u}\left(\hat{R}^t, \hat{R}^c\right).$$

Proof.

$$\begin{aligned}\mathcal{L}\left(R^t, \hat{R}^c\right) &= \mathcal{L}^{S_t}\left(R^t, \hat{R}^c\right) + \mathcal{L}^{S_c}\left(R^t, \hat{R}^c\right) + \mathcal{L}^{S_u}\left(R^t, \hat{R}^c\right) \\ &\leq \mathcal{L}^{S_t}\left(R^t, \hat{R}^c\right) + \mathcal{L}^{S_c}\left(R^t, R^c\right) + \mathcal{L}^{S_c}\left(R^c, \hat{R}^c\right) + \mathcal{L}^{S_u}\left(R^t, \hat{R}^t\right) + \mathcal{L}^{S_u}\left(\hat{R}^t, \hat{R}^c\right),\end{aligned}$$

where S_u denotes the set of unobserved feedback, that is, $\mathcal{D} = S_c \cup S_t \cup S_u$, $\mathcal{L}^{S_*}(\cdot,\cdot) = \frac{1}{|\mathcal{D}|}\sum_{(u,v)\in I_{S_*}} \ell(\cdot,\cdot)$, and $S_* \in \{S_c, S_t, S_u\}$. We first divide Equation (3) into a summation of three disjoint subsets, and apply the triangle inequality to $\mathcal{L}^{S_c}(R^t, \hat{R}^c)$ and $\mathcal{L}^{S_u}(R^t, \hat{R}^c)$. Note that, as described in Section 3.1, the disjoint properties of S_c and S_t are ensured during the data collection phase. □

The fourth term in Proposition 4.1 is difficult to solve because we know the true labels of only a small part of R_t, that is, S_t, but not the true labels of R_t on the specific user–item pair index set I_{S_u}. Therefore, through Hoeffding's inequality [12], we convert it into an easy-to-solve alternative and further analyze the generalization error bound of the unbiased ideal loss function.

Theorem 4.2 (Generalization Error Bound of Unbiased Ideal Loss I). *Assume that two predicted matrices $\hat{R}^t$ and $\hat{R}^c$ are given, and a loss function ℓ obeys the triangle inequality and is bounded by a positive constant Δ. Then, for any finite hypothesis space of predictions $\mathcal{H} = \{\hat{R}^t_1, \ldots, \hat{R}^t_{|\mathcal{H}|}\}$, and for any $\eta \in (0,1)$, the ideal loss $\mathcal{L}(R^t, \hat{R}^c)$ is bounded with probability $1-\eta$ by:*

$$\begin{aligned}\mathcal{L}\left(R^t, \hat{R}^c\right) \leq\ & \underbrace{\mathcal{L}^{S_t}\left(R^t, \hat{R}^c\right)}_{(a)} + \underbrace{\mathcal{L}^{S_c}\left(R^t, R^c\right)}_{(b)} + \underbrace{\mathcal{L}^{S_c}\left(R^c, \hat{R}^c\right)}_{(c)} \\ & + \underbrace{\mathcal{L}^{S_u}\left(\hat{R}^t, \hat{R}^c\right)}_{(d)} + \underbrace{\mathcal{L}^{S_t}_{|S_t|}\left(R^t, \hat{R}^t\right)}_{(e)} + bias\left(\mathcal{L}^{S_t}_{|S_t|}\left(R^t, \hat{R}^t\right)\right) \\ & + \frac{\Delta}{|S_t|}\sqrt{\frac{|\mathcal{D}|}{2}\log\left(\frac{2|\mathcal{H}|}{\eta}\right)},\end{aligned} \tag{4}$$

where $bias(\mathcal{L}^{S_t}_{|S_t|}(R^t, \hat{R}^t)) = \mathcal{L}^{S_u}(R^t, \hat{R}^t) - \mathbb{E}[\mathcal{L}^{S_t}_{|S_t|}(R^t, \hat{R}^t)]$ is the error term caused by using $\mathbb{E}[\mathcal{L}^{S_t}_{|S_t|}(R^t, \hat{R}^t)]$ to replace $\mathcal{L}^{S_u}(R^t, \hat{R}^t)$, and $\mathcal{L}^{S_t}_{|S_t|}(R^t, \hat{R}^t) = \frac{1}{|S_t|}\sum_{(i,j)\in S_t} \ell(R^t_{i,j}, \hat{R}^t_{i,j})$.

Proof. Our goal is to use the easy-to-solve term (e) in Equation (4) to replace the fourth difficult-to-solve term in Proposition 4.1, and obtain the approximate error term corresponding to this operation.

First, we have the following equation:

$$\begin{aligned}\mathcal{L}^{S_u}\left(R^t,\hat{R}^t\right) &= \mathcal{L}^{S_u}\left(R^t,\hat{R}^t\right) - \mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right] + \mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right] \\ &= \mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right] + bias\left(\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right). \end{aligned} \tag{5}$$

Using Hoeffding's inequality and union bounds to make a uniform convergence argument, we get that

$$\begin{aligned} & P\left(\left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right] - \mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right| \le \epsilon\right) \ge 1-\eta \\ \Leftarrow\ & P\left(\max_{\hat{R}^t_h\in\mathcal{H}}\left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t_h\right)\right] - \mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t_h\right)\right| \le \epsilon\right) \ge 1-\eta \\ \Leftrightarrow\ & P\left(\bigcup_{\hat{R}^t_h\in\mathcal{H}}\left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t_h\right)\right] - \mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t_h\right)\right| \ge \epsilon\right) \le \eta \\ \Leftarrow\ & \sum_{h=1}^{|\mathcal{H}|} P\left(\left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t_h\right)\right] - \mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t_h\right)\right| \ge \epsilon\right) \le \eta \\ \Leftarrow\ & |\mathcal{H}| \times 2\exp\left(\frac{-2\,|S_t|^2\,\epsilon^2}{|\mathcal{D}|\,\Delta^2}\right) \le \eta. \end{aligned}$$

Solving for ϵ yields the bound

$$\begin{aligned} & \left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right] - \mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right| \le \frac{\Delta}{|S_t|}\sqrt{\frac{|\mathcal{D}|}{2}\log\left(\frac{2\,|\mathcal{H}|}{\eta}\right)} \\ \Rightarrow\ & \mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right] \le \mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right) + \frac{\Delta}{|S_t|}\sqrt{\frac{|\mathcal{D}|}{2}\log\left(\frac{2\,|\mathcal{H}|}{\eta}\right)}. \end{aligned} \tag{6}$$

By combining Equations (5) and (6), we get the following inequality, which holds with a probability of at least $1-\eta$:

$$\mathcal{L}^{S_u}\left(R^t,\hat{R}^t\right) \le \mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right) + \frac{\Delta}{|S_t|}\sqrt{\frac{|\mathcal{D}|}{2}\log\left(\frac{2\,|\mathcal{H}|}{\eta}\right)} + bias\left(\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)\right). \tag{7}$$

Then, by combining Proposition 4.1 and Equation (7), the proof is completed. □

4.1.2 A Generalization Error Bound with Separability. Note that in recommender systems, some widely used loss functions do not satisfy the triangular inequality, for example, the cross-entropy loss and the mean square error. To further expand the optional range of the loss function, we propose a new prior constraint on the loss function,

Definition 4.3 (Separability). A loss is considered to satisfy the separability if and only if the following inequality holds:

$$\mathcal{L}^{\ell}(c,a) \le \mathcal{L}^{\ell}(b,a) + \mathcal{L}^{\ell}(c-b,a).$$

Proof. As an example, we prove that the binary cross-entropy loss satisfies the separability, and other loss functions can be checked in a similar process. Given a form of the binary cross-entropy loss $\mathcal{L}^{\ell}(y,\hat{y}) = -[y\log\hat{y} + (1-y)\log(1-\hat{y})]$, where $y \in \{0,1\}$, we can derive that

$$\begin{aligned}\mathcal{L}^{\ell}(c,a) - \mathcal{L}^{\ell}(b,a) &= -[c\log a + (1-c)\log(1-a)] + [b\log a + (1-b)\log(1-a)]\\ &= -[(c-b)\log a - (c-b)\log(1-a)]\\ &\le -[(c-b)\log a - (c-b)\log(1-a)] - \log(1-a)\\ &= -[(c-b)\log a + (1-(c-b))\log(1-a)]\\ &= \mathcal{L}^{\ell}(c-b,a).\end{aligned}$$

The inequality conversion in the process can be obtained because of the non-negativity of $-\log(1-a)$, where $0 \le a \le 1$. Then, the binary cross-entropy loss satisfies the separability. □

Based on the separability, similar to the proof process of Proposition 4.1 and Theory 4.2, we can get Proposition 4.4 and Theory 4.5.

