Mitigating Confounding Bias in Recommendation via Information Bottleneck

Dugang Liu[*]
Shenzhen University
Shenzhen, China
dugang.ldg@gmail.com

Pengxiang Cheng[*]
Huawei Noah's Ark Lab
Shenzhen, China
chengpengxiang1@huawei.com

Hong Zhu
Zhenhua Dong
Xiuqiang He
Huawei Noah's Ark Lab
Shenzhen, China

Weike Pan[†]
Zhong Ming[†]
Shenzhen University
Shenzhen, China
panweike,mingz@szu.edu.cn

ABSTRACT

How to effectively mitigate the bias of feedback in recommender systems is an important research topic. In this paper, we first describe the generation process of the biased and unbiased feedback in recommender systems via two respective causal diagrams, where the difference between them can be regarded as the source of bias. We then define this difference as a confounding bias, which can be regarded as a collection of some specific biases that have previously been studied. For the case with biased feedback alone, we derive the conditions that need to be satisfied to obtain a debiased representation from the causal diagrams. Based on information theory, we propose a novel method called debiased information bottleneck (DIB) to optimize these conditions and then find a tractable solution for it. In particular, the proposed method constrains the model to learn a biased embedding vector with independent biased and unbiased components in the training phase, and uses only the unbiased component in the test phase to deliver more accurate recommendations. Finally, we conduct extensive experiments on a public dataset and a real product dataset to verify the effectiveness of the proposed method and discuss its properties.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Confounding bias, Causal diagrams, Recommender systems, Information bottleneck

[*]Equal contribution
[†]Co-corresponding authors

ACM Reference Format:
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 *Fifteenth ACM Conference on Recommender Systems (RecSys '21), September 27-October 1, 2021, Amsterdam, Netherlands.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3460231.3474263

1 INTRODUCTION

As a feedback loop system, a recommender system is associated with various biases during the interaction between the user and the system, such as position bias [3, 38], selection bias [27, 32] and popularity bias [1, 6]. Ignoring these biases will cause a recommendation model to converge to a biased sub-optimal solution, and have harmful effects on the recommender system and the users, such as filter bubbles [16], echo chambers [11] and unfairness [10]. Therefore, how to effectively alleviate the bias of the feedback data collected in a recommender system is an important problem.

The previous works solving the bias problem in recommender systems mainly include the following four lines, i.e., heuristic-based methods [25, 43], inverse propensity score-based methods [32, 44, 45], unbiased data augmentation methods [5, 23, 39, 46], and some theoretical tools-based methods [30, 31]. The first line assumes that user feedback depends on certain specific factors and models this relationship, such as item features [12, 20] and public opinions [22, 24]. The second line uses the inverse propensity score as the sample weight to adjust the biased feedback distribution. The third line introduces a special uniform data as an unbiased target data to guide the training of the biased feedback. The last line aims to couple certain theoretical tools with the bias problem, and uses these theoretical tools to design some debiasing models, such as information bottleneck and causal inference techniques [37, 40–42]. However, most methods ignore the bias generation process, and thus may only be applicable to a certain type of bias problem.

In this paper, inspired by [13, 18], we first describe the generation process of the biased feedback and unbiased feedback in recommender systems via two respective causal diagrams, where the difference between them can be regarded as the source of bias. We define this difference as a *confounding bias*, which can be regarded as a collection of some specific biases that have been studied in previous works. To simplify and match the main models in the recommendation field, we generally assume that the confounding bias will be reflected in the embedding representation of a recommendation model trained with a biased feedback data. Moreover, we propose a *debiased information bottleneck* (DIB) objective function to alleviate the confounding bias in the biased feedback without an unbiased data.

Specifically, the proposed method is based on our observations in the causal diagrams of the feedback generation process described above. In the training phase, we constrain the model to learn a special *biased embedding vector*, including a biased component responsible for the effect of the confounding bias, and an unbiased component responsible for the effect of the user's true preference. To remove the influence of the confounding bias in the test phase, we only retain the unbiased component in the embedding vector in the process of recommending items, i.e., a *debiased embedding vector.* The proposed method has better interpretability because it is directly derived from the causal diagram of the bias generation process. In addition, the proposed method can be used to solve a more general bias problem because the confounding bias is essentially a fusion of some specific biases. Finally, we conduct extensive experiments on a public dataset and a real product dataset to verify the effectiveness of the proposed method, including standard unbiased tests, ablation studies, and some in-depth analysis of the proposed method.

2 RELATED WORK

2.1 Debiasing in Recommender Systems

Solving the bias problem in recommender systems is an important topic that has gradually received more attention by both the researchers and practitioners from the academia and industry. The existing works on debiasing in recommender systems can be categorized into four classes, including heuristic-based methods, inverse propensity score-based methods [32, 44, 45], unbiased data augmentation methods [5, 23, 39, 46], and some theoretical tools-based methods. A heuristic-based method links a user's feedback with different specific factors, such as item features [12, 20], user ratings [25, 43] and public opinions [22, 24], and uses some probabilistic graphical models for modeling. An inverse propensity score-based method corrects the biased feedback distribution by introducing some sample weights. An unbiased data augmentation method guides the model training on biased feedback by introducing an unbiased data collected by a special uniform policy. Since the collection process of an unbiased data is very expensive in practice, some recent methods directly perform debiasing on one single biased data with some theoretical tools, such as information bottleneck [40], upper bound minimization [31], asymmetric tri-training [30], and causal inference techniques [37, 41, 42].

2.2 Information Bottleneck

The information bottleneck method is a technique in information theory for finding the best trade-off between accuracy and complexity [35]. It has been regarded as a theoretical foundation of deep learning and been applied to many fields, such as robust or invariant representation learning [9, 26], disentangled representation learning [2], compressed representation learning [8], causal inference [28] and feature detection [33]. The works most relevant to ours in this line are [7, 15]. They focus on learning disentangled representation of texts through customized information bottleneck loss when the labels that only indicate the style information of texts are given. This representation includes independent style embedding and content embedding, which is similar to our expectation that the biased and unbiased components in the biased embedding vector are as independent as possible. However, these methods are not applicable to the recommender systems since the labels in the recommendation field indicate a mixture of user preferences and bias. In addition, as far as we know, there is only one work that considers the use of information bottleneck to solve the bias problem in recommender systems. That work is based on contrastive analysis of feature embedding, which proposes a counterfactual variational information bottleneck method to solve the selection bias in the recommender systems [40].

