



Debiased Recommendation with User Feature Balancing

MENGYUE YANG, University College London, United Kingdom

GUOHAO CAI, Noah's Ark Lab, Huawei, China

FURUI LIU, Zhejiang Laboratory, China

JIARUI JIN, Shanghai Jiao Tong University, China

ZHENHUA DONG, XIUQIANG HE, and JIANYE HAO, Noah's Ark Lab, Huawei, China

WEIQI SHAO, Renmin University of China, China

JUN WANG, University College London, United Kingdom

XU CHEN*, Beijing Key Laboratory of Big Data Management and Analysis Methods, Gaoling School of Artificial Intelligence, Renmin University of China, China

Debiased recommendation has recently attracted increasing attention from both industry and academic communities. Traditional models mostly rely on the inverse propensity score (IPS), which can be hard to estimate and may suffer from the high variance issue. To alleviate these problems, in this article, we propose a novel debiased recommendation framework based on user feature balancing. The general idea is to introduce a projection function to adjust user feature distributions, such that the ideal unbiased learning objective can be upper bounded by a solvable objective purely based on the offline dataset. In the upper bound, the projected user distributions are expected to be equal given different items. From the causal inference perspective, this requirement aims to remove the causal relation from the user to the item, which enables us to achieve unbiased recommendation, bypassing the computation of IPS. To efficiently balance the user distributions upon each item pair, we propose three strategies, including clipping, sampling, and adversarial learning to improve the training process. For more robust optimization, we deploy an explicit model to capture the potential latent confounders in recommendation systems. To the best of our knowledge, this article is the first work on debiased recommendation based on confounder balancing. In the experiments, we compare our framework with many state-of-the-art methods based on synthetic, semi-synthetic, and real-world datasets. Extensive experiments demonstrate that our model is effective in promoting the recommendation performance.

*Xu Chen is the corresponding author.

This work is supported in part by National Natural Science Foundation of China (No. 62102420), Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China, and Public Computing Cloud, Renmin University of China. The work is sponsored by Huawei Innovation Research Programs.

Authors' addresses: M. Yang and J. Wang, University College London, London, UK; emails: mengyue.yang.20@ucl.ac.uk, j.wang@cs.ucl.ac.uk; G. Cai, Z. Dong, X. He, and J. Hao, Noah's Ark Lab, Huawei, Longgang District, Shenzhen, China; emails: {caiguohao1, dongzhenhua, hexiuqiang1, haojianye}@huawei.com; F. Liu, Zhejiang Laboratory, Hangzhou, China; email: liufurui@zhejianglab.com; J. Jin, Shanghai Jiao Tong University, Minhang District, Shanghai, China; email: jinjiarui97@sjtu.edu.cn; W. Shao, Zhejiang Laboratory, Hangzhou, China; Renmin University of China, Haidian District, Beijing, China; email: shaoweiqi@ruc.edu.cn; X. Chen, Beijing Key Laboratory of Big Data Management and Analysis Methods, Gaoling School of Artificial Intelligence, Haidian District, Beijing, China; email: successcx@gmail.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.

1046-8188/2023/04-ART114 \$15.00

<https://doi.org/10.1145/3580594>

CCS Concepts: • **Information systems** → **Data mining**;

Additional Key Words and Phrases: Recommendation system, Debias Representation

ACM Reference format:

Mengyue Yang, Guohao Cai, Furui Liu, Jiarui Jin, Zhenhua Dong, Xiuqiang He, Jianye Hao, Weiqi Shao, Jun Wang, and Xu Chen. 2023. Debaised Recommendation with User Feature Balancing. *ACM Trans. Inf. Syst.* 41, 4, Article 114 (April 2023), 25 pages.
<https://doi.org/10.1145/3580594>

1 INTRODUCTION

Recommender system, as an effective remedy for information overloading, is playing a key role in the modern E-commerce. Traditional recommender models are directly learned based on the observed user behaviors, which may have been contaminated by the exposure or selection bias [5, 29, 41, 43]. Typically, the users in a recommender system can only access and evaluate a small part of the whole items due to the effect of the former recommendation algorithm or the user intrinsic tendency. This means that the observed data is a skewed version of the real user preference, which may bias the recommender models and limit the performance.

For solving this problem, the most reliable method is conducting online random experiments. However, to collect enough training samples, such random policy has to occupy the business traffic for a long time, which can be detrimental to the user experience. As a result, recent years have witnessed many offline debiasing methods [3, 29, 36, 42, 46, 47], among which the **inverse propensity score (IPS)** is one of the most popular strategies. The basic idea of IPS is to adjust the offline data distribution to be aligned with the ideal learning objective by re-weighting the training samples. While IPS-based models have been widely leveraged for debaised recommendation, there are some significant weaknesses: To begin with, different from the classical IPS models [18, 33, 34], which are mostly evaluated based on small scale or simulated datasets, in the recommendation domain, the sample weights are estimated from the noisy and extremely sparse datasets, which can be highly unreliable. The error of the weights will be propagated to lower the final recommendation performance. In addition, the IPS-based learning objective usually suffers from the high variance issue [23, 32–34], which makes the learned model quite fragile in facing with the complex recommendation environments.

To avoid these problems, in this article, we propose a novel debaised recommendation framework based on confounder balancing. In specific, we formulate the recommendation task by a causal inference tool called **potential outcome framework (POF)**. Ideally, for learning user preferences on different items in an unbiased manner, the dataset used for training the recommender model should be generated based on the causal graph presented in Figure 1(b), where the items are uniformly exposed for all the users.¹ However, in practice, the offline dataset is generated by the causal graph shown in Figure 1(c), where the user and item are linked because of the personalized recommender systems or the active user self-selection tendencies. To learn an unbiased model only with the offline dataset, we introduce a projection function to adjust the user feature distributions, aiming to derive a solvable upper bound of the ideal unbiased objective. In this upper bound, the user feature distributions are expected to be balanced given different items. This requirement actually aims to cut down the causal relation between the user and item for achieving debaised recommendation, where we do not need to compute IPS in the whole modeling process.

While this seems to be a promising idea, it is non-trivial due to the following challenges: To begin with, while there are many studies on causal inference based on confounder balancing [16,

¹It should be noted that here the causal graph is leveraged to describe the data generation process, while in the testing phase, the items are also recommended according to the user preference.

[17, 19, 20, 30], they mostly focus on treatment-effect estimation, where the loss function is limited to the **mean square error (MSE)** and the treatment is assumed to be binary. However, in the recommendation problem, the user preference label is usually categorical implicit feedback, which should be better optimized with the cross-entropy loss. Directly replacing the loss function in previous work is not easy, since they are highly tailored for MSE. More principled methods are needed to derive the learning objective suitable for the recommendation task. In addition, the large number of items in recommender systems makes the binary treatment assumption impractical. More effective and efficient objectives are needed to handle the large treatment space. At last, there can be many unobserved factors (or called latent confounders) that simultaneously influence the items and user feedback; for example, the sales promotion or user emotions. Such factors make it hard to cut down the user-item causal relation only by adjusting user feature distributions. How to model them is still unexplored in the recommendation domain.

For solving these challenges, we derive a general upper bound of the ideal learning objective based on Jensen's gap bound [10], which is compatible with any loss function. In this upper bound, the user distributions are expected to be equal given different items. Since the number of items can be very large in practice, we design three strategies, including clipping, sampling, and adversarial learning to facilitate model optimization. To handle the latent confounders, we deploy a neural model to infer them, which are then integrated into our framework to make more robust user feedback estimation. Our main contributions can be summarized as follows:

- We propose to build debiased recommender models based on confounder balancing, which, to the best of our knowledge, is the first time in the recommendation domain.
- To achieve our idea, we derive a surrogate learning objective, which can be optimized based on the offline dataset. To make this objective more effective, we propose three strategies to handle the potential large number of items and deploy an explicit model to capture the latent confounders.
- We conduct extensive experiments based on different types of datasets to demonstrate the effectiveness of our framework.