Proposition 4.4. *Assume that the loss function ℓ obeys the separability. Then, for any given predicted label matrices $\hat{R}_t$ and $\hat{R}_c$, the following inequality holds.*

$$\begin{aligned}\mathcal{L}\left(R^t,\hat{R}^c\right) \le\ & \mathcal{L}^{S_t}\left(R^t,\hat{R}^c\right) + \mathcal{L}^{S_c}\left(R^t - R^c,\hat{R}^c\right) + \mathcal{L}^{S_c}\left(R^c,\hat{R}^c\right)\\ &+ \mathcal{L}^{S_u}\left(R^t - \hat{R}^t,\hat{R}^c\right) + \mathcal{L}^{S_u}\left(\hat{R}^t,\hat{R}^c\right).\end{aligned}$$

Proof.

$$\begin{aligned}\mathcal{L}\left(R^t,\hat{R}^c\right) &= \mathcal{L}^{S_t}\left(R^t,\hat{R}^c\right) + \mathcal{L}^{S_c}\left(R^t,\hat{R}^c\right) + \mathcal{L}^{S_u}\left(R^t,\hat{R}^c\right)\\ &\le \mathcal{L}^{S_t}\left(R^t,\hat{R}^c\right) + \mathcal{L}^{S_c}\left(R^t - R^c,\hat{R}^c\right) + \mathcal{L}^{S_c}\left(R^c,\hat{R}^c\right)\\ &+ \mathcal{L}^{S_u}\left(R^t - \hat{R}^t,\hat{R}^c\right) + \mathcal{L}^{S_u}\left(\hat{R}^t,\hat{R}^c\right),\end{aligned}$$

where we apply the separability to $\mathcal{L}^{S_c}(R^t,\hat{R}^c)$ and $\mathcal{L}^{S_u}(R^t,\hat{R}^c)$. □

Theorem 4.5 (Generalization Error Bound of Unbiased Ideal Loss II). *Assume that two predicted matrices $\hat{R}^t$ and $\hat{R}^c$ are given, and a loss function ℓ obeys the separability and is bounded by a positive constant Δ. Then, for any finite hypothesis space of predictions, $\mathcal{H} = \{\hat{R}^c_1,\ldots,\hat{R}^c_{|\mathcal{H}|}\}$, and for any $\eta \in (0,1)$, the ideal loss $\mathcal{L}(R^t,\hat{R}^c)$ is bounded with probability $1-\eta$ by*

$$\begin{aligned}\mathcal{L}\left(R^t,\hat{R}^c\right) \le\ & \underbrace{\mathcal{L}^{S_t}\left(R^t,\hat{R}^c\right)}_{(a)} + \underbrace{\mathcal{L}^{S_c}\left(R^t - R^c,\hat{R}^c\right)}_{(b)} + \underbrace{\mathcal{L}^{S_c}\left(R^c,\hat{R}^c\right)}_{(c)}\\ &+ \underbrace{\mathcal{L}^{S_u}\left(\hat{R}^t,\hat{R}^c\right)}_{(d)} + \underbrace{\mathcal{L}^{S_t}_{|S_t|}\left(R^t - \hat{R}^t,\hat{R}^c\right)}_{(e)}\\ &+ \frac{\Delta}{|S_t|}\sqrt{\frac{|\mathcal{D}|}{2}\log\left(\frac{2\,|\mathcal{H}|}{\eta}\right)} + bias\left(\mathcal{L}^{S_t}_{|S_t|}\left(R^t - \hat{R}^t,\hat{R}^c\right)\right).\end{aligned} \tag{8}$$

Proof. Our goal is to use the easy-to-solve term (e) in Equation (8) to replace the fourth difficult-to-solve term in Proposition 4.4 and obtain the approximate error term corresponding to this operation.

First, we have the following equation:

$$\begin{aligned}\mathcal{L}^{S_u}\left(R^t-\hat{R}^t,\hat{R}^c\right) &= \mathcal{L}^{S_u}\left(R^t-\hat{R}^t,\hat{R}^c\right)-\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right]+\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right]\\ &= \mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right]+bias\left(\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right). \end{aligned} \tag{9}$$

Using Hoeffding's inequality and union bounds to make a uniform convergence argument, we get that

$$\begin{aligned} & P\left(\left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right]-\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right|\le\epsilon\right)\ge 1-\eta\\ \Leftarrow\ & P\left(\max_{\hat{R}^c_h\in\mathcal{H}}\left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c_h\right)\right]-\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c_h\right)\right|\le\epsilon\right)\ge 1-\eta\\ \Leftrightarrow\ & P\left(\bigcup_{\hat{R}^c_h\in\mathcal{H}}\left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c_h\right)\right]-\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c_h\right)\right|\ge\epsilon\right)\le\eta\\ \Leftarrow\ & \sum_{h=1}^{|\mathcal{H}|}P\left(\left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c_h\right)\right]-\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c_h\right)\right|\ge\epsilon\right)\le\eta\\ \Leftarrow\ & |\mathcal{H}|\times 2\exp\left(\frac{-2\,|S_t|^2\,\epsilon^2}{|\mathcal{D}|\,\Delta^2}\right)\le\eta. \end{aligned}$$

Solving for ϵ yields the bound

$$\begin{aligned} & \left|\mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right]-\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right|\le\frac{\Delta}{|S_t|}\sqrt{\frac{|\mathcal{D}|}{2}\log\left(\frac{2\,|\mathcal{H}|}{\eta}\right)}\\ \Rightarrow\ & \mathbb{E}\left[\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right]\le\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)+\frac{\Delta}{|S_t|}\sqrt{\frac{|\mathcal{D}|}{2}\log\left(\frac{2\,|\mathcal{H}|}{\eta}\right)}. \end{aligned} \tag{10}$$

By combining Equations (9) and (10), we get the following inequality, which holds with a probability of at least $1-\eta$:

$$\mathcal{L}^{S_u}\left(R^t-\hat{R}^t,\hat{R}^c\right)\le\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)+\frac{\Delta}{|S_t|}\sqrt{\frac{|\mathcal{D}|}{2}\log\left(\frac{2\,|\mathcal{H}|}{\eta}\right)}+bias\left(\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)\right). \tag{11}$$

Then, by combining Proposition 4.4 and Equation (11), the proof is completed. □

4.2 Analysis of the Generalization Error Bounds

As suggested in Theory 4.2 and Theory 4.5, we list the corresponding explanation for each term in the generalization error bounds. For different terms in the two generalization error bounds, we use indexes 1 and 2 to denote the upper bound of the triangle inequality and the upper bound of the separability, respectively. The two generalization error bounds are the same in terms (a), (c), and (d), but are different in terms (b) and (e).

(a) By definition, $\mathcal{L}^{S_t}(R^t,\hat{R}^c)=\frac{1}{|\mathcal{D}|}\sum_{(u,v)\in I_{S_t}}\ell(R^t,\hat{R}^c)$, that is, the predicted loss of M_c with the size of the whole set as the denominator with regard to the true feedback labels on S_t.

(b.1) By definition, $\mathcal{L}^{S_c}(R^t,R^c)=\frac{1}{|\mathcal{D}|}\sum_{(u,v)\in I_{S_c}}\ell(R^t,R^c)$, that is, the difference between the true feedback labels of policy π_c and policy π_t on the specific user–item pair index set I_{S_c}.

(b.2) By definition, $\mathcal{L}^{S_c}(R^t - R^c, \hat{R}^c) = \frac{1}{|\mathcal{D}|}\sum_{(u,v)\in I_{S_c}} \ell(R^t - R^c, \hat{R}^c)$, that is, the predicted loss of M_c with regard to the difference between the true feedback labels of policy π_c and policy π_t on the specific user–item pair index set I_{S_c}.