3 NOTATIONS AND PROBLEM FORMULATION

3.1 Notations

Let $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_d$ be a d-dimensional observed feature space and $\mathcal{Y} = \{0, 1\}$ be a label space. In this paper, we focus on solving the bias problem in recommender systems. Specifically, suppose we have the following N events

$$\left(x^1, y^1\right), \dots, \left(x^N, y^N\right), \tag{1}$$

where $x^i = \left(x_1^i, \cdots, x_d^i\right) \in \mathcal{X}$ and $y^i \in \mathcal{Y}$ are the feature vector and the label of the i-th event, respectively. According to the nature of feedback in a typical recommender system, we assume that the label y satisfies

$$y = \begin{cases} 1 & \text{the event was displayed and clicked,} \\ 0 & \text{the event was displayed but not clicked.} \end{cases} \tag{2}$$

An event represents an interaction between a user and a system. For example, a commercial system recommends a movie or displays an advertisement to a user. In particular, when no side information is available, i.e., only the user ID u and the item ID i are provided, an event can be simplified to $(x = (u, i), y)$.

The feedback events are used to train the recommendation model, which evaluates the users' preference on the item set as accurately as possible by learning a decision function $\hat{y}(x) \in \{-\infty, +\infty\}$. In practice, the decision function is usually implemented based on a low-rank model and a neural network model, which are also included in our experiments. Both of them apply an embedding vector z^* as a representation of the input x. Therefore, we can describe the decision function as a mapping process from the representation z^* to the label $\hat{y}$, which passes through one or more hidden layers h_j. This process can be formalized as a Markov chain of successive representations [40], $y \to x \to z^* \to h_1 \to \cdots \to h_L \to \hat{y}$.

3.2 Confounding Bias and Problem Formulation

To better understand the source of bias and address it in a targeted manner, in Figure 1(a), we show the generation process of feedback events in recommender systems from the perspective of causal

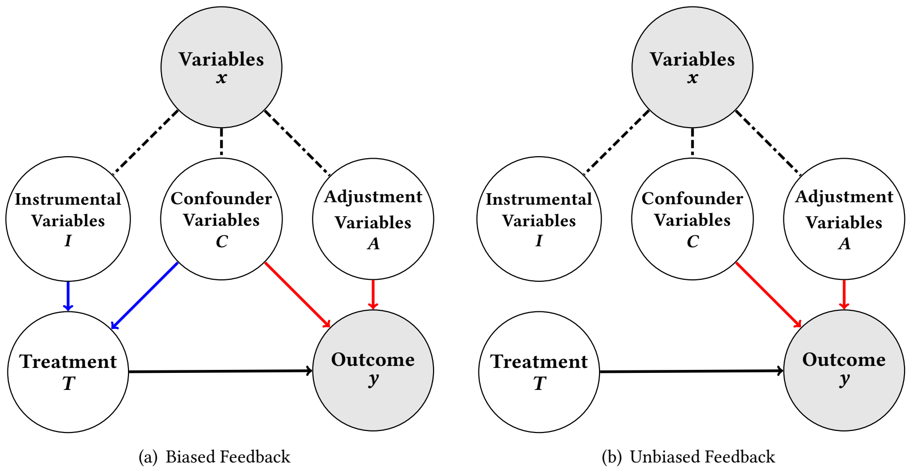


Figure 1: (a) A causal diagram of biased feedback, where x and y are known variables, blue arrows indicate the indirect effect and red arrows indicate the direct effect. (b) A causal diagram of unbiased feedback where the indirect effect are truncated.

inference. The input variables x (i.e., the feature vector) can be divided into three parts, including the instrumental variables I, the confounder variables C and the adjustment variables A. The instrumental variables I and the confounder variables C determine the treatment T, and they have an indirect effect on the outcome y (i.e., the label) through the path $\{I, C\} \rightarrow T \rightarrow y$. The confounder variables C and the adjustment variables A have a direct effect on y through the path $\{C, A\} \rightarrow y$. Similar causal diagrams can also be found in previous works on treatment effect estimation [13, 18], and we follow their assumptions, i.e., *strong ignorability* [29]. The term "treatment" in recommender systems can be thought of as referring to a recommendation strategy, this is, which items the system selects, as well as how the system organizes and shows these items to some specific users.

Different recommendation strategies will produce different indirect effects on feedback events by influencing the treatment T. This will lead to the inherent variability in the feedback events, and make most recommendation models that aim to minimize the error of the observed feedback not have good generalizability. Conversely, because the direct effect does not depend on the treatment, it can be considered as a stable and true user preference. This means that if we can cut off the indirect effect, i.e., we have a special strategy that only depends on the direct effect, then the collected feedback events are relatively stable and unbiased. We show the generation process of this unbiased feedback event in Figure 1(b). By comparing Figure 1(a) and 1(b), we call the bias brought by the recommendation strategy the *confounding bias*. Confounding bias is essentially a collection of biases at the system level, such as the position bias and popularity bias.

Definition 3.1 (Confounding Bias). *Suppose that the variables x, the outcome y, the indirect effect $\{I, C\} \rightarrow T \rightarrow y$ and the direct effect $\{C, A\} \rightarrow y$ from x to y are given. Confounding bias refers to the confusion of the observed feedback in recommender systems caused by the indirect effect.*

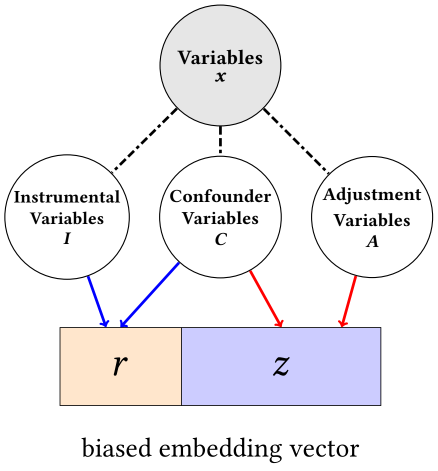
On the other hand, previous works have shown that the unbiased feedback can be obtained through a uniform policy [5, 23, 39, 46], i.e., for user requests, the system randomly samples items from a candidate set, and displays them after some random arrangement. However, the uniform policy will harm the user experience and reduce the platform revenue. Hence, a more appealing scenario is that only the biased feedback is available. Therefore, in this paper, we focus on mitigating the confounding bias with the biased feedback alone.