2 PRELIMINARIES

2.1 Selection Debias in Recommendation System

Recent years have witnessed a surge of research efforts on debiased recommendation system. Bias problems have multiple definition in terms of different aspect across different papers, such as "selection bias" [22], "position bias" [37], "popularity bias" [47] and so on. This article focuses on the "selection bias." A rigorous definition of "selection bias" is given by Chen et al. [3], which describes the user rating of items is missing not at random. The phenomenon is caused by non-uniform historical recommendation policy; user ratings only concentrate on a few recommended items that are not selected randomly from item sets. Recommendation feedback loop in online real-world recommendation system will reinforce the bias, because the recommendation policy is always trained by the observational data that contains bias, and the recommendation policy will be more likely to recommend the item that was selected by users in previous recommendation process. Recent works consider the recommendation process as an extension of treatment effect task that the treatment on patients and recommended item are both regarded as interventions. In these two tasks, most of the effect of different treatment on patients (users) is unknown. The debias problem focuses on how to evaluate the unbiased potential outcome on missing value. Existing methods provide three approaches to deal with these problem. The first branch is based on propensity score method [29]. Propensity score is defined as the marginal probability of observing a rating value for user-item pairs. Inverse propensity score is a kind of re-weighting-based method

to achieve debias. The next approach is direct method, which utilizes an imputation model to fill out the missed rating records [18]. Based on the imputation model, the target method is evaluated on full recommendation records including factual and counterfactual data. The third approach, called Doubly Robust method [8], integrates the advantages of both Inverse Propensity Score method and Direct method. The aforementioned three approaches help alleviate the selection bias problem from the aspect of potential outcome framework.

2.2 Potential Outcome Framework

Potential outcome framework (POF) is an effective tool for causal estimation and discovery under both experimental and observational settings. In POF, there are three basic elements:

- **Unit (U)** is the atom research objective, e.g., a user in the recommender system.
- **Treatment (I)** is the action imposed on the unit, e.g., an item recommended to the user.
- **Outcome (Y)** is the result after applying the treatment to the unit, e.g., the user feedback on the item.

The relations between different variables in POF are described by a **causal graph**. For example, the recommendation task can be formulated by the causal graph in Figure 1(a), where the user is a common cause of the item and user feedback, and the user feedback is jointly determined by the user and item. The variables simultaneously influence the treatment, and outcome is called **confounder**. In Figure 1(a), the user is a confounder of the item and user feedback. Specially, if $p(U = u|I = i) = p(U = u|I = i')$ holds for any treatment pair (i, i') , then U is independent with I , that is, there is no causal relation between U and I . At this time, we say the confounder is balanced on the treatment. In the following, we indiscriminately use the notations of confounder balancing and user feature balancing. Since POF is not the focus of this article, we refer the readers to Reference [45] for more details.

2.3 Debaised Recommendation

Debaised recommendation is a recently emerged research topic. Here, we explain it from the perspective of potential outcome framework. As mentioned above, the recommendation task can be formulated by the causal graph in Figure 1(a). Formally, let u , i , and y denote the user, item, and user feedback on the item. Suppose δ is a loss function, f^* is a recommender model, then the ideal learning objective for debaised recommendation is:

$$L_{\text{ideal}} = \mathbb{E}_u \left[\frac{1}{N} \sum_{i=1}^N \delta(r_{ui}, f^*(u, i)) \right] = \mathbb{E}_u [\mathbb{E}_{i \sim p^{\text{uni}}(i)} [\delta(r_{ui}, f^*(u, i))]], \quad (1)$$

where N is the number of items. $r_{ui} = \mathbb{E}[y|u, i]$ is the expectation of the user feedback on the item. $p^{\text{uni}}(i) = \frac{1}{N}$ is the uniform distribution supported by the whole item set. We regard the user feedback as a random variable, which can be more practical considering the potential noisy information in recommender systems.

In this objective, the training data is assumed to be generated by a causal graph (see Figure 1(b)), where there is no edge from the user to the item, and the item is uniformly exposed to the users. However, this causal graph is usually not aligned with the one (see Figure 1(c)) inducing the offline datasets, thus the ideal learning objective can be intractable. To achieve debaised recommendation based on offline datasets, IPS is a very common strategy [23, 32], where the offline samples are re-weighted to approach the data distribution induced from the ideal causal graph. While IPS-based methods have achieved many promising results, the sample weights need to be predicted from the offline dataset, which can be inaccurate and may lead to the high variance

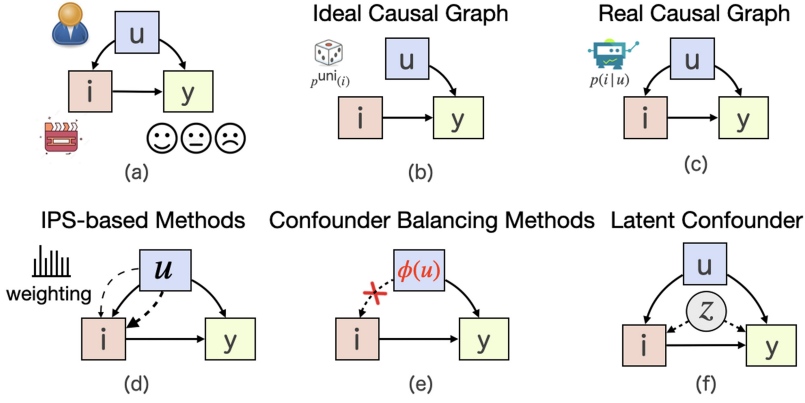


Fig. 1. (a) Illustration of the recommendation task within the potential outcome framework. (b) The causal graph of the ideal unbiased learning objective. (c) The causal graph generating the offline dataset. (d)–(e) Comparison on the debiasing mechanisms between the IPS-based and confounder balancing methods. (f) Illustration of the latent confounders in potential outcome framework.

issue [33, 34]. In the following sections, we propose an alternative debiased recommendation strategy based on confounder balancing,² which avoids the drawbacks of IPS.

3 DEBIASED RECOMMENDATION WITH CONFOUNDER BALANCING

In this section, we detail our idea. To begin with, we derive an upper bound of the ideal learning objective (1) by adjusting the user distributions. In the derived upper bound, the confounder (i.e., user) needs to be balanced for each item pair. Since the number of items can be quite large in practice, we design three strategies, including clipping, sampling, and adversarial learning, to facilitate the model training process. For more robust optimization, we, at last, propose to capture the potential latent confounders in recommender systems and combine them into the final learning objective.

3.1 Upper Bound of the Ideal Objective

In IPS-based methods, the ideal learning objective is achieved by re-weighting the offline training samples. To avoid the weaknesses of IPS, we take a different strategy. In specific, we hope to derive a surrogate learning objective, satisfying two properties: On one hand, it should be strictly not smaller than (Equation (1)), so minimizing the surrogate can approximately lower the ideal learning objective; on the other hand, the derived objective should be solvable only with the offline dataset. To proceed, we introduce a representative function $\phi : \mathcal{U} \rightarrow \mathbb{R}^d$, which projects the user information (defined on \mathcal{U}) into a d -dimensional space. To make it more clear, we summarize all the notations in Table 1. Then, we have the following theory:

THEOREM 1. *For measuring distribution distance, suppose IPM_G is the **integral probability metric (IPM)** defined in Reference [31], that is:*

$$IPM_G(p, q) := \sup_{g \in G} \int_T |g(t)(p(t) - q(t))| ds, \quad (2)$$

where G : is a function class, p and q are two density functions on representation T .

Let $l_{f, \phi}(u, i) = \int_y \delta(y, f(\phi(u), i)) p(y|u, i) dy$ be the local loss for the user-item pair (u, i) , where we regard the combination between f and ϕ as the recommender model f^* in Equation (1). We define

²Hereafter, we do not distinguish confounder balancing with user feature balancing.