(c) By definition, $\mathcal{L}^{S_c}(R^c, \hat{R}^c) = \frac{1}{|\mathcal{D}|}\sum_{(u,v)\in I_{S_c}} \ell(R^c, \hat{R}^c)$, that is, the supervised loss of M_c with the size of the whole set as the denominator with regard to the true feedback labels on S_c.

(d) By definition, $\mathcal{L}^{S_u}(\hat{R}^t, \hat{R}^c) = \frac{1}{|\mathcal{D}|}\sum_{(u,v)\in I_{S_u}} \ell(\hat{R}^t, \hat{R}^c)$, that is, the unsupervised loss between M_t and M_c on the specific user-item pair index set I_{S_u}.

(e.1) By definition, $\mathcal{L}^{S_t}_{|S_t|}(R^t, \hat{R}^t) = \frac{1}{|S_t|}\sum_{(u,v)\in I_{S_t}} \ell(R^t, \hat{R}^t)$, that is, the supervised loss of M_t with regard to the true feedback labels on S_t.

(e.2) By definition, $\mathcal{L}^{S_t}_{|S_t|}(R^t - \hat{R}^t, \hat{R}^c) = \frac{1}{|S_t|}\sum_{(u,v)\in I_{S_t}} \ell(R^t - \hat{R}^t, \hat{R}^c)$, that is, the predicted loss of M_c with regard to the prediction error of M_t on the specific user–item pair index set I_{S_t}.

Intuitively, the three common terms (a), (c), and (d) can be viewed as the supervised loss of M_c on S_c and S_t, and the unsupervised alignment loss between M_c and M_t on S_u, respectively. Since they all have the corresponding supervision information, all three terms can be optimized directly. Under the triangle inequality, term $(b.1)$ can be seen as the difference between both S_c and S'_c when S_c's corresponding feedback S'_c in R^t is known. Therefore, term $(b.1)$ is a constant that can be used to estimate the degree of difference between the two policies and is usually small since the system-induced biases do not have an excessive effect on the user's true preference. The term $(e.1)$ is the supervised loss of M_t itself on S_t and, thus, can also be optimized directly. Under the separability, term $(b.2)$ and term (c) jointly adjust M_c's trade-off in the supervised loss on S_c. Since we do not have the true feedback labels of R_t on the specific user–item pair index set I_{S_c}, we cannot optimize the term $(b.2)$ directly. Fortunately, our experiments show that our method still has a significant advantage even in its absence, and we leave its further processing as future work. Similarly, term $(e.2)$ and term (a) jointly adjust M_c's trade-off in the supervised loss on S_t. Since the prediction error of M_t on the specific user–item pair index set I_{S_t} is available, term $(e.2)$ can also be optimized directly. In short, no matter which generalization error bound is satisfied by the adopted loss function, we can improve the unbiased performance of the recommendation model by simultaneously minimizing the terms (a), (c), (d), and (e) in the generalization error bound. Note that the last two terms in the generalization error bound as shown in Equation (4) are the error terms that arise when we use the easy-to-solve term (e) in Equation (4) to approximate the fourth difficult-to-solve term in Proposition 4.1. Their values depend on the confidence of this approximation process and are independent of the model. In particular, we can find that as the size of a randomized dataset gradually increases, the values of these error terms gradually decrease, which means that the approximation operation is more reliable. It is expected that when a randomized dataset is large, the training of the model can benefit more from more reliable unbiased information. The last two terms of another generalization error bound shown in Equation (8) have similar properties.

4.3 Debiasing Approximate Upper Bound with a Randomized Dataset

Based on the analysis for each term of the generalization error bound in Section 4.2, we propose a novel method called *debiasing approximate upper bound* (*DUB*) with a randomized dataset, which aims to optimize the upper bound of the unbiased ideal loss function directly. Note that we use the term "approximate upper bound" to distinguish it from the term "upper bound" since our DUB considers the terms in Equation (4) (or Equation (8)) that can be optimized directly but not all of the terms. Depending on the types of loss functions used, we have two types of objective functions to be optimized. When the used loss function satisfies the triangular inequality, the optimization goal is shown in Equation (12), which is to minimize a proxy of the upper bound shown in Equation (4).

$$\min_{\mathcal{W}_c,\mathcal{W}_t} \underbrace{\mathcal{L}^{S_t}\left(R^t,\hat{R}^c\right)}_{(a)} + \underbrace{\mathcal{L}^{S_c}\left(R^c,\hat{R}^c\right)}_{(c)} + \underbrace{\mathcal{L}^{S_t}_{|S_t|}\left(R^t,\hat{R}^t\right)}_{(e.1)} + \gamma \underbrace{\mathcal{L}^{S_u}\left(\hat{R}^t,\hat{R}^c\right)}_{(d)} + \lambda_c \mathrm{Reg}\left(\mathcal{W}_c\right) + \lambda_t \mathrm{Reg}\left(\mathcal{W}_t\right), \tag{12}$$

where γ is the weight parameter of $\mathcal{L}^{S_u}(\hat{R}^t,\hat{R}^c)$, and $\mathcal{W}_c$ and $\mathcal{W}_t$ denote the parameters of M_c and M_t, respectively. Note that $\mathrm{Reg}(\cdot)$ is the regularization term, and λ_c and λ_t are the parameters of the regularization. Recall from the analysis in Section 4.2 that all of the terms that can be optimized directly in the generalization error bound as shown in Equation (4) include the terms (a), (c), (d), and $(e.1)$. This corresponds to each optimization term in Equation (12). Note that since the unsupervised loss of M_t and M_c on S_u may contain too much noise when the size of a randomized dataset S_t is small, we introduce a weight parameter γ to control its influence. For the stability of model training, we also include two regularization terms for the model parameters. An intuitive explanation of Equation (12) is to use a non-randomized dataset S_c and a randomized dataset S_t for the trade-off learning of M_c, and to further provide the unbiased information for M_c through the imputation labels provided by M_t. Therefore, our DUB can be viewed as a combination of sample-based debiasing distillation and label-based debiasing distillation defined in [20].

When the used loss function satisfies the separability, the optimization problem is shown in Equation (13), which is to minimize a proxy of the upper bound shown in Equation (8).

$$\min_{\mathcal{W}_c} \underbrace{\mathcal{L}^{S_t}\left(R^t,\hat{R}^c\right)}_{(a)} + \underbrace{\mathcal{L}^{S_c}\left(R^c,\hat{R}^c\right)}_{(c)} + \underbrace{\mathcal{L}^{S_t}_{|S_t|}\left(R^t-\hat{R}^t,\hat{R}^c\right)}_{(e.2)} + \gamma \underbrace{\mathcal{L}^{S_u}\left(\hat{R}^t,\hat{R}^c\right)}_{(d)} + \lambda_c \mathrm{Reg}\left(\mathcal{W}_c\right). \tag{13}$$

Similarly, based on the analysis in Section 4.2, all of the terms that can be optimized directly in the generalization error bound shown in Equation (8) include the terms (a), (c), (d), and $(e.2)$. This corresponds to each optimization term in Equation (13). For the same reason, we also introduce a weight parameter γ and a regularization term for the model parameters. Note that no supervised loss related to M_t is included in Equation (13). Thus, we introduce a regularization term only for M_c. An intuitive explanation of Equation (13) is similar to Equation (12) except that Equation (13) also includes an optimization term (i.e., term $(e.2)$) to enhance M_c's learning of S_t. This can make the model more robust when the relative unbiasedness of a randomized dataset is not high, such as being affected by business rules. Regardless of whether Equation (12) or Equation (13) is used, the proposed method includes all of the terms that can be optimized directly as analyzed in Section 4.2. Our method is a more sufficient optimization of the upper bounds, which is expected to further improve performance.

However, in real applications, we observe an implied limitation of our method due to the large difference in the number of non-uniform data S_c and the uniform data S_t. Since the scale of S_c is usually much larger than that of S_t, this will lead to the inconsistency of training difficulty between M_c and M_t, that is, M_t will converge faster. This asynchrony will have an undesirable effect on the prediction alignment term, $\mathcal{L}^{S_u}(\hat{R}^t,\hat{R}^c)$. Finally, the overall training is unstable. To alleviate this problem, we first pretrain M_c and M_t. Subsequently, we refine the pretrained models again according to the above loss function. The pseudo-code of DUB is shown in Algorithm 1.