4 THE PROPOSED METHOD

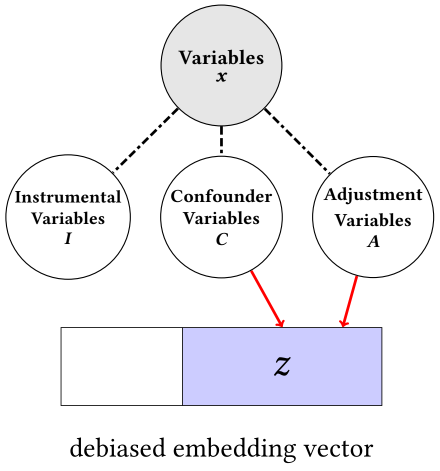
In practice, it may be difficult to directly group the variables x to obtain the instrumental variables I, the confounder variables C and the adjustment variables A. Instead, since the goal of most recommendation models is to learn an accurate embedding representation vector derived from the variables x, we make a general assumption, which is intuitively reasonable. The representation z^*, which is a good proxy for the variables x, naturally suffers from the influence of the biased variables in x. Since the correspondence between the biased variables and the semantics of the dimension in z^* is difficult to be obtained, z^* is also indistinguishable.

Assumption 4.1. *The confounding bias will be reflected in the embedding representation learned by a traditional recommendation model, and the bias will usually corrupt all the dimensions, i.e., the original representation z^* is biased and indistinguishable.*

Based on this assumption, we illustrate the main idea of the proposed method in Figure 2. In the training phase (i.e., Figure 2(a)), we constrain the model to obtain a special biased representation vector from the variables x, which includes two independent components, i.e., a biased component r and an unbiased component z. The biased component r is responsible for the indirect effect, while the unbiased component z is responsible for the direct effect. Note that this special biased embedding vector, i.e., $[r, z]$, is easier to distinguish the influence of the confounding bias than the original representation z^*. In the test phase (i.e., Figure 2(b)), we discard the biased component and only use the unbiased component for more accurate recommendation.



(a) Training Phase



(b) Test Phase

Figure 2: (a) In the training phase, the model obtains a special biased embedding vector from the variables x, which includes a biased component r and an unbiased component z. (b) In the test phase, the biased component r is discarded, and the unbiased component z is used in the recommendation process.

From Figure 1(a), we can see that in order to meet our expectation on more accurate recommendation, we have to satisfy the following conditions: 1) to avoid the influence of the biased variables, the unbiased component z should not overfit the variables x; 2) due to the role of the direct effect, the unbiased component z needs to predict the label y as accurately as possible; 3) the biased component r and the unbiased component z must be as independent as possible to get a better distinction, i.e., $z \perp r$; and 4) due to the role of the indirect effect, the biased component r is also beneficial to predict the label y to a certain extent. Note that we have not constrained the relationship between the biased component r and the variables x, because the degree of dependence between them is determined by the strength of the bias in the feedback data, and blindly optimizing it may lead to bad results. Inspired by information theory, we can then derive the required objective function to be minimized from the above analysis,

$$\mathcal{L}_{DIB} := \min \underbrace{\beta I(z;x)}_{①} - \underbrace{I(z;y)}_{②} + \underbrace{\gamma I(z;r)}_{③} - \underbrace{\alpha I(r;y)}_{④}, \tag{3}$$

where term ① is a compression term that describes the mutual information between the variables x and the unbiased embedding z; term ② is an accuracy term that describes the performance of the unbiased embedding z; term ③ is a de-confounder penalty term, which describes the dependency degree between the biased embedding r and the unbiased embedding z; and term ④ is similar to term ②, which is used for the potential gain from the biased embedding r. Note that β, γ and α are the weight parameters. Since terms ① and ② in Eq.(3) are similar to a standard information bottleneck, we call the proposed method a *debiased information bottleneck*, or DIB for short. By optimizing $\mathcal{L}_{DIB}$, we expect to get the desired biased and unbiased components, and can then prune the confounding bias.

5 A TRACTABLE OPTIMIZATION FRAMEWORK

$\mathcal{L}_{DIB}$ is clearly an intractable optimization function, especially the key term $I(z;r)$ used to induce the desired embedding separation. Next, we discuss the optimization of the de-confounder penalty term $I(z;r)$ and the compression term $I(z;x)$, and derive an upper bound of Eq.(3). Finally, we describe a tractable solution for the objective function $\mathcal{L}_{DIB}$ according to the upper bound.

5.1 The De-confounder Penalty Term

Based on the chain rule of mutual information, we have the following equation about the de-confounder penalty term ③ in Eq.(3),

$$I(z;r) = I(z;y) - I(z;y|r) + I(z;r|y). \tag{4}$$

We further inspect the term $I(z;r|y)$ in Eq.(4). Since the distribution of z depends solely on the variables x, and x is affected by y, we have $H(z|y,r) = H(z|y)$, where $H(\cdot|\cdot)$ denotes the conditional entropy [15, 26]. Combining the properties of mutual information, we have,

$$I(z;r|y) = H(z|y) - H(z|y,r) = H(z|y) - H(z|y) = 0. \tag{5}$$

By substituting Eq.(5) into Eq.(4), we have,

$$I(z;r) = I(z;y) - I(z;y|r). \tag{6}$$

Since the term $I(z;y|r)$ in Eq.(6) is still difficult to be calculated, we use the form of conditional entropy to further simplify it,

$$I(z;r) = I(z;y) - H(y|r) + H(y|z,r). \tag{7}$$

Finally, combining Eq.(3) and Eq.(7), we can rewrite the objective function $\mathcal{L}_{DIB}$ in Eq.(3) as follows,

$$\begin{aligned}\mathcal{L}_{DIB} &= \beta I(z;x) - I(z;y) + \gamma I(z;r) - \alpha I(r;y)\\ &= \beta I(z;x) - I(z;y) + \gamma\left[I(z;y) - H(y|r) + H(y|z,r)\right] - \alpha I(r;y)\\ &= \beta I(z;x) - (1-\gamma) I(z;y) - \gamma H(y|r) + \gamma H(y|z,r) - \alpha I(r;y)\,.\end{aligned} \tag{8}$$