Table 1. Notations Used throughout This Article

Notation	Meaning
$u \in \mathcal{U}$	User Information
$i \in [1, N]$	Item Index
$y_{ui} \in \mathbb{R}$	User Feedback of Item
$r_{ui} \in \mathbb{R}$	Expected User Feedback of Item
$t \in \mathbb{R}^d$	Debiased Representation
S	Set of Item Pairs
$z \in \mathbb{R}^c$	Confounder
$p_s(i u, z)$	Item Observing Probability
$\phi : \mathcal{U} \rightarrow \mathbb{R}^d$	Representation Function
$\delta : \mathbb{R} \rightarrow [0, 1]$	Loss Function
$f : \mathbb{R}^d \rightarrow \mathbb{R}$	Recommendation Function
$D : \mathbb{R}^d \rightarrow \mathbb{R}$	Discriminator
IPM_G	Integral Probability Metric
$g : \mathbb{R}^d \rightarrow \mathbb{R}$	IPM Discriminator
$c : \mathcal{U} \rightarrow \mathbb{R}^c$	Confounder Inference
L_{ideal}	Ideal Objective
L_{emp}	Empirical Objective
γ	Hyperparameter for tuning the importance of different objectives in Equation (11)
α, β	Hyperparameters for generating simulation datasets

$s_i = \int_{\mathcal{U}} l_{f, \phi}(u, i) p(u|i) du$, where $p(u|i)$ is the probability of observing u given i . Then, for any loss function δ , we have:

$$\begin{aligned}
& \mathbb{E}_u[\mathbb{E}_{i \sim p^{\text{uni}}(i)}[\delta(r_{ui}, f(\phi(u), i))]] \\
& \leq \frac{1}{N} \sum_{i=1}^N s_i + \frac{1}{N} \sum_{\substack{i, i' \in [1, N], \\ i \neq i'}} \{p(i) + p(i')\} B_{\phi} IPM_G(p_{\phi}^i, p_{\phi}^{i'}) + C,
\end{aligned} \tag{3}$$

where N is the number of items. $p(i)$ is the marginal probability of observing item i . Similar to Reference [16], we assume ϕ is invertible. Suppose ψ is the inverse of ϕ , then if $\mathbf{t} = \phi(\mathbf{u})$, we have $\mathbf{u} = \psi(\mathbf{t})$. We assume p_{ϕ}^i is the user distributions by projecting $p(u|i)$ with ϕ , that is, $p_{\phi}^i(\phi(u)) \stackrel{\text{def}}{=} p_{\phi}(\phi(u)|i)$. $J_{\psi}(\mathbf{t})$ is the Jacobian determinant of ψ . B_{ϕ} is a constant, such that $\frac{l_{f, \phi}(u, i) \cdot |J_{\psi}(\mathbf{t})|}{B_{\phi}} \in G$, $\forall i \in [1, N]$. C is a constant.

PROOF. To begin with, we decompose the ideal objective (1) as follows:

$$\begin{aligned}
L_{\text{ideal}} &= \mathbb{E}_u[\mathbb{E}_{i \sim p^{\text{uni}}(i)}[\delta(r_{ui}, f(\phi(u), i))]] \\
&= \int_{\mathcal{U}} \frac{1}{N} \sum_i \delta(r_{ui}, f(\phi(u), i)) p(u) du \\
&= \frac{1}{N} \sum_i \int_{\mathcal{U}} \delta(r_{ui}, f(\phi(u), i)) p(u, j) du dj,
\end{aligned} \tag{4}$$

where $p(u, j)$ is the probability of user-item pair (u, j) in the observed dataset. The last equation holds due to the properties on marginal distribution. $\delta(\cdot, \cdot)$ can be arbitrary loss functions such as RSME, binary cross entropy, among others.

To bound Equation (4), we introduce the following two notations:

$$\begin{aligned} k_1(i) &= \int_u \delta(r_{ui}, f(\phi(u), i)) p(u|i) du, \\ k_2(i, i') &= \int_u \delta(r_{ui}, f(\phi(u), i)) p(u|i') du \quad i \neq i'. \end{aligned} \quad (5)$$

Then, the ideal learning objective can be written as as: $L_{\text{ideal}} = \frac{1}{N} \sum_i [p(i)k_1(i) + \sum_{i' \neq i} p(i')k_2(i, i')]$.

By the above definition, we relate the idea learning objective to the observed data distribution. Recall that $r_{ui} = \mathbb{E}[y|u, i]$, which is intractable only from empirical samples. Thus, we further define the following notations:

$$\begin{aligned} \epsilon_F &= \int \delta(y_{ui}, f(\phi(u), i)) p(y_{ui}|u, i) p(u, i) du di dr_i \\ &= \sum_i p(i) \underbrace{\int_u \left[\int_{y_{ui}} \delta(y_{ui}, f(\phi(u), i)) p(y_{ui}|u, i) dy_{ui} \right] p(u|i) du}_{s_i} = \sum_i p(i) s_i \\ \epsilon_{CF}^i &= \sum_{i' \neq i} \int \delta(y_{ui}, f(\phi(u), i)) p(y_{ui}|u, i) p(u, i') du dy_{ui} \\ &= \sum_{i' \neq i} p(i') \underbrace{\int_u \left[\int_{y_{ui}} \delta(y_{ui}, f(\phi(u), i)) p(y_{ui}|u, i) dy_{ui} \right] p(u|i') du}_{t_{i'}} = \sum_{i' \neq i} p(i') t_{i'}, \end{aligned} \quad (6)$$

where y_{ui} is the observed feedback of user u on item i .³

Based on the above definition, we formally derive the upper bound of L_{ideal} following three key steps: (1) the first step is to relate $k_1(i)$ and $k_2(i, i')$ with s_i and $t_{i'}$ by bounding $|s_i - k_1(i)|$ and $|t_{i'} - k_2(i, i')|$, respectively. Then, L_{ideal} can be represented by s_i and $t_{i'}$. (2) Since $t_{i'}$ contains unobserved user-item pairs that are intractable, the second step is to connect s_i and ϵ_{CF}^i , so L_{ideal} can be only dependent on s_i . (3) Based on (1) and (2), we derive the upper bound of L_{ideal} to end the proof.

(1) Relating $k_1(i)$ and $k_2(i, i')$ with s_i and $t_{i'}$: Let $P_i = |s_i - k_1(i)|$ and $Q_i = |t_{i'} - k_2(i, i')|$, then we have following Lemma:

LEMMA 3.1. *If the loss function δ is differentiable, then for any item i there exist constants $M_i > 0, \beta > 0, \gamma \geq \beta$, such that:*

$$\begin{aligned} P_i &= |s_i - k_1(i)| \\ &\leq \int_u |E_{y_{ui}}[\delta(y_{ui}, f(\phi(u), i))] - \delta(r_{ui}, f(\phi(u), i))| p(u|i) du \\ &\leq \int_u M_i(\rho_{\beta i} + \rho_{\gamma i}) p(u|i) du \\ &= M_i(\rho_{\beta i} + \rho_{\gamma i}), \\ Q_i &= |t_{i'} - k_2(i, i')| \\ &\leq \int_u |E_{y_{ui}}[\delta(y_{ui}, f(\phi(u), i))] - \delta(r_{ui}, f(\phi(u), i))| p(u|i') du \\ &\leq \int_u M_i(\rho_{\beta i} + \rho_{\gamma i}) p(u|i') du \\ &= M_i(\rho_{\beta i} + \rho_{\gamma i}). \end{aligned} \quad (7)$$

³It should noted that, in the main article, we use u_t , i_t , and y_t to represent the same concepts.

The second inequality in P_i and Q_i holds because of the Jensen bound demonstrated in Reference [10].

Based on Lemma 3.1, we could get the following upper bound of L_{ideal} :

$$\begin{aligned}
 L_{\text{ideal}} &= \frac{1}{N} \sum_i \left[p(i)k_1(i) + \sum_{i' \neq i} p(i')k_2(i, i') \right] \\
 &\leq \frac{1}{N} \sum_i [p(i)[s_i + M_i(\rho_{\beta i} + \rho_{\gamma i})] + \sum_{i' \neq i} p(i')[t_{ii'} + M_i(\rho_{\beta i} + \rho_{\gamma i})]] \\
 &= \frac{1}{N} \epsilon_F + \frac{1}{N} \sum_i \epsilon_{CF}^i + \frac{1}{N} \underbrace{\sum_i [p(i)x(i) + \sum_{i'} p(i')x(i)]}_{\text{constant}} \\
 &= \frac{1}{N} \sum_i p(i)s_i + \frac{1}{N} \sum_i \epsilon_{CF}^i + C,
 \end{aligned} \tag{8}$$

where C is a constant.