Note that, similar to most existing debiasing methods, our DUB does not depend on a specific model architecture when deploying or applying it in practice. The process of integrating our DUB into any recommendation model is as follows. (1) After collecting a non-randomized dataset S_c and a randomized dataset S_t, we pretrain a recommendation model M_c and an auxiliary model M_t based on a traditional optimization objective function and an arbitrary recommendation model, respectively (lines 1 and 2 of Algorithm 1). (2) In the model refinement stage, we need to modify only the optimization objective function of these models to that of DUB in the training stage;

according to the type of loss function used, we choose Equation (12) or Equation (13) as the new objective function (lines 4–6 of Algorithm 1).

ALGORITHM 1: Debiased Upper Bound With a Randomized Dataset (DUB)

Require: A non-randomized dataset S_c and a randomized dataset S_t.

1: Train a pretrained recommendation model M_c based on a backbone model on S_c.
2: Train a pretrained auxiliary model M_t based on a backbone model on S_t.
3: **Repeat**
4: An auxiliary set S_a with the same size as the training sample is randomly sampled from the unobserved feedback S_u;
5: Based on S_c, S_t, and S_a, use the pretrained M_c and M_t to calculate each loss term in Equation (12) or Equation (13) (according to the conditions satisfied by the adopted loss function);
6: Update the parameters of the recommendation model M_c.
7: **Until** convergence

5 ANALYSIS OF EXISTING METHODS

In this section, we will introduce and analyze some existing methods. In contrast to the proposed method, we show that these methods optimize only some terms in the generalization error bounds of the unbiased ideal loss function or optimize some weak proxy of these terms, that is, they provide insufficient optimization of the generalization error bound. This means that these methods may converge to only a suboptimal solution. Note that insufficient optimization for the generalization error bound is different from a more compact generalization error bound. The former means that the model considers only some optimization items and ignores the constraints on some optimization items during the training process. This may lead to the fact that although some optimization terms are gradually minimized, the generalization error bound may be unchanged, and even grow in reverse, due to the gradual increase in the loss of the neglected optimization terms. The latter means that it is closer to the ideal optimization objective function than the other generalization error bounds.

5.1 Causal Embeddings

Causal Embeddings (CausE) [35] is pioneering work in counterfactual recommendation. By introducing causal inference into the representation learning of recommendation, CausE is implemented in a multi-task learning framework, including a treatment task loss (M_c's own supervised loss), a control task loss (M_t's own supervised loss), and a regularizer between tasks (the parameter alignment terms of M_c and M_t). The loss function of CausE can be written as follows,

$$\min_{\mathcal{W}_c,\mathcal{W}_t} \underbrace{\mathcal{L}^{S_c}(R^c,\hat{R}^c)}_{(c)} + \underbrace{\mathcal{L}^{S_t}(R^t,\hat{R}^t)}_{(e.1)} + \lambda_c \mathrm{Reg}\left(\mathcal{W}_c\right) + \lambda_t \mathrm{Reg}\left(\mathcal{W}_t\right) + \gamma_{\mathrm{tc}}^{\mathrm{CausE}} \underbrace{\|\mathcal{W}_t - \mathcal{W}_c\|_F}_{(d)}, \tag{14}$$

where γ_{tc}^{CausE} is the weight parameter of the alignment term between M_c and M_t.

By comparing Equation (14) with Theory 4.2, the objective function of CausE can be regarded as a combination of term (c), term $(e.1)$, and a proxy of term (d) ($\|\mathcal{W}_t - \mathcal{W}_c\|_F$). Similarly, in comparison with Theory 4.5, it can be regarded as a combination of term (c) and a proxy of term (d). This means that CausE is an insufficient optimization of the generalization error bound. In addition, we find that the parameter alignment term may not be a reasonable proxy for the term (d): (1) The parameter alignment term restricts the parameters of M_c and M_t to have the same

dimension. However, in view of the difference in data scale between S_c and S_t, this constraint may be too strong. (2) The alignment of the parameters will cause difficulty in training in the case of high dimensions and multi-layer networks. The lack of optimization for terms (a) and $(e.2)$ will also result in CausE not being able to make M_c fully benefit from S_t during training, especially when S_t has a particularly small scale.

5.2 Bridge Strategy

Recently, Liu et al. explained and resolved counterfactual recommendation from the perspective of knowledge distillation [20]. They propose a general knowledge distillation framework for counterfactual recommendation and list some practical solutions as examples. The Bridge strategy is one of these solutions with the best performance, which also best matches our focus. The Bridge strategy first ensures the supervised loss of M_c and M_t. In addition, an auxiliary set S_a is randomly sampled from $\mathcal{D}$ in each iteration, and the predictions of M_c and M_t in S_a are constrained to be close. Note that most of S_a belongs to S_u because of the data sparsity in recommender systems. The loss function of the Bridge strategy can be rewritten as follows,

$$\min_{\mathcal{W}_c,\mathcal{W}_t} \underbrace{\mathcal{L}^{S_c}(R^c,\hat{R}^c)}_{(c)} + \underbrace{\mathcal{L}^{S_t}(R^t,\hat{R}^t)}_{(e.1)} + \gamma\mathcal{L}^{S_a}\underbrace{(\hat{R}^t,\hat{R}^c)}_{(d)} + \lambda_c \mathrm{Reg}(\mathcal{W}_c) + \lambda_t \mathrm{Reg}(\mathcal{W}_t). \tag{15}$$

By comparing Equation (15) with Theory 4.2, the objective function of Bridge can be regarded as a combination of terms (c), $(e.1)$, and (d). Similarly, in comparison with Theory 4.5, it can be regarded as a combination of terms (c) and (d). This means that the Bridge strategy is also an insufficient optimization of the generalization error bound. However, it optimizes term (d) directly instead of using a weak proxy and, thus, achieves a better performance in the experiments [20]. Similarly, the lack of optimization for terms (a) and $(e.2)$ can also cause Bridge to fail to make M_c fully benefit from S_t in some cases.

5.3 Remarks

Note that our discussion does not include another recent method, AutoDebias [7], in which meta-learning is introduced into a doubly robust (DR) framework to learn better unbiased parameters. On one hand, it can be seen as an improvement in the training process rather than in the loss function, which is different from our DUB as well as the existing methods mentioned earlier. On the other hand, the DR framework is also a representative method in another route without a randomized dataset [10, 40], that is, a randomized dataset is not necessary. Therefore, it is difficult to put it into a specific category. Theoretical insights on debiased recommendation are also provided in [7], which are quite different from our DUB, however. The authors aim to analyze the theoretical generalization error bound of AutoDebias, whereas we directly optimize a proxy of the upper bound derived from the unbiased ideal loss function in Equation (3).

6 EMPIRICAL EVALUATION

In this section, we conduct experiments with the aim of answering the following five key questions. The source codes and results are available at https://github.com/dgliu/TOIS_DUB.

- RQ1: How does the proposed method perform against the baselines in an unbiased evaluation?
- RQ2: What is the role of the additional terms in the loss function of the proposed method (i.e., the ablation studies of our DUB)?
- RQ3: What impact does the proposed method have on the item distribution of the recommendation lists?

Table 2. Statistics of the Datasets

	Yahoo! R3		Product	
	#Feedback	P/N	#Feedback	P/N
S_c	311,704	67.02%	4,798,776	12.97%
S_t	5,400	9.05%	34,755	0.99%
S_{va}	5,400	9.31%	34,755	0.75%
S_{te}	43,200	9.76%	278,043	0.88%

P/N denotes the ratio between the numbers of positive and negative feedback.

- RQ4: How do some key factors affect the performance of the proposed method?
- RQ5: How does the proposed method perform against the baselines in a general biased evaluation?

6.1 Experimental Setup

6.1.1 Datasets. To evaluate the performance of the model in an ideal unbiased scenario, we need to use a dataset containing a randomized subset. We thus use the following two datasets in the experiments; the statistics are shown in Table 2.