5.2 The Compression Term

We can find that only the compression term $I(z;x)$ is related to the variables x in Eq.(8). To optimize it directly, we describe a simple and precise expression of this mutual information using a method similar to that in [15, 40]. First, based on the relationship between mutual information and Kullback-Leibler (KL) divergence, the compression term $I(z;x)$ can be calculated as follows,

$$\begin{aligned}I(z;x) &= \mathbb{E}_x\left[D_{\mathrm{KL}}(p(z|x)\parallel p(z))\right]\\ &= \sum_x p(x)\sum_z p(z|x)\log p(z|x) - \sum_z p(z)\log p(z).\end{aligned} \tag{9}$$

However, the marginal probability $p(z) = \sum_x p(z|x)p(x)$ is usually difficult to be calculated in practice. We use variational approximation to address this issue, i.e., we use a variational distribution $q(z)$ instead of $p(z)$. According to Gibbs' inequality, we know that the KL divergence is non-negative. Therefore, we can derive an upper bound of Eq.(9),

$$\begin{aligned}&D_{\mathrm{KL}}(p(z)\parallel q(z)) \geq 0\\ &\Rightarrow -\sum_z p(z)\log p(z) \leq -\sum_z p(z)\log q(z)\\ &\Rightarrow D_{\mathrm{KL}}(p(z|x)\parallel p(z)) \leq D_{\mathrm{KL}}(p(z|x)\parallel q(z))\,.\end{aligned} \tag{10}$$

Similar to most previous works [21], we can assume that the posterior $p(z|x) = \mathcal{N}\left(\mu(x), \operatorname{diag}\left\{\sigma^2(x)\right\}\right)$ is a Gaussian distribution, where $\mu(x)$ is the encoded embedding of the variables x and $\operatorname{diag}\left\{\sigma^2(x)\right\}$ is the diagonal matrix indicating the variance. Through the reparameterization trick, the embedding z can be generated according to $z = \mu(x) + \epsilon \odot \sigma(x)$, where $\epsilon \sim \mathcal{N}(0, I)$. Obviously, if we fix $\sigma(x)$ to be an all-zero matrix, z will reduce to a deterministic embedding. On the other hand, the prior $q(z)$ is assumed to be a standard Gaussian variational distribution, i.e., $q(z) = \mathcal{N}(0, I)$. Finally, we can rewrite the above upper bound,

$$D_{\mathrm{KL}}(p(z|x)\parallel q(z)) = \frac{1}{2}\|\mu(x)\|_2^2 + \frac{1}{2}\sum_d\left(\sigma_d^2 - \log\sigma_d^2 - 1\right), \tag{11}$$

where σ_d^2 is an element in $\operatorname{diag}\left\{\sigma^2(x)\right\}$, i.e., $\operatorname{diag}\left\{\sigma^2(x)\right\} = \{\sigma_d^2\}_{d=1}^D$. This means that for a deterministic embedding z, we can optimize this upper bound by directly applying the ℓ_2-norm regularization on the embedding vector z, which is equivalent to optimizing the compression term $I(z;x)$. Note that the compression term in previous works act on the entire biased representation z^*, and we only compress the unbiased component z of the representation.

5.3 Algorithm

For the mutual information $I(z;y)$ in Eq.(8), we have $I(z;y) = H(y) - H(y|z)$. Since $H(y)$ is a positive constant and can be ignored, we have the following inequality,

$$I(z;y) \geq -H(y|z)\,. \tag{12}$$

This inequality also applies to the mutual information $I(r;y)$ in Eq.(8). Combining Eq.(11) and Eq.(12), we can rewrite Eq.(8) as follows,

$$\begin{aligned}\mathcal{L}_{DIB} &= \beta I(z;x) - (1-\gamma) I(z;y) - \gamma H(y|r) + \gamma H(y|z,r) - \alpha I(r;y)\\ &\leq \beta\|\mu(x)\|_2^2 + (1-\gamma)H(y|z) - (\gamma-\alpha)H(y|r) + \gamma H(y|z,r).\end{aligned} \tag{13}$$

Finally, we get a tractable solution for $\mathcal{L}_{DIB}$,

$$\hat{\mathcal{L}}_{DIB} = \underbrace{(1-\gamma)H(y|z)}_{(a)} - \underbrace{(\gamma-\alpha)H(y|r)}_{(b)} + \underbrace{\gamma H(y|z,r)}_{(c)} + \underbrace{\beta\|\mu(x)\|_2^2}_{(d)}, \tag{14}$$

where $0 < \alpha < \gamma < 1$. Let $\hat{y}_r$, $\hat{y}_z$, and $\hat{y}_{z,c}$ be the predicted labels generated by the biased component r, the unbiased component z, and the biased embedding vector $[z, r]$ as shown in Figure 2(a), respectively, the final objective function also contains four terms: term (a) denotes the cross entropy between $\hat{y}_z$ and y; term (b) denotes the cross entropy between $\hat{y}_r$ and y; term (c) denotes the cross entropy between $\hat{y}_{z,r}$ and y; and term (d) is the regularization term to improve the robustness of the embedding representation. A complete optimization process of DIB is shown in Algorithm 1.

Algorithm 1 Debiased Information Bottleneck (DIB)

```
Input: Observed feedback events {(x, y)}; hyper-parameters α, β, γ;
1: Initialize the parameters of the model θ (including the biased embedding representation r and the unbiased embedding representation z);
2: repeat
3:     Calculate term (c) in Eq.(14) by the concatenation of z and r, i.e., H(y|z, r);
4:     Calculate term (a) in Eq.(14) by z, i.e., H(y|z) (to keep the dimensions consistent, z is concatenated with a zero vector with the same dimension of r);
5:     Calculate term (b) in Eq.(14) by r, i.e., H(y|r) (similar to the above operation);
6:     Calculate term (d) in Eq.(14) by applying regularization to z;
7:     Update the model parameters θ via stochastic gradient descent based on $\hat{\mathcal{L}}_{DIB}$ (i.e., Eq.(14));
8: until training loss stops to decrease.
```

6 EXPERIMENTS

In this section, we conduct comprehensive experiments with the aim of answering the following three key questions.