(2) Relating ϵ_{CF}^i with s_i : In Equation (8), ϵ_{CF}^i cannot be estimated from the observation data, since it contains user-item pairs that are unobserved. To solve this problem, we connect ϵ_{CF}^i with ϵ_F by the following lemma:

LEMMA 3.2. *Let ψ denote the inverse function of ϕ , t denotes representation of u (i.e., $t = \phi(u)$), then:*

$$\begin{aligned}
 \epsilon_{CF}^i - \sum_{i' \neq i} p(i')s_i &= \sum_{i' \neq i} p(i') \int_u \mathbb{E}_{y_{ui}} [\delta(y_{ui}, f(\phi(u), i))] (p(u|i) - p(u|i')) du \\
 &= \sum_{i' \neq i} p(i') \int_t l_{f, \phi}(\psi(t), i) (p_\phi^i(t) - p_\phi^{i'}(t)) |J_\psi(t)| dt \\
 &\leq \sum_{i' \neq i} p(i') B_\phi \text{IPM}_G(p_\phi^i(t), p_\phi^{i'}(t)).
 \end{aligned} \tag{9}$$

In Lemma 3.2, we use s_i to bound the crucial intractable term ϵ_{CF}^i . The $p_\phi^i(t), p_\phi^{i'}(t)$ in the bottom line are the converted user feature representations for item i and i' , respectively. The distance between ϵ_{CF}^i and $\sum_{i' \neq i} p(i')s_i$ will be minimized when $p_\phi^i(t)$ is equal to $p_\phi^{i'}(t)$ for any $i' \neq i$.

(3) Deriving the final upper bound of L_{ideal} : By bringing Equation (9) into (8), we have:

$$\begin{aligned}
 L_{\text{ideal}} &= \frac{1}{N} \sum_{i=1}^N p(i)k_1(i) + \frac{1}{N} \sum_{i=1}^N \sum_{i' \neq i} p(i')k_2(i, i') \\
 &\leq \frac{1}{N} \sum_{i=1}^N p(i)s_i + \frac{1}{N} \sum_{i=1}^N \sum_{i' \neq i} p(i')t_{ii'} + C \\
 &\leq \frac{1}{N} \sum_{i=1}^N p(i)s_i + \frac{1}{N} \sum_{i=1}^N \sum_{i' \neq i} p(i')[s_i + B_\phi \text{IPM}_G(p_\phi^i, p_\phi^{i'})] + C \\
 &= \frac{1}{N} \sum_{i=1}^N s_i + \frac{1}{N} \sum_{\substack{i, i' \in [1, N], \\ i \neq i'}} \{p(i') + p(i)\} B_\phi \text{IPM}_G(p_\phi^i, p_\phi^{i'}) + C,
 \end{aligned} \tag{10}$$

where C absorbs all the constants, and the last equation holds because $\text{IPM}_G(p_\phi^i, p_\phi^{i'})$ is symmetrical for i and i' . \square

On this theory, we have the following remarks:

- This objective mainly includes two parts: The first one aims to learn the recommender model based on the projected user information $\phi(u)$, and the second one targets at minimizing the distance between the projected user distributions given each pair of items. Since s_i is defined on the observational data, this upper bound can be optimized with offline datasets. It should be noted that while we do not directly optimize the unbiased objective, lowering the upper bound is empirically demonstrated to be effective, which, we speculate, shares similar insights with many machine learning models like variational inference [39].
- The representative function ϕ has two effects: (i) “adjusting” all the irregular user feature distributions to a target one and (ii) “learning” the recommender model based on the target distribution. Such “adjusting-learning” effects actually aim to solve the key problem of distribution shift in debiased recommendation [50]. For a shifted user feature distribution in the testing set, the learned ϕ can effectively project it to the target distribution, which has been well handled in the training phrase.
- From the causal graph perspective, the effect of the second part in the upper bound is to cut down the causal relation from the user to the item. The optimal value of minimizing $\text{IPM}_G(p_\phi^i, p_\phi^{i'})$ for each item pair is $p_\phi(\phi(u)|i = 1) = p_\phi(\phi(u)|i = 2) = \dots = p_\phi(\phi(u)|i = N) = p_\phi(\phi(u))$, which implies that $\phi(u)$ is independent of i , that is, there is no “user \rightarrow item” edge in the causal graph.
- Notably, many previous works on confounder balancing leverage similar techniques to derive solvable learning objectives [16]. However, our result is more general in two aspects: (i) previous studies are highly tailored for MSE, but our learning objective is compatible with any loss function, and (ii) the treatment is no longer restricted to be binary—its size can be much larger.

Empirical loss of the upper bound. Suppose the offline recommender dataset is $\{(u_t, i_t, y_t)\}_{t=1}^T$, where each sample (u_t, i_t, y_t) includes a user, an item, and the user feedback on the item, and there are in total T samples. Then, the empirical loss corresponding to the upper bound in Equation (3) is:

$$\begin{aligned}
 & L_{\text{emp}}(\theta_f, \theta_\phi) \\
 &= \frac{1}{T} \sum_{t=1}^T \frac{1}{p(i_t)} \delta(y_t, f(\phi(u_t), i_t)) + \lambda_f \|\theta_f\|_2^2 + \lambda_\phi \|\theta_\phi\|_2^2 + \\
 & \quad \gamma \sum_{\substack{i, i' \in [1, N], \\ i \neq i'}} \{p(i) + p(i')\} \text{IPM}_G(\{\phi(u_t)\}_{t: i_t=i}, \{\phi(u_t)\}_{t: i_t=i'}),
 \end{aligned} \tag{11}$$

where θ_f and θ_ϕ are the parameters of the recommender model f and representative function ϕ . $p(i)$ is approximated as $\frac{T_i}{T}$, and T_i is the frequency of item i in the dataset. Thus, the weight $\frac{1}{p(i_t)}$ compensates for different item observation frequencies. Similar to Reference [30], $\text{IPM}_G(\cdot; \cdot)$ is the empirical integral probability metric w.r.t. G , and B_ϕ is regarded as a part of the hyper-parameter γ .

3.2 Efficient Confounder Balancing

To achieve confounder balancing, the user distribution distance is minimized for each item pair in objective (11). Suppose there are N items in the system, then we need to optimize C_N^2 balancing

terms (i.e., $\text{IPM}_G(\cdot; \cdot)$), which can be quite inefficient or even infeasible in practical recommender systems. To alleviate this problem, we propose the following three strategies to facilitate model training:

Clipping. This method directly drops the balancing terms that are not important. In specific, let $p_{ii'} = p(i') + p(i)$ be the importance score of item pair (i, i') . Then, we define S as the set of item pairs, which has the largest $p_{ii'}$'s, and we suppose $|S| = K_1$. At last, we change the last part of Equation (11) to:

$$\gamma \sum_{i, i' \in S} p_{ii'} \text{IPM}_G(\{\phi(u_t)\}_{t:i_t=i}, \{\phi(u_t)\}_{t:i_t=i'}), \quad (12)$$

where we directly remove the balancing terms with smaller importance scores.

Sampling. In this method, we randomly sample K_2 balancing terms for optimization in each training epoch. The balancing term for (i, i') is sampled according to the probability $\frac{p_{ii'}}{\sum_{j, j' \in [1, N], j \neq j'} p_{jj'}}$. By this method, the number of optimized balancing terms is reduced from C_N^2 to K_2 , which improves the training efficiency. Since the balancing terms are selected in a random manner, the item pairs with smaller $p_{ii'}$'s still have chances to be optimized, which improves the model robustness as compared with the clipping strategy.