- **Yahoo! R3** [26]: This is the most commonly adopted standard dataset in previous works, including a user subset and a random subset. The former can be regarded as being collected under a stochastic recommendation policy, while the latter corresponds to a uniform policy. We binarize the ratings via a threshold $\epsilon = 3$, where a rating $> \epsilon$ is considered as a positive feedback ($R_{ij} = 1$); otherwise, it is a negative feedback ($R_{ij} = 0$). The user subset is used as a training set in a biased environment (S_c). For the random subset, we randomly split the user-item interactions into three subsets, including 10% for training in an unbiased environment (S_t), 10% for validation to tune the hyper-parameters (S_{va}), and 80% for the test (S_{te}).
- **Product**: This is a large-scale dataset for CTR prediction, which includes two weeks of users' click records from a real-world advertising system. The dataset contains two subsets: a subset (S_c) logged by several traditional ranking policies and a subset (S_t) logged by a uniform policy π_t. To remove the effect of the position bias in our experiments, we filter out the samples at positions 1 and 2. The dataset covers 217 displayed ads and more than two million users. To get the training set, validation set, and test set from the uniform subset, we randomly split the S_t subset using the same proportions as that for Yahoo! R3.

6.1.2 Backbones. The debiasing methods are usually model agnostic and are integrated into some backbone models. To comprehensively evaluate generalization ability, we use two representative shallow and deep models as the backbone models in the experiments: matrix factorization (MF) [15] and neural collaborative filtering (NCF) [11]. Similar settings can be found in previous works [5, 33, 35, 42].

6.1.3 Baselines. For the basic model, it can be regarded as three variants according to the different data sources used: training only with a non-randomized dataset S_c, training only with a randomized dataset S_t, and training with both data (i.e., $S_c \cup S_t$). We call the latter two variants *Unif* and *Combine* in the experiments. For debiased recommendation models, we choose the representative methods among the three lines summarized in Section 1. For the first line, the inverse propensity score (IPS) [35] is one of the most classic methods, which thus also serves as one of

Table 3. Hyper-parameters Tuned in the Experiments

Name	Range	Functionality
rank	{50, 100, 200}	Embedded dimension
λ	$\left\{1e^{-5}, 1e^{-4} \cdots 1e^{-1}\right\}$	Regularization
γ	$\left\{1e^{-5}, 1e^{-4} \cdots 1e^{-1}\right\}$	Loss weighting

our baselines. We adopt the naïve Bayes estimator in [35] to estimate the propensity score. For the second line, a recent method AutoDebias is introduced, in which the information of a randomized dataset is used more effectively by combining meta-learning strategies in a doubly robust framework [7]. For the third line, as described in Section 5, CausE [5] and Bridge [20] are two important baselines because they are the state-of-the-art methods that best match our focus.

6.1.4 Evaluation Metrics. We employ four evaluation metrics that are widely used in recommender systems: precision (P@K), recall (R@K), the area under the ROC curve (AUC) and normalized discounted cumulative gain (nDCG). We choose AUC as our main evaluation metric because it is one of the most important metrics in industry and previous works on debiasing. We report the results with K set to 5 and 10. The candidate items to be recommended for a user are from the set of items that have not had interaction from the user.

6.1.5 Implementation Details. All methods are implemented on TensorFlow 1.2 [1] except *AutoDebias*, which refers to its official PyTorch [31] version. We use the Adam [14] optimizer and cross-entropy loss in the experiments, that is, we choose Equation (13) as the optimization objective of the model. The learning rate is fixed as $1e^{-3}$. By evaluating the AUC on the validation data S_{va}, we perform grid search to tune the hyper-parameters for the candidate methods. To avoid over-fitting, we adopt an early stopping mechanism with the patience set to 5 times. The range of the values of the hyper-parameters are shown in Table 3.

6.2 RQ1: Comparison Results of Unbiased Evaluation

We report the comparison results of the unbiased evaluation in Tables 4 and 5. For the Yahoo! R3 dataset, as shown in Table 4, the proposed method outperforms all baselines in most cases except on P@5 and R@5, for which NCF is used as the backbone model. We have the following observations: (1) The baselines based on the use of a randomized dataset usually have a better performance than the basic model but may suffer from a performance bottleneck in some cases. (2) The performance of the baseline AutoDebias depends on the backbone model used, which may be because the designed meta-learning strategy is mainly for low-rank models. (3) In contrast, our DUB is relatively stable for different backbone models. For the Product dataset, as shown in Table 5, the proposed method consistently outperforms all baselines on the AUC, and maintains advantages on other metrics in most cases. We can get similar observations as that on Yahoo! R3. Note that since the baseline AutoDebias has a prediction step for all of the unobserved samples, it requires far more memory than that of a single GPU (e.g., 32G) and a specific parallelization. This weakens its scalability, and we do not report its results. In general, our DUB is relatively stable for datasets of different sizes.

6.3 RQ2: Results of Ablation Studies

As described in Section 4, the proposed method further improves performance by sufficiently optimizing the upper bound of the unbiased ideal loss function. A key question is what the role of

Table 4. Comparison Results of Unbiased Testing on Yahoo! R3

Method	AUC	nDCG	P@5	P@10	R@5	R@10
MF	0.7282	0.0434	0.0059	0.0051	0.0207	0.0332
Unif-MF	0.5625	0.0291	0.0049	0.0041	0.0135	0.0245
Combine-MF	0.7357	0.0489	0.0073	0.0061	0.0243	0.0401
IPS-MF	0.7300	0.0407	0.0052	0.0054	0.0171	0.0344
AutoDebias-MF	0.7502	0.0691	0.0119	0.0104	0.0403	0.0683
CausE-MF	0.7285	0.0445	0.0059	0.0058	0.0192	0.0372
Bridge-MF	0.7376	0.0557	0.0099	0.0076	0.0308	0.0478
DUB-MF	**0.7578***	**0.0727**	**0.0128**	**0.0112**	**0.0438**	**0.0770**
NCF	0.7245	0.0279	0.0029	0.0031	0.0089	0.0199
Unif-NCF	0.6050	0.0275	0.0043	0.0037	0.0113	0.0204
Combine-NCF	0.7268	0.0327	0.0032	0.0033	0.0092	0.0189
IPS-NCF	0.7273	0.0304	0.0036	0.0031	0.0111	0.0210
AutoDebias-NCF	0.7140	0.0385	0.0052	0.0047	0.0188	0.0333
CausE-NCF	0.7284	0.0287	0.0029	0.0033	0.0089	0.0210
Bridge-NCF	0.7367	0.0439	**0.0056**	0.0056	**0.0192**	0.0371
DUB-NCF	**0.7421***	**0.0491**	0.0051	**0.0058**	0.0164	**0.0390**

The best results are marked in bold. AUC is the main evaluation metric. Note that * indicates a significance level $p \leq 0.05$ based on two sample t-tests between the best and second best results.

the additional optimization terms is in our method. To answer this question, we conduct ablation studies of the proposed method by removing certain terms. The results are shown in Tables 6 and 7. Note that after removing terms (a) and (e), our DUB is equivalent to the Bridge strategy. Thus, we do not remove more terms in the experiments. We can see that removing any term will hurt the performance in most cases, and removing more terms results in worse performance. There are some unexpected cases in Table 6, for example, when K takes a small value, the full version with NCF as the backbone model has a slight disadvantage on a few metrics. This may be due to the noise caused by considering only the AUC as the evaluation metric in parameter tuning. In general, all terms in the proposed method can synergistically produce the greatest gain.

6.4 RQ3: Item Distribution of the Recommendation Lists

An interesting question concerns the difference between the distributions of the recommendation lists of the proposed method and the baseline methods. To answer this question, we show in Figure 3 the item distribution of the recommendation lists generated by different methods, in which popular items are the 20% most frequent items in the training set and the rest are unpopular items. Figure 3(a) is the distribution of a randomized dataset, from which we can find that although the probability of popular and unpopular items being recommended is even (e.g., popular items account for 20% of the total items and the probability of being recommended also accounts for 20%), the utility (i.e., the probability of hit divided by the probability of being recommended) brought by popular items is higher. This means that a practical ideal recommendation strategy may not excessively pursue a balance between popular and unpopular items. Note that for the brevity of the legend in the figure, we use the abbreviation Auto to refer to the baseline AutoDebias.