- RQ1: How does the proposed method perform against the baselines in an unbiased evaluation?
- RQ2: What is the role of each term in the proposed method (i.e., the ablation studies of our DIB)?
- RQ3: What are the characteristics of the proposed method in the training process?

6.1 Experiment Setup

6.1.1 Datasets. In order to evaluate the performance of the model in mitigating the confounding bias, we need a test set that includes

Table 1: Statistics of the datasets. P/N represents the ratio between the numbers of positive and negative feedback.

	Yahoo! R3		Product	
	#Feedback	P/N	#Feedback	P/N
training	254,713	67.02%	4,160,414	2.21%
validation	56,991	67.00%	897,449	2.15%
test	54,000	9.64%	225,762	1.03%

unbiased feedback events to simulate the ideal distribution. We consider both a public dataset and a real-world product dataset in the experiments, where the statistics are described in Table 1.

- **Yahoo! R3**: This dataset contains ratings collected from two different sources on Yahoo! Music services, containing 15,400 users and 1000 songs. The user set consists of ratings provided by users' subjective choices and is therefore considered to be biased. The random set consists of ratings collected during an online survey, i.e., each of the first 5400 users is required to provide ratings on ten randomly displayed songs, which can be considered unbiased. We binarize each rating by checking whether it is larger than 3 and then obtain some positive feedback and negative feedback. Note that the missing entries are treated as negative feedback. For the user set, we randomly split it into two subsets from the user level: 80% of each user's feedback for training and the remaining 20% for validation to tune the hyper-parameters. The random set is used as the test set for evaluating our method.
- **Product**: This is a large-scale dataset for CTR prediction, which includes two weeks of user click records from a real-world advertising system. This data covers 122 displayed advertisements and approximately 300,000 users. Similarly, it contains a user set and a random set. The user set is recorded when the system is using several traditional ranking-oriented recommendation policies, and the random set is recorded using a uniform policy. Next, we randomly split the user set into two subsets in the same way as that for Yahoo! R3, i.e., 80% feedback is used as the training set, and the rest is used as the validation set. The entire random set is used as the test set.

6.1.2 Evaluation Metrics. For all experiments, we employ four evaluation metrics that are widely used in recommender systems, including the area under the ROC curve (AUC), precision (P@K), recall (R@K) and normalized discounted cumulative gain (nDCG). We set the length of the recommendation list as 50, and choose AUC as our main evaluation metric because it is one of the most important metrics in the industry and previous works on debiasing. Due to space limitation, we report the results of P@K and R@K when K is 5 and 10, and the results of nDCG when K is 50.

6.1.3 Baselines. Since most debiasing methods are integrated into some existing recommendation models, in order to evaluate the generalization of our method, we adopt two of the most common recommendation models as the backbones, i.e., matrix factorization (MF) [17] and neural collaborative filtering (NCF) [14]. In addition, among the four lines of debiasing methods summarized in Sec. 2.1, the heuristic-based methods are usually inefficient, and the unbiased data augmentation methods are not suitable for the scenario where only a biased data is available for training in this paper. Therefore, we choose the representative methods in the other two lines as the baselines. Note that based on different backbones, each of these baselines has two versions.

- **MF [17]:** This is one of the most classic recommendation models. The user-item interaction matrix is factorized to learn the representation of the users and items, and their inner products are used to predict the preferences.
- **NCF [14]:** This is one of the most classic recommendation models combined with neural networks. Compared with the linearity of matrix factorization, neural networks are used to model the nonlinear relationship between users and items to further improve the recommendation performance.
- **IPS [32]:** This is a representative method based on the inverse propensity score (IPS) in debiasing recommendation. The estimated propensity score is used as the weight of the training feedback to adjust the biased distribution of the feedback. We estimate the propensity score of item i for any user via the relative item popularity,

$$P_{*,i} = \left(\frac{\sum_{u \in \mathcal{U}} O_{u,i}}{max_{i \in \mathcal{I}} \sum_{u \in \mathcal{U}} O_{u,i}} \right)^{\eta}, \tag{15}$$

  where $O_{u,i}$ indicates the observation status of user u for item i, i.e., $O_{u,i} = 1$ means that the user u observes the item i, while $O_{u,i} = 0$ is the opposite, and η is a control weight used to adapt to different datasets. Similar methods have also been adopted in previous works [19, 31].
- **SNIPS [34]:** The above method based on the inverse propensity score has the problem of high variance. To solve this problem, a self-normalized inverse propensity score is proposed and can be written as,

$$\mathcal{L}_{SNIPS} = \frac{\sum_{(u,i):O_{u,i}=1} \frac{\mathcal{L}\left(\hat{Y}_{u,i}, Y_{u,i}\right)}{P_{u,i}}}{\sum_{(u,i):O_{u,i}=1} \frac{1}{P_{u,i}}}, \tag{16}$$

  where $P_{u,i}$ is calculated according to Eq.(15), and $\mathcal{L}\left(\hat{Y}_{u,i}, Y_{u,i}\right)$ is the cross-entropy loss of the observed feedback. $Y_{u,i}$ and $\hat{Y}_{u,i}$ are the true and predicted label of user u for item i, respectively.
- **AT [30]:** For the selection bias in recommender systems, an upper bound of the generalization error is derived and solved by an asymmetric tri-training method. Two pre-models are used to generate a pseudo-label set, and the target model and the two pre-models are then retrained on this set. This process is repeated until the debiasing performance of the target model is improved the most.
- **Rel [31]:** Based on the idea of positive unlabeled learning, an unbiased estimation is derived for binary feedback modeling in recommender systems. The inverse propensity score is estimated and weighted differently for positive feedback and negative feedback. The objective function can be represented as follows,

$$\hat{\mathcal{L}}_{Rel} = \frac{1}{|\mathcal{D}|} \sum_{(u,i) \in \mathcal{D}} \left[\frac{Y_{u,i}}{P_{u,i}} \delta_{u,i}^{(1)} + (1 - \frac{Y_{u,i}}{P_{u,i}}) \delta_{u,i}^{(0)} \right], \tag{17}$$

Table 2: Hyper-parameters and their values tuned in the experiments.