Adversarial learning. While the above two methods can enhance the training efficiency, they have to abandon some balancing terms either deterministically or stochastically. To fully satisfy the balancing requirements, we propose an adversarial training strategy. Our general idea is to introduce a discriminator D , which serves as a classifier to identify which distribution (of $p(u|i = t), t \in [1, N]$) an instance is sampled from. Then, the representative function ϕ is optimized to make the instance $\phi(u)$ unidentifiable. Formally, we have the following learning objective:

$$\begin{aligned} & L_{\text{emp}}^{\text{adv}}(\theta_f, \theta_\phi, \theta_D) \\ &= \frac{1}{T} \sum_{t=1}^T \frac{1}{p(i_t)} \delta(y_t, f(\phi(u_t), i_t)) + \gamma \sum_{i=1}^N \mathbb{E}_{u \sim p(u|i)} [\log D^i(\phi(u))] \\ &+ \sum_{k \in \{f, \phi, D\}} \lambda_k \|\theta_k\|_2^2, \end{aligned} \quad (13)$$

where θ_D is the parameter of D . The output layer of D is softmax for classification, and D^i denotes the i th element. Thus, $\sum_{i=1}^N D^i(\phi(u)) = 1$. $\mathbb{E}_{u \sim p(u|i)}$ can be empirically estimated by $\sum_{u \in I_u}$, and I_u is the item set interacted by user u . We optimize this objective based on the following constraint minimax game:

$$\begin{aligned} & \min_{\theta_f, \theta_\phi} \max_{\theta_D} L_{\text{emp}}^{\text{adv}}(\theta_f, \theta_\phi, \theta_D) \\ & \text{s.t. } \sum_{i=1}^N D^i(\phi(u)) = 1, \end{aligned} \quad (14)$$

where θ_D is learned to maximize the likelihood of the right classification results. θ_f is optimized to fit the recommendation data. θ_ϕ is learned to jointly hide the identification of $\phi(u)$ and achieve better recommendation results. To see the effect of the above adversarial learning strategy, we have the following theory:

THEOREM 2. Let p_ϕ^i denote the user distributions by projecting $p(u|i)$ with ϕ , then the minimax game defined by

$$\begin{aligned} \min_{\theta_\phi} \max_{\theta_D} \sum_{i=1}^N \mathbb{E}_{u \sim p(u|i)} [\log D^i(\phi(u))] \\ \text{s.t. } \sum_{i=1}^N D^i(\phi(u)) = 1 \end{aligned} \quad (15)$$

has a global optimal solution when $p_\phi^1 = p_\phi^2 = \dots = p_\phi^N$.

PROOF. Suppose $t = \phi(u)$, then objective (15) can be written as:

$$\begin{aligned} \max_{\theta_D} \sum_{i=1}^N \int_t p_\phi^i(t) \log D^i(t) dt \\ \text{s.t. } \sum_{i=1}^N D^i(t) = 1. \end{aligned} \quad (16)$$

We use Lagrange multiplier to solve the above constraint optimization problem, which induces the following objective:

$$L(D, \lambda) = \max_{\theta_D} \sum_{i=1}^N \int_s p_\phi^i(t) \log D^i(t) dt + \lambda \left(\sum_{i=1}^N D^i(t) - 1 \right). \quad (17)$$

Let $\frac{\partial L(D, \lambda)}{\partial D} = 0, \frac{\partial L(D, \lambda)}{\partial \lambda} = 0$, we have:

$$\begin{aligned} D^i(t) &= -\frac{p_\phi^i(t)}{\lambda} \\ \lambda &= -\sum_k^N p_\phi^k(t). \end{aligned} \quad (18)$$

We then substitute Equation (18) into $\sum_{i=1}^N \int_t p_\phi^i(t) \log D^i(t)$ and obtain:

$$B = \sum_{i=1}^N \int_t p_\phi^i(t) \log \frac{p_\phi^i(t)}{\sum_k^N p_\phi^k(t)} dt. \quad (19)$$

The following proof is based on **Jensen-Shannon Divergence (JSD)**, which is defined as follows:

Definition 1 (JSD [9]). Denote P_i a random distribution where $i \in \{1, 2, \dots, N\}$. Let $M := \sum_{i=1}^N \frac{1}{N} P_i$, the general multivariate Jensen-Shannon Divergence (JSD) is defined as

$$\begin{aligned} \text{JSD}(P_1, P_2, \dots, P_n) &= \sum_i \frac{1}{N} \text{KL}(P_i \| M) \\ &= H\left(\sum_{i=1}^n \frac{1}{N} P_i\right) - \sum_{i=1}^n \frac{1}{N} H(P_i), \end{aligned}$$

where KL is the Kullback–Leibler divergence, and H is the Shannon entropy. According to Reference [9], JSD is maximized when $P_1 = P_2 = \dots = P_N$.

Finally, we have:

$$\begin{aligned}
 B &= \sum_{i=1}^N \int_t p_{\phi}^i(t) \log \frac{p_{\phi}^i(t)}{\sum_k^N p_{\phi}^k(t)} dt \\
 &= \sum_{i=1}^N E_{p_{\phi}^i(t)} \left[\log \frac{p_{\phi}^i(t)}{\sum_k^N p_{\phi}^k(t)} \right] + \log N \\
 &= \sum_{i=1}^N D_{KL}(p_{\phi}^i(t) \parallel \frac{1}{N} \sum_k^N p_{\phi}^k(t)) + \log N \\
 &= N \text{JSD}(p_{\phi}^1(t), p_{\phi}^2(t), \dots, p_{\phi}^N(t)).
 \end{aligned} \tag{20}$$

The above equation is maximized when all $p_{\phi}^i(t)$'s are equal, that is:

$$p_{\phi}^1(t) = p_{\phi}^2(t) = \dots = p_{\phi}^N(t). \tag{21}$$

Thus, the optimal solution for the adversarial objective in Equation (13) is the same as that of the IPM balancing terms in Equation (11). \square

From this theory, we can see, the optimal solution is exactly what we expect from the pair-wise balancing terms in objective (11). However, by this adversarial learning strategy, the number of optimization terms is reduced from $O(N^2)$ to $O(N)$.

3.3 Latent Confounder Modeling

Real-world recommender systems are usually very complex; there can be many unobserved factors (e.g., item promotion, user emotions) that simultaneously influence the item exposure and user feedback. Such factors are called latent confounders in potential outcome framework (see Figure 1(f)). The existence of latent confounders may: (i) lead to lowered recommendation performance due to the lack of sufficient context modeling and (ii) introduce uncontrollable cause to influence the item exposure probability, which cannot be removed by just balancing the user distributions, and thus influences the debiasing effect. To capture latent confounders, we follow the previous work [12, 25, 38] to use a neural network c to infer them. More specifically, it is intuitive that latent confounders should be related with user personalities; for example, the confounders on user emotions. In addition, according to the causal graph in Figure 1(f), the latent confounders are also related with the item. As a result, we estimate the latent confounders by: $z_t = c(i_t, u_t)$. It should be noted that the latent confounder prediction is an inference process; it does not mean there is a causal edge from the item to the latent confounders, since causal graph describes the data generation process. Besides, the output y_t is not leveraged as a signal to estimate z_t , since it is not observed in advance. Similar to the previous work [12, 25, 38], we learn the parameters in c by maximizing the likelihood of observing the exposed items. The final learning objective by incorporating the latent confounders is:

$$\begin{aligned}
 &L_{\text{emp}}^{\text{conf}}(\theta_f, \theta_{\phi}, \theta_D, \theta_c, \theta_s) \\
 &= \frac{1}{T} \sum_{t=1}^T \frac{1}{p(i_t)} \delta(y_t, f(\phi(u_t, z_t), i_t)) + \gamma \sum_{i=1}^N \mathbb{E}_{u \sim p(u|i)} [\log D^i(\phi(u, z_t))] \\
 &\quad - \frac{1}{T} \sum_{t=1}^T \log p_s(i_t | u_t, z_t) + \sum_{k \in \{f, \phi, D, c, s\}} \lambda_k \|\theta_k\|_2^2,
 \end{aligned} \tag{22}$$

ALGORITHM 1: Learning algorithm of our model

```

Randomly initialize  $\theta_f, \theta_\phi, \theta_D, \theta_c, \theta_s$ .
Indicate the number of training epochs  $E$ .
Indicate the iteration numbers  $E_D$  and  $E_G$ .
Indicate the learning rates  $\eta_1$  and  $\eta_2$ .
for  $e$  in  $[0, E]$  do
     $\hat{\theta}_c \leftarrow \theta_c, \hat{\theta}_\phi \leftarrow \theta_\phi$ 
    for  $k$  in  $[0, E_D]$  do
        #  $L_{dis}(\theta_D, \hat{\theta}_c, \hat{\theta}_\phi) = \sum_{i=1}^N \mathbb{E}_{u \sim p(u|i)} [\log D^i(\phi(u, z_t))]$ .
         $\theta_D \leftarrow \theta_D + \eta_1 \frac{\partial L_{dis}(\theta_D, \hat{\theta}_c, \hat{\theta}_\phi)}{\partial \theta_D}$ .
    end
     $\hat{\theta}_D \leftarrow \theta_D$ .
    for  $l$  in  $[0, E_G]$  do
        for  $k$  in  $\{f, \phi, c, s\}$  do
             $\theta_k \leftarrow \theta_k - \eta_2 \frac{\partial L_{emp}^{conf}(\theta_f, \theta_\phi, \hat{\theta}_D, \theta_c, \theta_s)}{\partial \theta_k}$ .
        end
    end
end

```

where θ_c collects all the parameters in c . The representative function ϕ is applied to both u_t and z_t , the third term aims to maximize the likelihood of observing item i_t under the effect of the latent confounders. p_s is a probability function with θ_s as its parameters.