Table 5. Comparison Results of Unbiased Testing on Product

Method	AUC	nDCG	P@5	P@10	R@5	R@10
MF	0.7115	0.0434	0.0105	0.0103	0.0518	0.1017
Unif-MF	0.6372	0.0604	0.0148	0.0135	0.0737	0.1332
Combine-MF	0.7145	0.0526	0.0121	0.0113	0.0601	0.1111
IPS-MF	0.7274	0.0484	0.0115	0.0114	0.0568	0.1114
AutoDebias-MF	-	-	-	-	-	-
CausE-MF	0.7158	0.0470	0.0107	0.0114	0.0529	0.1119
Bridge-MF	0.7069	0.0438	0.0107	0.0104	0.0529	0.1022
DUB-MF	**0.7374***	**0.0729**	**0.0158**	**0.0155**	**0.0787**	**0.1537**
NCF	0.7293	0.0616	0.0152	0.0131	0.0753	0.1299
Unif-NCF	0.6240	0.0557	0.0131	0.0132	0.0651	0.1307
Combine-NCF	0.7301	0.0674	0.0155	0.0142	0.0773	**0.1410**
IPS-NCF	0.7328	0.0616	0.0155	0.0126	0.0773	0.1249
AutoDebias-NCF	-	-	-	-	-	-
CausE-NCF	0.7351	0.0623	0.0158	0.0125	0.0789	0.1235
Bridge-NCF	0.7149	0.0628	0.0145	0.0126	0.0723	0.1255
DUB-NCF	**0.7382***	**0.0686**	**0.0165**	**0.0149**	**0.0851**	0.1380

• Note: the placeholder '-' means that the result is not reported because the memory space required by this method exceeds that of the GPU used.
The best results are marked in bold. AUC is the main evaluation metric. Note that * indicates a significance level $p \leq 0.05$ based on two sample t-tests between the best and second best results.

Table 6. Results of the Ablation Studies on Yahoo! R3

Method	AUC	nDCG	P@5	P@10	R@5	R@10
DUB-MF	**0.7578**	**0.0727**	**0.0128**	**0.0112**	**0.0438**	**0.0770**
w/o term $(e.2)$	0.7500	0.0702	0.0113	0.0108	0.0377	0.0744
w/o terms (a) & $(e.2)$	0.7376	0.0557	0.0099	0.0076	0.0308	0.0478
DUB-NCF	**0.7421**	**0.0491**	0.0051	**0.0058**	0.0164	**0.0390**
w/o term $(e.2)$	0.7386	0.0438	0.0050	0.0051	0.0165	0.0320
w/o terms (a) & $(e.2)$	0.7367	0.0439	**0.0056**	0.0056	**0.0192**	0.0371

The best results are marked in bold. AUC is the main evaluation metric.

Combining Figures 3(b) and 3(c), we can observe the following. (1) MF, IPS, and CausE tend to capture the recommendation patterns of popular and unpopular items similar to Figure 3(a), but unreasonably displaying too many unpopular items may not provide much benefit and will even cause user distrust. (2) AutoDebias can capture the utility information of popular items, but it tends to overexpose the popular items, which may also hurt the user experience. Note that our

Table 7. Results of the Ablation Studies on Product

Method	AUC	nDCG	P@5	P@10	R@5	R@10
DUB-MF	**0.7374**	**0.0729**	**0.0158**	**0.0155**	**0.0787**	**0.1537**
w/o term $(e.2)$	0.7091	0.0453	0.0115	0.0105	0.0571	0.1039
w/o terms (a) & $(e.2)$	0.7069	0.0438	0.0107	0.0104	0.0529	0.1022
DUB-NCF	**0.7382**	**0.0686**	**0.0165**	**0.0149**	**0.0851**	**0.1380**
w/o term $(e.2)$	0.7284	0.0648	0.0162	0.0132	0.0806	0.1313
w/o terms (a) & $(e.2)$	0.7149	0.0628	0.0145	0.0126	0.0723	0.1255

The best results are marked in bold. AUC is the main evaluation metric.

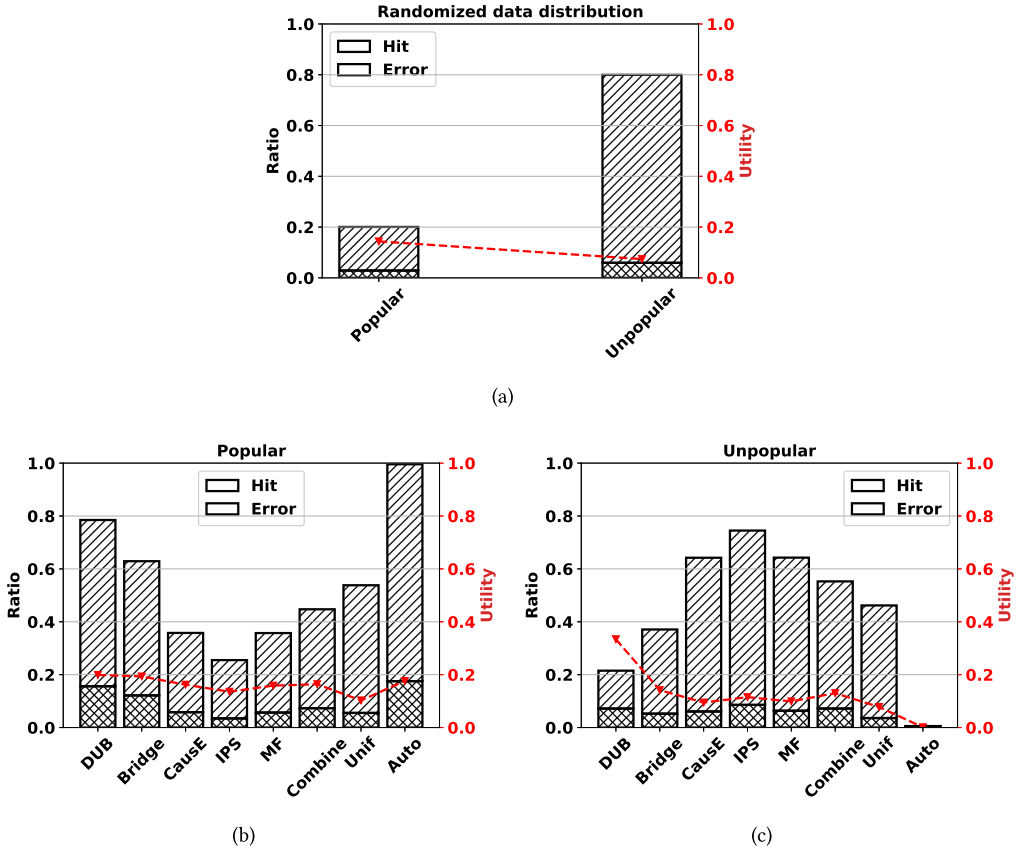


Fig. 3. Item distribution and utility of a randomized dataset and different methods with Yahoo! R3.

results differ somewhat from those in [7]. As described in Section 6.1.1, during data processing, we set the labels of positive and negative feedback to 1 and 0, respectively, to be compatible with the prediction layers with a sigmoid activation. However, the labels for positive and negative feedback in [7] are set to 1 and −1, respectively. (3) Our DUB keeps recommending popular items with high