Name	Range	Functionality
$rank$	$\{5, 10, \cdots, 195, 200\}$	Embedded dimension
β	$\{1e^{-5}, 1e^{-4}, \cdots, 1e^{-1}, 1\}$	Regularization
bs	$\{2^7, 2^8, \cdots, 2^{13}, 2^{14}\}$	Batch size
lr	$\{1e^{-4}, 5e^{-4} \cdots 5e^{-2}, 1e^{-1}\}$	Learning rate
γ	$[0.1, 0.2]$	Loss weighting
α	$[0.001, 0.1]$	Loss weighting

where $\delta_{u,i}^{(1)}$ and $\delta_{u,i}^{(0)}$ denote the positive loss and the negative loss, respectively, and $P_{u,i}$ is calculated by Eq.(15).

- **CVIB [40]:** It is an alternative framework for debiasing learning without an unbiased feedback data, which is called the information-theoretic counterfactual variational information bottleneck. By separating the task-aware mutual information term in standard information bottleneck into a factual part and a counterfactual part, it derives a contrastive information regularizer and an additional output confidence penalty to improve the learning balance between the observed and unobserved data.

6.1.4 Implementation Details. We implement all the methods on TensorFlow with the Adam optimizer[1]. By evaluating the performance on AUC on the validation set, we use the hyper-parameter search library *Optuna* [4] to accelerate the tuning process of all the methods. The range of the values of the hyper-parameters are shown in Table 2. Note that the source codes, parameter search records and results are available at https://github.com/dgliu/RecSys21_DIB.

6.2 RQ1: Comparison Results of Unbiased Evaluation

The comparison results are reported in Table 3 and Table 4. For the results of using matrix factorization as the backbone model shown in Table 3, the proposed method consistently outperforms all the baselines on all the metrics across the two datasets of Yahoo! R3 and Product. In addition, we have the following main observations: 1) The debiasing method is usually better than the basic baseline, i.e., matrix factorization, except for IPS-MF on Yahoo! R3. This may be due to the inaccurate estimation of the propensity score when the data size is small. 2) By avoiding the estimation of the propensity score and providing theoretical completeness, theoretical tools-based methods (i.e., CVIB, AT, and Rel) usually achieve better and more stable performance. 3) For the baseline CVIB-MF whose objective function is also induced by information theory, its improvement over MF on Yahoo! R3 is small, while the proposed method is better and more robust on both datasets.

For the results of using neural collaborative filtering as the backbone shown in Table 4, the proposed method outperforms all the baselines in most cases. Specifically, we have the following observations: 1) when K takes a relatively small value on Product, the proposed method is slightly worse than IPS-NCF. However, when K takes a larger value, the proposed method has obvious performance advantages. This means that the proposed method still has better overall performance. 2) Even with the use of the techniques to reduce variance, the incorrect estimation of the propensity score in a small dataset will be further exacerbated. 3) AT and Rel may not be well adapted to neural network-based models due to their dependence on certain types of loss functions.

[1]https://www.tensorflow.org

6.3 RQ2: Ablation Studies

To understand the respective effects of the last three terms in Eq.(14) on model training, we conduct ablation studies on Yahoo! R3 and Product. The results are shown in Table 5. In the first half of Table 5, we show the results using matrix factorization as the backbone. For the results on Yahoo! R3, we can see that removing any term will hurt the performance in most cases, and removing term (d) leads to the worst performance. This may be due to the small size of Yahoo! R3, which causes the term (d) to play a more important role in avoiding overfitting of the model to the training set. Conversely, we can find that the effect of removing the term (d) on the larger Product dataset is small.

We show the results using neural collaborative filtering as the backbone model in the second half of Table 5. We note that removing the term (d) will bring the greatest performance degradation on different data scales. One possible reason is that the neural network-based model introduces more parameters, and these parameters need to be well learned under the constraints of regularization terms. There are some unexpected cases in Table 5, i.e., when K takes a small value, the full version of the proposed method has a slight disadvantage on a few metrics. This may be due to the noise caused by only considering AUC in parameter tuning. In general, all terms in the proposed method can synergistically produce the greatest gain.

6.4 RQ3: Model Analysis

In this section, we conduct some more in-depth analysis of the proposed method in order to have a better understanding of the properties of the proposed method.

6.4.1 Visualization of the biased component r and the unbiased component z. A key question is whether the biased component r and the unbiased component z in the proposed method have a pattern that meets the expectation, i.e., they gradually become independent in training. To answer this question, at several training time points, we use the *t-SNE* [36] method to visualize the biased and unbiased components. The results are shown in Figure 3. We can see that as the training progresses, there is a clear separation between the biased component in blue and the unbiased component in red. This verifies the effectiveness of the proposed method. We also note that the unbiased component is looser than the biased component, and a small part of the unbiased component is still mixed with the biased component. This may be because the unbiased component needs to learn the personalized differences between the users, among whom there are still some users who are difficult to be distinguished.

6.4.2 Visualization of the loss of each term in $\mathcal{L}_{DIB}$. We next analyze the trend of the loss of each term in $\mathcal{L}_{DIB}$. The results are

Table 3: Comparison results of unbiased evaluation using MF as the backbone model, where the best results and the second best results are marked in bold and underlined, respectively. AUC is the main evaluation metric.

	Yahoo! R3						Product					
Method	**AUC**	**nDCG**	**P@5**	**P@10**	**R@5**	**R@10**	**AUC**	**nDCG**	**P@5**	**P@10**	**R@5**	**R@10**
MF	0.7081	0.0341	0.0043	0.0043	0.0123	0.0273	0.6936	0.0324	0.0085	0.0079	0.0407	0.0752
IPS-MF	0.7040	0.0259	0.0031	0.0033	0.0091	0.0182	0.7125	0.0408	0.0095	0.0105	0.0456	0.1019
SNIPS-MF	0.7124	0.0390	0.0057	0.0048	0.0182	0.0293	0.7098	0.0403	0.0092	0.0104	0.0438	0.1003
CVIB-MF	0.7086	0.0488	0.0080	0.0075	0.0253	0.0471	0.7143	0.0521	0.0106	0.0109	0.0520	0.1076
AT-MF	0.7290	0.0676	0.0113	0.0101	0.0345	0.0629	0.7191	0.0443	0.0110	0.0113	0.0532	0.1092
Rel-MF	0.7469	0.0843	0.0159	0.0131	0.0526	0.0863	0.6709	0.0656	0.0157	0.0142	0.0776	0.1386
DIB-MF	**0.7602**	**0.0932**	**0.0177**	**0.0151**	**0.0566**	**0.0960**	**0.7365**	**0.0819**	**0.0198**	**0.0173**	**0.0965**	**0.1688**

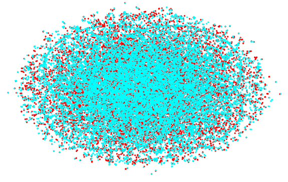
Table 4: Comparison results of unbiased evaluation using NCF as the backbone model, where the best results and the second best results are marked in bold and underlined, respectively. AUC is the main evaluation metric.