Model specification. In the above sections, we have detailed our idea. We specify different parts of objective (22) as follows: δ is implemented by the cross-entropy loss to model user implicit feedback. ϕ is a simple linear function. The categorical features for the users are first converted to one-hot vectors and then encoded by embedding matrices. The continuous features are directly multiplied by weighting matrices to derive the representation. The discriminator D is a fully connected neural network, that is,

$$D(\phi(u_t, z_t)) = \sigma(W_1 \text{ReLU}(W_2 \phi(u_t, z_t))), \quad (23)$$

where W_1 and W_2 are weighting parameters and σ and ReLU are activation functions. The model for inferring latent confounders c is a two-layer neural network with ReLU as the activation function, that is,

$$c(u_t; i_t) = V_1 \text{ReLU}(V_2 \text{ReLU}(V_3 [u_t; i_t])), \quad (24)$$

where V_1 , V_2 , and V_3 are weighting parameters. $[\cdot; \cdot]$ is the concatenate operation. p_s is also a two-layer neural network, but the input is u_t , z_t , and i_t , and the output is a scalar between 0 and 1, that is,

$$p_s(i_t | u_t, z_t) = \sigma(Q_1 \text{ReLU}(Q_2 \text{ReLU}(Q_3 [u_t; i_t; z_t]))), \quad (25)$$

where Q_1 , Q_2 , and Q_3 are weighting parameters.

Learning algorithm. The complete learning algorithm of our framework is summarized in Algorithm 1. To begin with, the model parameters are randomly initialized. In each training epoch, the discriminator θ_D is first optimized by maximizing L_{dis} with fixed $\hat{\theta}_g$ and $\hat{\theta}_\phi$. Then, the recommender model, representative function, and likelihood of the treatment are jointly optimized to minimize L_{emp}^{conf} by fixing the discriminator $\hat{\theta}_D$.

Remark. In IPS-based methods, the data distribution is adjusted to the ideal one by re-weighting the training samples, which basically smooths the conditional probability corresponding to the

causal relation “ $u \rightarrow i$ ” (see Figure 1(d)). In our user feature balancing method, we project the user distribution to make it independent of the item, which cuts down the causal relation “ $u \rightarrow i$ ” (see Figure 1(e)). Basically, IPS and our method achieve debiased recommendation with totally different principles. In the previous work, IPS-based methods have been well studied. In this article, we propose the first solution to debiased recommendation based on confounder balancing, which paves a new way for this research field.

4 RELATED WORK

Our work aims to push the boundary of debiased recommendation, which holds the promise of learning user unbiased preference for adapting complex testing environments. Traditionally, there are two types of strategies to build debiased recommender models [4, 6, 7, 39, 48, 48]. The first one is sample generation methods, where the key idea is to impute user preference on unexposed items, so the model can be directly trained on the full sample space. For example, Reference [18] proposes to train an imputation model based on the observed data, and then the recommendation algorithm is learned based on both of the imputed and original samples. The second strategy is distribution adjusting methods [28, 29]. The main idea is to re-weight the training samples for adjusting the offline data distribution to be similar with the ideal unbiased one. Almost all the models in this category rely on IPS; for example, References [2, 18, 33, 34] leverage different strategies to estimate the inverse propensity scores, which are used to re-weight the user-item interactions. Reference [34] proposes to compute the propensity score by self-normalization to reduce the variance. Xu et al. [44] design an adversarial method for debiased recommendation based on more robust IPS estimation. As mentioned above, IPS-based methods can be flawed when facing the recommendation domain, since the sample weights need to be estimated from the noisy recommendation data, which can be unreliable and lower the downstream recommendation performance. There are also many studies on combining the former two methods, aiming to take both of their advantages. For example, Reference [8] proposes to simultaneously estimate IPS and learn imputation models to obtain more robust performance. Yuan et al. [46] introduce uniform data to train the imputation model. Our idea lies in the second strategy. However, we do not estimate the sample weights and thus do not suffer from the weaknesses, such as low estimation accuracy and high variance, brought by IPS-based models.

In another related field of causal inference, there are many studies on confounder balancing. For example, Reference [15] estimates the **average treatment effect (ATE)** by balancing the heterogeneous confounders. Reference [30] uses representation learning to balance the confounders to predict the **individual treatment effect (ITE)**. Reference [21] also aims to learn ATE, but it targets at networked data, where the confounder is balanced by considering the graph structure information. These works mostly aim to solve the treatment-effect estimation problem, which is quite different from the recommendation task, ranging from the loss function, treatment space, to the potential latent confounders. Recently, we have noticed that Wang et al. [35] introduce a confounder inference method for building debiased recommender models. The basic idea of this work comes from Pearl’s causal inference framework [24], which focuses on the identifiability problem based on the back-door or front-door criterion. In our work, we build debiased recommender models based on potential outcome framework, and we achieve this goal by balancing user feature distributions for different items.

5 EXPERIMENTS

In this section, we conduct extensive experiments to demonstrate the effectiveness of our idea, focusing on the following research questions:

- What is the overall performance of our model comparing with the state-of-the-art methods?
- What are the effects of different model components?
- How does the data bias severity influence the recommendation performance?
- How does the data confounderness influence the recommendation performance?
- How does the data sparsity influence the recommendation performance?
- How does the balancing parameter γ influence the recommendation performance?

In the following sections, we first detail the experiment setup and then answer these questions based on the experiment results.

5.1 Experiment Setup

5.1.1 Datasets. We base our experiments on both synthetic and real-world datasets. The synthetic dataset is built according to the method leveraged in Reference [50]. In specific, we simulate 10,000 users and 32 items. For each user i or item j , the features $\mathbf{p}_i \in \mathbb{R}^d$ or $\mathbf{q}_j \in \mathbb{R}^d$ are generated from a multi-variable Gaussian distribution $\mathcal{N}(0, \mathbf{I})$, where d and \mathbf{I} represent the feature dimension and unit matrix, respectively. When generating the recommendation list for user i , we introduce a random variable $z \sim \mathcal{N}(0, 1)$ to model the influence of the latent confounders. In specific, for each item j , we define a score as follows:

$$r_{ij} = 1 - \sigma[(1 - \alpha)(1 - \beta)\mathbf{a}^T \kappa_1(\kappa_2([\mathbf{p}_i, \mathbf{q}_j])) + \alpha + \beta z + \mathcal{N}(0, 0.02)]], \quad (26)$$

where $\mathbf{a} \in \mathbb{R}^{2d}$ is specified as an all-one vector. $\alpha \in [0, 1]$ defines the bias severity, e.g., $\alpha = 1$ means the item is recommended in a complete random manner, that is, the recommendation of an item is irrelevant with the user. $\beta \in [0, 1]$ trades off the influence of latent confounders, e.g., $\beta = 1$ implies that the latent confounders dominate the system. In the experiment, the default settings of α and β are both 0.5. $\kappa_1(\cdot)$ and $\kappa_2(\cdot)$ as piecewise functions [50]. $\kappa_1(x) = x - 0.5$ if $x > 0$, otherwise $\kappa_1(x) = 0$. $\kappa_2(x) = x$ if $x > 0$, otherwise $\kappa_2(x) = 0$. For each user i , we recommend item j according to a Bernoulli distribution with the mean of r_{ij} . The length of the recommendation list is set as 5. We follow the previous work [50] to generate the feedback of a user i on an item j as follows:

$$\begin{aligned} x_1 &= \mathbf{a}^T \kappa_1(\kappa_2([\mathbf{p}_i, \mathbf{q}_j])) + b \\ x_2 &= \mathbf{a}^T \kappa_3(\kappa_2([\mathbf{-p}_i, \mathbf{-q}_j])) + b \\ s_{ij} &= \mathbb{I}(\sigma(x_1 + x_1 \cdot x_2) + \beta z - 0.5), \end{aligned} \quad (27)$$

where $\kappa_3(x) = x + 0.5$ if $x < 0$, otherwise $\kappa_3(x) = 0$. $\mathbb{I}(x)$ is an indicator function, which is 1 if $x > 0$, and 0 otherwise. In the experiment, we follow References [1, 51] to build the testing set by uniformly recommending different items to each user.