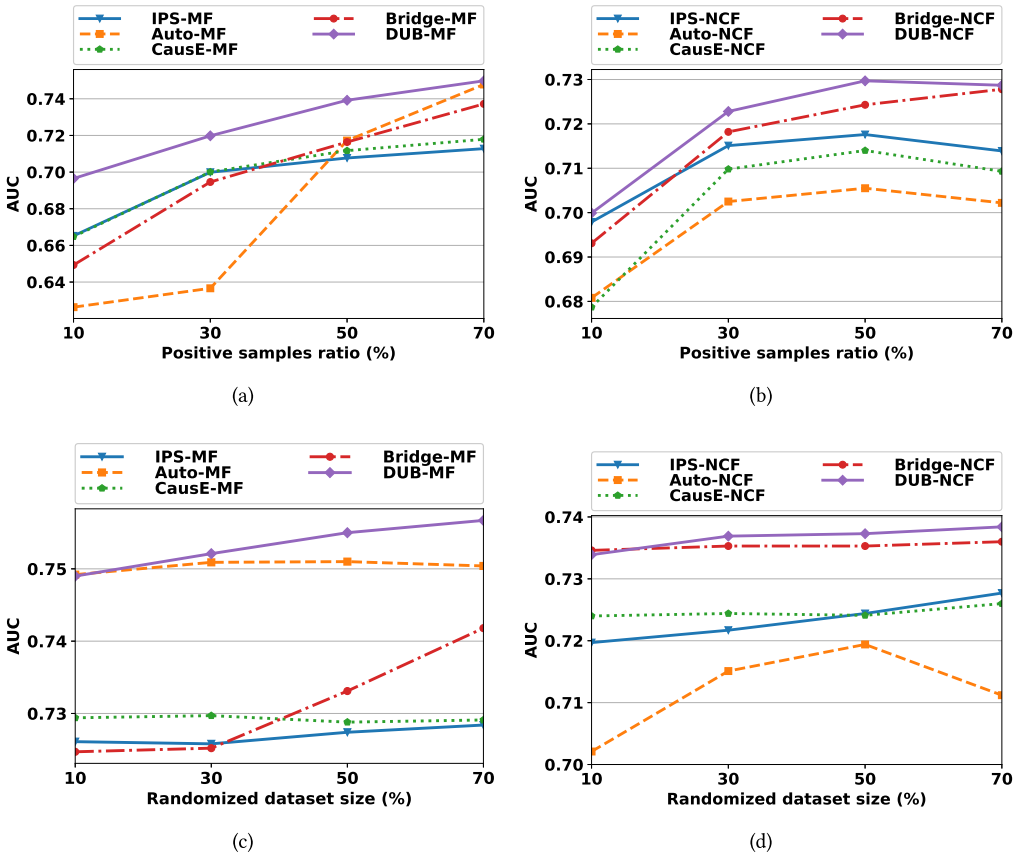


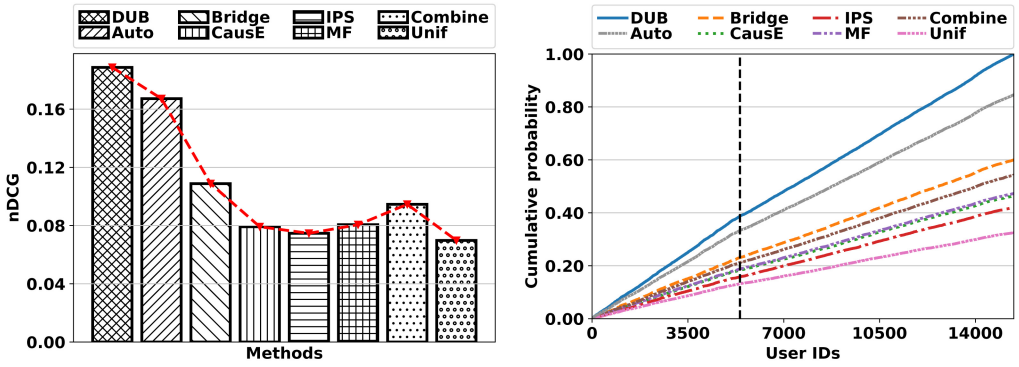
Fig. 4. The results of analysis of the key factors on Yahoo! R3, where (a) and (b) are considering that S_c has different positive sample ratios and (c) and (d) are considering that S_t has different data sizes.

utility and carefully displays the unpopular items with a higher hit rate and achieves the highest utility among unpopular items, that is, the DUB can more effectively weigh the use of information between a randomized dataset and a non-randomized dataset.

6.5 RQ4: Analysis Results of Key Factors

We further analyze some key factors that may affect the performance of the methods. The first key factor is the difference in the ratio of positive and negative samples between S_c and S_t. When S_c and S_t are too different, the difficulty of training the model will greatly increase. However, when S_c and S_t are too close, the assimilation will seriously damage the guiding role of S_t. In the experiments, we fix the size of a subset sampled from S_c to be 135,000 to include as many positive samples as possible. Then, we control this subset to contain a certain proportion of positive samples: we randomly sample $135{,}000 * ratio$ positive samples and $135{,}000 * (1 - ratio)$ negative samples from S_c. We set this ratio to 10%, 30%, 50%, and 70%, respectively. Note that when 10% is taken, the distribution of this subset is closest to that of S_t. From Figures 4(a) and 4(b), we find that our DUB consistently outperforms all of the baselines in all cases.

The second key factor that may affect the performance of the model is the size of S_t. As described in Section 3.1, the scale and scope of S_t is much smaller than that of S_c. When the number of S_t



(a) (b)

Fig. 5. (a) Comparison results in general evaluation. (b) The cumulative hit probability of different methods at the user level. Note that we use Yahoo! R3 in this study.

is smaller than a certain value, it can hardly guide S_c. By observing the performance trend of the model under different sizes of S_t, we can have a preliminary understanding of this lower bound. In the experiments, we keep the same data settings as the previous experiments, except that S_t is randomly sampled according to a certain proportion to obtain a subset. We set this ratio to 10%, 30%, 50%, and 70%, respectively. From Figures 4(c) and 4(d), we find that our DUB is also stable and accurate in all cases.

6.6 RQ5: Comparison Results of General Evaluation

Although using unbiased data for verification and evaluation is a promising choice, it also has some limitations because it may not cover all users and items. We are also interested in the performance of the proposed method and baselines in general evaluation with biased but high coverage, that is, both validation and testing use the non-uniform data. In the experiments, we randomly divide S_c according to the proportion of $5:2:3$ to obtain a training set, a validation set, and a test set. S_t is still used as the unbiased training set. We use the same settings in Section 6.1.5 to search for the best values, except that the reference metric becomes nDCG because it is one of the most adopted metrics in general evaluation. We can see from Figure 5(a) that our DUB and AutoDebias show a significant improvement over the other baselines. This is reasonable because their ability to capture the utility of popular items (as shown in Figure 3) can play a greater role in general evaluation. We show in Figure 5(b) the cumulative hit probability of different methods at the user level (i.e., the sum of the hit probabilities of the first x users), and find that introducing S_t in general evaluation is beneficial to better learn the corresponding preferences of the users involved in S_t (i.e., the first 5,400 users).

7 CONCLUSIONS AND FUTURE WORK

In this article, we propose a new debiased perspective based on directly optimizing the upper bound of an ideal objective function to facilitate the introduction of some theoretical insights and a more sufficient solution to the system-induced biases. We first formulate a new unbiased ideal loss function to more fully reduce the data bias when a small randomized dataset is available and then provide some theoretical insights about its upper bound. Moreover, we point out that most existing methods can be regarded as insufficient optimization of the upper bound. In response, we propose a novel method, debiasing approximate upper bound (DUB) with a randomized dataset, for

a more sufficient optimization of the upper bound. Finally, we conduct extensive empirical studies to show the effectiveness of the proposed method and explore the impact of some key factors that may affect performance.

For future works, we will obtain different upper bounds of the unbiased ideal loss function in different ways and comparatively evaluate them. We also plan to gain more theoretical insights on other ways of using a randomized dataset in debiased recommendation. We are also interested in exploring new techniques for debiased recommendation with only one non-randomized dataset or multiple non-randomized datasets.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their expert and constructive comments and suggestions.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th Symposium on Operating Systems Design and Implementation*. USENIX Association, Berkeley, CA, 265–283.

[2] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling popularity bias in learning-to-rank recommendation. In *Proceedings of the 11th ACM Conference on Recommender Systems*. ACM, Como, 42–46.

[3] Aman Agarwal, Xuanhui Wang, Cheng Li, Michael Bendersky, and Marc Najork. 2019. Addressing trust bias for unbiased learning-to-rank. In *Proceedings of the Web Conference 2019*. ACM, San Francisco, CA, 4–14.

[4] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. 2019. Estimating position bias without intrusive interventions. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*. ACM, Melbourne, 474–482.

[5] Stephen Bonner and Flavian Vasile. 2018. Causal embeddings for recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, Vancouver, BC, 104–112.

[6] Rocío Cañamares and Pablo Castells. 2018. Should I follow the crowd?: A probabilistic analysis of the effectiveness of popularity in recommender systems. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Ann Arbor, MI, 415–424.

[7] Jiawei Chen, Hande Dong, Yang Qiu, Xiangnan He, Xin Xin, Liang Chen, Guli Lin, and Keping Yang. 2021. AutoDebias: Learning to debias for recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Montréal, QC, 21–30.

[8] Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. 2017. Joint distribution optimal transportation for domain adaptation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Curran Associates Inc., Long Beach, CA, 3730–3739.

[9] Prem Gopalan, Jake M. Hofman, and David M. Blei. 2015. Scalable recommendation with hierarchical Poisson factorization. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*. AUAI Press, Arlington, VA, 326–335.