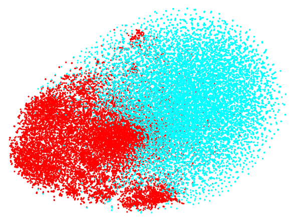
	Yahoo! R3						Product					
Method	**AUC**	**nDCG**	**P@5**	**P@10**	**R@5**	**R@10**	**AUC**	**nDCG**	**P@5**	**P@10**	**R@5**	**R@10**
NCF	0.7251	0.0350	0.0037	0.0036	0.0097	0.0203	0.7211	0.1465	0.0174	0.0136	0.0865	0.1343
IPS-NCF	0.7221	0.0322	0.0032	0.0039	0.0085	0.0219	0.7284	0.1463	**0.0176**	0.0135	**0.0870**	0.1330
SNIPS-NCF	0.7230	0.0310	0.0030	0.0031	0.0087	0.0175	0.7257	0.1454	0.0173	0.0135	0.0856	0.1323
CVIB-NCF	0.7265	0.0347	0.0036	0.0030	0.0105	0.0177	0.7291	0.1440	0.0149	0.0137	0.0732	0.1350
AT-NCF	0.7139	0.0333	0.0031	0.0033	0.0084	0.0179	0.6814	0.1464	0.0156	0.0139	0.0774	0.1375
Rel-NCF	0.6867	0.0507	0.0071	0.0064	0.0233	0.0408	0.6653	0.1404	0.0156	0.0145	0.0763	0.1423
DIB-NCF	**0.7553**	**0.0686**	**0.0108**	**0.0101**	**0.0339**	**0.0630**	**0.7345**	**0.1483**	0.0175	**0.0163**	0.0865	**0.1613**

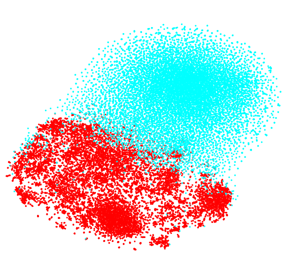
Table 5: Results of the ablation studies using MF and NCF as the backbone models, where the best results and the second best results are marked in bold and underlined, respectively. AUC is the main evaluation metric.

	Yahoo! R3						Product					
Method	AUC	nDCG	P@5	P@10	R@5	R@10	AUC	nDCG	P@5	P@10	R@5	R@10
DIB-MF	**0.7602**	**0.0932**	**0.0177**	**0.0151**	0.0566	**0.0960**	**0.7365**	**0.0819**	**0.0198**	**0.0173**	**0.0965**	**0.1688**
w/o term (b)	0.7505	0.0893	0.0175	0.0138	0.0563	0.0909	0.7173	0.0546	0.0126	0.0115	0.0614	0.1112
w/o term (c)	0.7545	0.0915	0.0173	0.0142	**0.0569**	0.0937	0.7156	0.0511	0.0109	0.0113	0.0527	0.1083
w/o term (d)	0.7342	0.0769	0.0144	0.0117	0.0478	0.0737	0.6809	0.0719	0.0178	0.0153	0.0867	0.1504
DIB-NCF	**0.7553**	**0.0686**	**0.0108**	**0.0101**	**0.0339**	**0.0630**	**0.7345**	**0.1483**	0.0175	**0.0163**	0.0865	**0.1613**
w/o term (b)	0.7326	0.0553	0.0089	0.0081	0.0271	0.0489	0.7274	0.1474	0.0181	0.0131	0.0901	0.1294
w/o term (c)	0.7373	0.0592	0.0097	0.0093	0.0292	0.0585	0.7276	0.1474	**0.0181**	0.0131	**0.0901**	0.1294
w/o term (d)	0.7243	0.0597	0.0102	0.0099	0.0318	0.0603	0.7133	0.1438	0.0177	0.0123	0.0876	0.1214

shown in Figure 4. A sufficient reduction in the loss of term (a) guarantees the basic performance of the proposed method (corresponding to the unbiased component z). By comparing the trends of term (b) and term (c), we can see that using the biased component r alone will lead to a very poor result, but as long as it is combined with the unbiased component z, the best result can be achieved. This is reasonable, because the biased component itself does not reflect the user's preferences well, but the combination with the unbiased component will lead to an over-approximation effect on the observed feedback. In addition, the trend of term (d)


(a) Initialization


(b) Epoch=100


(c) Epoch=200

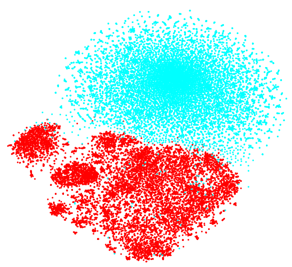

(d) Convergence

Figure 3: Dynamic visualization of the unbiased component z and the biased component r as training progresses, in which the blue and red points are used to denote the biased component and the debiased component, respectively.

indicates that the compression term is helpful in learning a more ideal unbiased component throughout the training process.

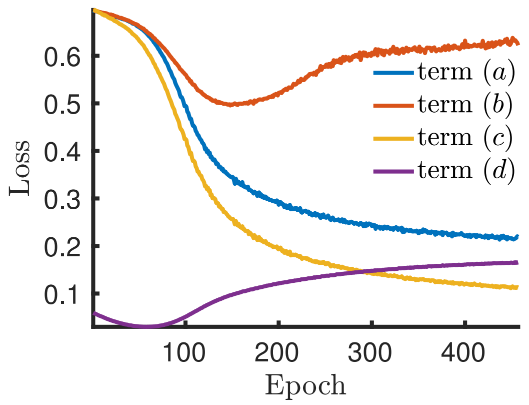


Figure 4: The trend of the loss of each term in $\mathcal{L}_{DIB}$ as training progresses.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we describe the generation process of the biased and unbiased feedback in recommender systems via two respective causal diagrams, and then define a new bias based on the difference between them, which is called confounding bias. When only the biased feedback is available, we analyze the conditions that need to be met to alleviate the confounding bias, and propose a debiased information bottleneck (DIB) method to perform this optimization process based on the guidance of information theory. Moreover, we also derive a tractable solution for the proposed method. We verify the effectiveness of the proposed method on a public dataset and a real product dataset. In addition, we also include some ablation studies and deep analysis of the proposed method.