[10] Siyuan Guo, Lixin Zou, Yiding Liu, Wenwen Ye, Suqi Cheng, Shuaiqiang Wang, Hechang Chen, Dawei Yin, and Yi Chang. 2021. Enhanced doubly robust learning for debiasing post-click conversion rate estimation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Montréal, QC, 275–284.

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the Web Conference 2017*. ACM, Perth, 173–182.

[12] Wassily Hoeffding. 1994. Probability inequalities for sums of bounded random variables. In *The Collected Works of Wassily Hoeffding*. Springer, New York, NY, 409–426.

[13] Alexandre Kaspar, Tae-Hyun Oh, Liane Makatura, Petr Kellnhofer, and Wojciech Matusik. 2019. Neural inverse knitting: From images to manufacturing instructions. In *Proceedings of the 36th International Conference on Machine Learning*. PMLR, Long Beach, CA, 3272–3281.

[14] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). https://arxiv.org/abs/1412.6980

[15] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[16] Jongyeong Lee, Nontawat Charoenphakdee, Seiichi Kuroki, and Masashi Sugiyama. 2019. Domain discrepancy measure for complex models in unsupervised domain adaptation. *arXiv preprint arXiv:1901.10654* (2019). https://arxiv.org/abs/1901.10654

[17] Jae-woong Lee, Seongmin Park, and Jongwuk Lee. 2021. Dual unbiased recommender learning for implicit feedback. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Montréal, QC, 1647–1651.

[18] Dawen Liang, Laurent Charlin, James McInerney, and David M. Blei. 2016. Modeling user exposure in recommendation. In *Proceedings of the Web Conference 2016*. ACM, Montréal, QC, 951–961.

[19] Chen Lin, Dugang Liu, Hanghang Tong, and Yanghua Xiao. 2022. Spiral of silence and its application in recommender systems. *IEEE Transactions on Knowledge and Data Engineering* 34, 6 (2022), 2934–2947.

[20] Dugang Liu, Pengxiang Cheng, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2020. A general knowledge distillation framework for counterfactual recommendation via uniform data. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Xi'an, 831–840.

[21] Dugang Liu, Pengxiang Cheng, Zinan Lin, Jinwei Luo, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2022. KDCRec: Knowledge distillation for counterfactual recommendation via uniform data. *IEEE Transactions on Knowledge and Data Engineering* (2022). https://doi.org/10.1109/TKDE.2022.3199585

[22] Dugang Liu, Pengxiang Cheng, Hong Zhu, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2021. Mitigating confounding bias in recommendation via information bottleneck. In *Proceedings of the 15th ACM Conference on Recommender Systems*. ACM, Amsterdam, 351–360.

[23] Dugang Liu, Pengxiang Cheng, Hong Zhu, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2022. Debiased representation learning in recommendation via information bottleneck. *ACM Transactions on Recommender Systems* 1, 1 (2022), 5.

[24] Dugang Liu, Chen Lin, Zhilin Zhang, Yanghua Xiao, and Hanghang Tong. 2019. Spiral of silence in recommender systems. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*. ACM, Melbourne, 222–230.

[25] Yiming Liu, Xuezhi Cao, and Yong Yu. 2016. Are you influenced by others when rating? Improve rating prediction by conformity modeling. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, Boston, MA, 269–272.

[26] Benjamin M. Marlin and Richard S. Zemel. 2009. Collaborative prediction and ranking with non-random missing data. In *Proceedings of the 3rd ACM Conference on Recommender Systems*. ACM, New York City, NY, 5–12.

[27] Marco Morik, Ashudeep Singh, Jessica Hong, and Thorsten Joachims. 2020. Controlling fairness and bias in dynamic learning-to-rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Xi'an, 429–438.

[28] Harrie Oosterhuis and Maarten de Rijke. 2021. Unifying online and counterfactual learning to rank: A novel counterfactual estimator that effectively utilizes online interventions. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM, Jerusalem, 463–471.

[29] Zohreh Ovaisi, Ragib Ahsan, Yifan Zhang, Kathryn Vasilaky, and Elena Zheleva. 2020. Correcting for selection bias in learning-to-rank systems. In *Proceedings of the Web Conference 2020*. ACM, Taipei, 1863–1873.

[30] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Proceedings of the 8th IEEE International Conference on Data Mining*. IEEE, Pisa, 502–511.

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Vancouver, BC, 8026–8037.

[32] Judea Pearl and Dana Mackenzie. 2018. *The Book of Why: The New Science of Cause and Effect*. Basic Books, Inc., New York, NY.

[33] Yuta Saito. 2020. Asymmetric tri-training for debiasing missing-not-at-random explicit feedback. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Xi'an, 309–318.

[34] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased recommender learning from missing-not-at-random implicit feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. ACM, Houston, TX, 501–509.

[35] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. In *Proceedings of the 33rd International Conference on Machine Learning*. PMLR, New York City, NY, 1670–1679.

[36] Pannaga Shivaswamy and Ashok Chandrashekar. 2021. Bias-variance decomposition for ranking. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM, Jerusalem, 472–480.

[37] Harald Steck. 2013. Evaluation of recommendations: Rating-prediction and ranking. In *Proceedings of the 7th ACM Conference on Recommender Systems*. ACM, Hong Kong, 213–220.
[38] Wenjie Wang, Fuli Feng, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2021. Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Montréal, QC, 1288–1297.
[39] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. ACM, Los Angeles, CA, 610–618.
[40] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2019. Doubly robust joint learning for recommendation on data missing not at random. In *Proceedings of the 36th International Conference on Machine Learning*. PMLR, Long Beach, CA, 6638–6647.
[41] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2021. Combating selection biases in recommender systems with a few unbiased ratings. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM, Jerusalem, 427–435.
[42] Zifeng Wang, Xi Chen, Rui Wen, Shao-Lun Huang, Ercan E. Kuruoglu, and Yefeng Zheng. 2020. Information theoretic counterfactual learning from missing-not-at-random feedback. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*. Curran Associates Inc., Online, 1854–1864.
[43] Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. 2021. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, Singapore, 1791–1800.
[44] Xinwei Wu, Hechang Chen, Jiashu Zhao, Li He, Dawei Yin, and Yi Chang. 2021. Unbiased learning to rank in feeds recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM, Jerusalem, 490–498.
[45] Himank Yadav, Zhengxiao Du, and Thorsten Joachims. 2021. Policy-gradient training of fair and unbiased ranking functions. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Montréal, QC, 1044–1053.
[46] Haiqin Yang, Guang Ling, Yuxin Su, Michael R. Lyu, and Irwin King. 2015. Boosting response aware model-based collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering* 27, 8 (2015), 2064–2077.
[47] Jiangxing Yu, Hong Zhu, Chih-Yao Chang, Xinhua Feng, Bowen Yuan, Xiuqiang He, and Zhenhua Dong. 2020. Influence function for unbiased recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Xi'an, 1929–1932.
[48] Bowen Yuan, Jui-Yang Hsia, Meng-Yuan Yang, Hong Zhu, Chih-Yao Chang, Zhenhua Dong, and Chih-Jen Lin. 2019. Improving ad click prediction by considering non-displayed events. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM, Beijing, 329–338.
[49] Shuxi Zeng, Murat Ali Bayir, Joseph J. Pfeiffer III, Denis Charles, and Emre Kiciman. 2021. Causal transfer random forest: Combining logged data and randomized experiments for robust prediction. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM, Jerusalem, 211–219.
[50] Wenhao Zhang, Wentian Bao, Xiao-Yang Liu, Keping Yang, Quan Lin, Hong Wen, and Ramin Ramezani. 2020. Large-scale causal approaches to debiasing post-click conversion rate estimation with multi-task learning. In *Proceedings of the Web Conference 2020*. ACM, Taipei, 2775–2781.
[51] Xiaoying Zhang, Junzhou Zhao, and John C. S. Lui. 2017. Modeling the assimilation-contrast effects in online product rating systems: Debiasing and recommendations. In *Proceedings of the 11th ACM Conference on Recommender Systems*. ACM, Como, 98–106.
[52] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling user interest and conformity for recommendation with causal embedding. In *Proceedings of the Web Conference 2021*. ACM, Ljubljana, 2980–2991.
[53] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. 2021. Popularity-opportunity bias in collaborative filtering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM, Jerusalem, 85–93.

Received 6 February 2022; revised 18 September 2022; accepted 9 January 2023