For future works, we plan to extend the proposed method to scenarios where more than one biased data is available. We are also interested in further relaxing the independent assumptions of the unbiased and biased components, that is, there may be a special mixed component entangled with the biased or unbiased component in some tasks.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their expert and constructive comments and suggestions, and the support of National Natural Science Foundation of China Nos. 61836005 and 61872249.

REFERENCES

[1] 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*. 42–46.
[2] Alessandro Achille and Stefano Soatto. 2018. Information dropout: Learning optimal representations through noisy computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 12 (2018), 2897–2905.
[3] 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*. 474–482.
[4] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2623–2631.
[5] Stephen Bonner and Flavian Vasile. 2018. Causal embeddings for recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 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*. 415–424.
[7] Pengyu Cheng, Martin Renqiang Min, Dinghan Shen, Christopher Malon, Yizhe Zhang, Yitong Li, and Lawrence Carin. 2020. Improving disentangled text representation learning with information-theoretic guidance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7530–7541.
[8] Bin Dai, Chen Zhu, Baining Guo, and David Wipf. 2018. Compressing neural networks using the variational information bottleneck. In *Proceedings of the 35th International Conference on International Conference on Machine Learning*. 1135–1144.
[9] Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. 2020. Learning robust representations via multi-view information bottleneck. In *Proceedings of the 8th International Conference on Learning Representations*.
[10] Ruoyuan Gao and Chirag Shah. 2020. Counteracting bias and increasing fairness in search and recommender systems. In *Proceedings of the 14th ACM Conference on Recommender Systems*. 745–747.
[11] Yingqiang Ge, Shuya Zhao, Honglu Zhou, Changhua Pei, Fei Sun, Wenwu Ou, and Yongfeng Zhang. 2020. Understanding echo chambers in e-commerce recommender systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2261–2270.
[12] 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*. 326–335.
[13] Negar Hassanpour and Russell Greiner. 2020. Learning disentangled representations for counterfactual regression. In *Proceedings of the 8th International Conference on Learning Representations*.
[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 27th International Conference on World Wide Web*. 173–182.

[15] Ayush Jaiswal, Rob Brekelmans, Daniel Moyer, Greg Ver Steeg, Wael AbdAlmageed, and Premkumar Natarajan. 2019. Discovery and separation of features for invariant representation learning. *arXiv preprint arXiv:1912.00646* (2019).

[16] Ray Jiang, Silvia Chiappa, Tor Lattimore, András György, and Pushmeet Kohli. 2019. Degenerate feedback loops in recommender systems. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. 383–390.

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

[18] Kun Kuang, Peng Cui, Bo Li, Meng Jiang, Shiqiang Yang, and Fei Wang. 2017. Treatment effect estimation with data-driven variable decomposition. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. 140–146.

[19] Dawen Liang, Laurent Charlin, and David M Blei. 2016. Causal inference for recommendation. In *Workshop on Causation: Foundation to Application co-located with the 32nd Conference on Uncertainty in Artificial Intelligence*.

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

[21] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational autoencoders for collaborative filtering. In *Proceedings of the 27th International Conference on World Wide Web*. 689–698.

[22] Chen Lin, Dugang Liu, Yanghua Xiao, and Hanghang Tong. 2020. Spiral of silence and its application in recommender systems. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[23] 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*. 831–840.

[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*. 222–230.

[25] 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*. 5–12.

[26] Daniel Moyer, Shuyang Gao, Rob Brekelmans, Greg Ver Steeg, and Aram Galstyan. 2018. Invariant representations without adversarial training. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 9102–9111.

[27] 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 29th International Conference on World Wide Web*. 1863–1873.

[28] Sonali Parbhoo, Mario Wieser, and Volker Roth. 2018. Causal deep information bottleneck. *arXiv preprint arXiv:1807.02326* (2018).

[29] Paul R Rosenbaum and Donald B Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (1983), 41–55.

[30] 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*. 309–318.

[31] 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 ACM International Conference on Web Search and Data Mining*. 501–509.

[32] 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*. 1670–1679.

[33] Yixin Su, Rui Zhang, Sarah Erfani, and Zhenghua Xu. 2021. Detecting beneficial feature interactions for recommender systems. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*. 4357–4365.

[34] Adith Swaminathan and Thorsten Joachims. 2015. The self-normalized estimator for counterfactual learning. In *Proceedings of the 29th International Conference on Neural Information Processing Systems*. 3231–3239.

[35] Naftali Tishby, Fernando C Pereira, and William Bialek. 1999. The information bottleneck method. In *Proceedings of the 37th Annual Allerton Conference on Communications, Control and Computing*. 368–377.

[36] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 11 (2008), 2579–2605.

[37] Wenjie Wang, Fuli Feng, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2020. "Click" is not equal to "like": Counterfactual recommendation for mitigating clickbait issue. *arXiv preprint arXiv:2009.09945* (2020).

[38] 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*. 610–618.

[39] 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*. 427–435.

[40] 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. *arXiv preprint arXiv:2009.02623* (2020).

[41] Tianxin Wei, Fuli Feng, Jiawei Chen, Chufeng Shi, Ziwei Wu, Jinfeng Yi, and Xiangnan He. 2020. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. *arXiv preprint arXiv:2010.15363* (2020).

[42] Shuyuan Xu, Yingqiang Ge, Yunqi Li, Zuohui Fu, Xu Chen, and Yongfeng Zhang. 2021. Causal collaborative filtering. *arXiv preprint arXiv:2102.01868* (2021).

[43] 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.

[44] 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*. 1929–1932.

[45] 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*. 329–338.

[46] Shuxi Zeng, Murat Ali Bayir, Joel Pfeiffer, 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*. 211–219.