For real-world experiments, we evaluate our model based on the following datasets:

- **Amazon**⁴ is an e-commerce dataset including user feedback on products from multiple categories.
- **Anime**⁵ is crawled from myanimelist.net, which includes the user preference on a large number of animes.
- **Yahoo! R3**⁶ is an online music recommendation dataset, which contains the user survey data and ratings for randomly selected songs between August 22, 2006, and September 7, 2006.

⁴<http://jmcauley.ucsd.edu/data/amazon/>.

⁵<https://www.kaggle.com/CooperUnion/anime-recommendations-database>.

⁶<https://webscope.sandbox.yahoo.com/catalog.php?datatype=r>.

Table 2. Summarization of the datasets. “Full,” “Semi,” and “No” indicate fully synthetic, semi-synthetic (only simulating the test set) and real-world datasets

Dataset	# User	# Item	Density	Domain	Type
Synthetic	10,000	32	1.756%	-	Full
Yahoo! R3	14,877	1,000	0.812%	Music	No
Coat	290	300	5.333%	Clothing	No
PCIC	1,000	1,720	0.241%	Movie	No
Amazon	1,323,884	61,275	0.003%	E-commerce	Semi
Anime	73,516	12,294	0.86%	Anime	Semi

- **Coat**⁷ is a commonly used dataset for evaluating debiased recommender models.
- **PCIC**⁸ is a recently released dataset for debiased recommendation, where we are provided with the user preferences on uniformly exposed movies.

Evaluating debiased recommender models should be based on uniform testing sets, which has been provided in Yahoo! R3, Coat, and PCIC. For Amazon and Anime, we follow the previous work [27, 49] to resample the original testing sets to simulate uniform data according to the inverse item frequency. The statistics of all the above datasets are summarized in Table 2, where we can see they can cover different application domains, and their densities vary a lot. By experimenting on these datasets, we hope to fairly evaluate different models and demonstrate the generality of our idea.

5.1.2 Baselines. The following representative baselines are selected in our experiments:

- **Direct Method (Direct)** [18] is a sample generation model, where the counterfactual data is filled by a learned model before model optimization. In this model, the label of counterfactual data is estimated by a model learned from observational data.
- **Inverse Propensity Score (IPS)** [33] is a distribution adjusting model, where the samples are re-weighted to convert the observed data distribution to the ideal unbiased one. The method is achieved by calculating propensity score over observational data first.
- **Self-normalized IPS (SNIPS)** [34] is an extension of IPS, where the learning objective variance is reduced by a normalization strategy.
- **Doubly Robust (DR)** [8] is a combination between IPS and Direct Method.
- **ATT** [27] is an extension of the direct method based on meta-learning, which leverages two imputation models to double-check the correctness of counterfactual labels.
- **CVIB** [40] is a debiased method based on information bottleneck.
- **ACL** [44] is a debiased recommender model based on adversarial training.

In the experiments, we denote our model by **CBR**, which is short for **c**onfounder **b**alancing-based **r**ecommender method. For demonstrating the effectiveness of our neural projection model, we also introduce a baseline by specifying $\phi(u_t, z_t) = w \cdot (u_t + z_t) + b$, where $w, b \in \mathbb{R}$ are both scalars. We call this method as CBR(new).

5.1.3 Implementation Details. For Yahoo! R3, Coat, and PCIC, we directly use the testing set provided in the original data to validate and evaluate different models. For the simulation dataset, the ratio between the training and testing (including validation) sets is controlled as 3:1, where the training set is biased by the recommender model, while the testing set is unbiased. For the other

⁷<https://www.cs.cornell.edu/~schnabts/mnar/>.

⁸<https://competition.huaweicloud.com/information/1000041488/introduction>.

datasets, we use 20% of each user's interactions for testing, while the others are left for training (70%) and validation (10%). We evaluate different recommender models based on the well-known metrics, including Recall, AUC [26], ACC [11], and NDCG. Recall measures the overlapping between the recommended items and the ground truth. The latter three metrics are ranking-sensitive, where the higher-ranked correct items contribute more to the results. For each model, we generate 10 recommendations to be compared with the ground truth.

The hyper-parameters are determined based on grid search. In specific, the learning rate and batch size are tuned in the ranges of $[10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}]$ and $[64, 128, 256, 512, 1024]$, respectively. The weighting parameter λ is determined in $[0.001, 0.01, 0.05, 0.1, 0.5]$. The regularization coefficients λ_k 's are all searched in $[0.001, 0.005, 0.01, 0.05]$. The user/item embedding dimension is empirically set as 32. We apply our framework to different base models including **MLP** and **GMF** [14], which are both generalized matrix factorization methods, covering different merging strategies between the user and item embeddings. For the baselines, we set the parameters as the optimal values reported in the original paper or tune them in the same ranges as our model's. All the experiments are conducted based on a server with a 16-core CPU, 128 G memory and RTX 5000 GPU.

5.2 Overall Performance Comparison

The overall comparison results can be seen in Table 3, from which we can see: In most cases, direct method performs worse than IPS or SNIPS. We speculate that the recommendation dataset is quite noisy, and the imputation model learned based on it can be unreliable. Since the recommender model is learned based on a large amount of user feedback estimated from the imputation model, the performance can not be guaranteed. ACL and DR can usually achieve better performance than IPS, SNIPS, and Direct, which agrees with the previous work [8]. Two state-of-art methods ATT and CVIB perform better than other baselines in most cases. It is because they both consider more general situation in real-world system to avoid the noise interference and have stronger theoretical guarantee. In most cases, CVIB is better than ATT; it is because ATT is an extension of direct method, which depends to some extent on how well the imputation model performs. Encouragingly, our model can achieve the best performance on all the datasets across different base models and evaluation metrics. In specific, for MLP, our model can, on average, improve the performance of the base model by about 10.88%, 6.87%, 4.21%, and 4.03% on NDCG@10, Recall@10, AUC, and ACC, respectively. For GMF, the performance gains on these metrics are about 12.06%, 8.27%, 11.24%, and 5.58%. This observation demonstrates the effectiveness of our idea comparing with existing mainstream debiased recommender models. The reason can be that the baseline models usually need to introduce additional models for estimating the inverse propensity score or user feedback. The prediction error of these models may worsen the downstream recommendation performance. However, in our framework, all the parameters are learned in an end-to-end manner, where there is no accumulation error, and the item balancing and recommendation tasks are jointly optimized for beneficial knowledge transfer. As a result, we can observe improved performance of our framework. By comparing our model with its variant based on linear projection function (i.e., CBR(new)), we find that our model can achieve better performance in most cases. This suggests that the neural projection method is indeed helpful, which positively confirms our model designs.

5.3 Ablation Studies

For better understanding our model, in this section, we conduct many ablation studies. We compare our model with its three variants:

Table 3. Results of the Overall Comparison

Dataset		GMF				MLP			
Metrics		NDCG@10	Recall@10	AUC	ACC	NDCG@10	Recall@10	AUC	ACC
Synthetic	Base	0.8479	0.5364	0.8184	0.7408	0.8777	0.5822	0.7465	0.7552
	IPS	0.8578	0.5524	0.8102	0.7531	0.8991	0.575	0.7542	0.7496
	SNIPS	0.8634	0.5514	0.8193	0.7509	0.9016	0.5726	0.7672	0.7644
	Direct	0.8095	0.4398	0.7301	0.6764	0.8783	0.5355	0.6081	0.5812
	DR	0.8642	0.5787	0.8294	0.7524	0.8946	0.5641	0.7533	0.7548
	ATT	0.8686	0.5676	0.8219	0.7635	0.8804	0.5881	0.7559	0.7525
	CVIB	0.8875	0.5765	0.8269	0.7627	0.9004	0.5746	0.7678	0.7514
	ACL	0.8976	0.5865	0.8175	0.7316	0.8985	0.5951	0.7691	0.7303
	CBR(new)	0.8933	0.5904	0.7868	0.7214	0.8821	0.5677	0.7526	0.6882
	CBR	0.9034*	0.5969*	0.8336*	0.7708*	0.9047	0.5987	0.7763*	0.7702*
Yahoo! R3	Base	0.5162	0.4874	0.6793	0.5619	0.5089	0.4839	0.6762	0.5696
	IPS	0.6456	0.6001	0.6822	0.5671	0.6533	0.5889	0.6825	0.5677
	SNIPS	0.6404	0.6014	0.6899	0.569	0.6618	0.5946	0.6866	0.5701
	Direct	0.6361	0.568	0.6723	0.5766	0.6488	0.5298	0.6883	0.5801
	DR	0.657	0.5787	0.6849	0.5698	0.6592	0.5953	0.6922	0.5823
	ATT	0.6331	0.5624	0.6519	0.5835	0.644	0.5607	0.6609	0.5725
	CVIB	0.6564	0.5856	0.6869	0.5627	0.6614	0.5955	0.6935	0.5514
	ACL	0.6307	0.5684	0.6878	0.5917	0.6303	0.5966	0.6916	0.5775
	CBR(new)	0.6544	0.5777	0.6759	0.5732	0.6448	0.5742	0.6594	0.5381
	CBR	0.6601*	0.6027*	0.6909*	0.5788	0.6649	0.6015	0.6955	0.5746
Coat	Base	0.61	0.6683	0.5778	0.5198	0.6277	0.6775	0.5926	0.5346
	IPS	0.6437	0.711	0.6224	0.5996	0.6312	0.685	0.5918	0.5493
	SNIPS	0.6461	0.7164	0.6236	0.5984	0.629	0.6976	0.6046	0.5751
	Direct	0.5362	0.6431	0.5188	0.5315	0.5012	0.6178	0.479	0.4014
	DR	0.6269	0.7008	0.6026	0.5729	0.6207	0.6876	0.5948	0.5228
	ATT	0.602	0.6811	0.5756	0.5985	0.6244	0.6818	0.5976	0.5311
	CVIB	0.6458	0.7102	0.6208	0.5729	0.6324	0.703	0.6057	0.5576
	ACL	0.6609	0.7004	0.6181	0.5914	0.6317	0.6885	0.5998	0.5641
	CBR(new)	0.6773	0.7152	0.6254	0.5942	0.6577	0.6994	0.6147	0.5803
	CBR	0.6788*	0.7344*	0.6401*	0.6223*	0.6735*	0.7318*	0.6362*	0.6092*
PCIC	Base	0.4921	0.801	0.6561	0.641	0.4821	0.8279	0.6802	0.6537
	IPS	0.4808	0.7971	0.6783	0.6302	0.5381	0.8279	0.6788	0.6591
	SNIPS	0.4977	0.7948	0.6835	0.6331	0.5404	0.8664	0.6897	0.6508
	Direct	0.3152	0.7368	0.5526	0.6017	0.3125	0.7317	0.5623	0.6208
	DR	0.4987	0.8294	0.6909	0.6262	0.5139	0.8151	0.688	0.6468
	ATT	0.5053	0.8254	0.6904	0.6207	0.5381	0.8292	0.6938	0.6336
	CVIB	0.5079	0.8352	0.7036	0.6307	0.5443	0.8279	0.6936	0.6441
	ACL	0.4839	0.8122	0.6837	0.6370	0.5631	0.8435	0.7103	0.6566
	CBR(new)	0.4980	0.8433	0.7066	0.6242	0.5619	0.8498	0.7184	0.6640
	CBR	0.5033	0.8408	0.7174*	0.6341	0.5661	0.8305	0.7191*	0.665*
Amazon	Base	0.8287	0.9659	0.5469	0.6441	0.8989	0.9136	0.5643	0.5762
	IPS	0.9344	0.9293	0.5437	0.6189	0.9826	0.9357	0.5721	0.6004
	SNIPS	0.8705	0.9312	0.5395	0.6448	0.9711	0.9333	0.5791	0.5988
	Direct	0.896	0.9379	0.5718	0.6751	0.9605	0.9387	0.552	0.5941
	DR	0.9097	0.9286	0.5467	0.6745	0.9684	0.9367	0.5546	0.6023
	ATT	0.9254	0.9363	0.6173	0.6881	0.9318	0.9453	0.5532	0.6098
	CVIB	0.8810	0.9348	0.5594	0.6783	0.9588	0.9337	0.5647	0.5803
	ACL	0.8676	0.9626	0.5581	0.6444	0.9528	0.9391	0.5681	0.5892
	CBR(new)	0.9217	0.9689	0.6162	0.6545	0.8244	0.9372	0.5653	0.5882
	CBR	0.9345*	0.9777	0.6427*	0.6908*	0.9626	0.9408	0.5813*	0.6061
Anime	Base	0.8828	0.9864	0.5411	0.8781	0.9910	0.9663	0.6072	0.8769
	IPS	0.8890	0.8506	0.5425	0.7919	0.9577	0.8617	0.5651	0.7837
	SNIPS	0.8908	0.8493	0.5314	0.8787	0.9612	0.8627	0.5774	0.7926
	Direct	0.9139	0.8543	0.5872	0.8658	0.963	0.8608	0.5716	0.8285
	DR	0.9001	0.8525	0.5918	0.8526	0.9591	0.8551	0.5669	0.8306
	ATT	0.9117	0.9273	0.5695	0.8284	0.9380	0.9289	0.6142	0.8320
	CVIB	0.8905	0.9228	0.5142	0.8203	0.9343	0.9291	0.5948	0.8316
	ACL	0.8942	0.9636	0.5394	0.8447	0.9528	0.9691	0.6306	0.8499
	CBR(new)	0.9388	0.9684	0.6664	0.8456	0.9565	0.9199	0.5879	0.8154
	CBR	0.9857*	0.9730	0.6833*	0.8828	0.989	0.9937*	0.6215	0.8811

“Base” is the original model. “*” indicates that the improvement of our framework against the best baseline is significant under paired t test with “ $p < 0.05$.”

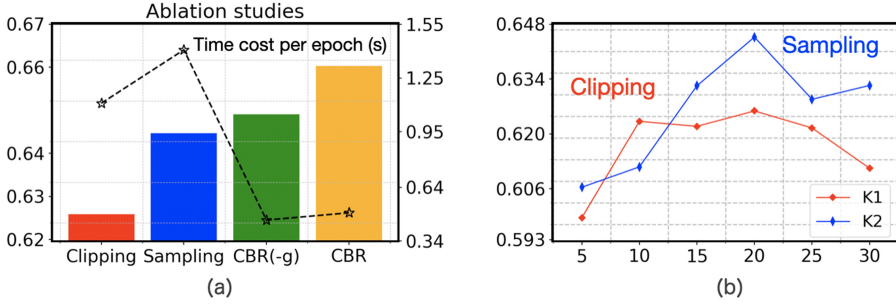


Fig. 2. (a) Performance and time cost of different model variants based on AUC (y-axis). (b) The influences of the selected balancing terms, where the performance is evaluated based on AUC (y-axis), and x-axis represents the number of balancing terms.

- In $CBR(\text{clipping})$, we use the clipping strategy proposed in Section 3.2 to handle the item balancing terms.
- In $CBR(\text{sampling})$, the item balancing terms are optimized based on the sampling strategy.
- In $CBR(-g)$, we do not capture the latent confounders, that is, objective (13) is used for model optimization.

The model parameters are set as their optimal values tuned above. To implement IPM_G for $CBR(\text{clipping})$ and $CBR(\text{sampling})$, we first define $v(i) = \sum_{u \in I_i} g(\phi(u), i)$, where g is a neural network with the same architecture as the discriminator in CBR, and I_i is sampled from the set of users who have interacted with item i . Then, we compute $IPM_G(\{\phi(u_t)\}_{t:i_t=i}, \{\phi(u_t)\}_{t:i_t=i'})$ as $|v(i) - v(i')|$. In the experiments, we report the results based on AUC, MLP, and the dataset of PCIC, while similar conclusions can be drawn for the other metric, base models, and datasets.

The results are presented in Figure 2(a). We find that $CBR(\text{sampling})$ can achieve better performance than $CBR(\text{clipping})$. The reason can be that in $CBR(\text{clipping})$, the balancing terms (i.e., $IPM_G(\cdot; \cdot)$) are fixed in each training epoch. The non-selected terms have no chance to be optimized, which may still reveal important signals. In $CBR(\text{sampling})$, the balancing terms are chosen in a random manner, and every term has an opportunity to be optimized, which can cover more important information. From the efficiency perspective, since $CBR(\text{sampling})$ needs to sample item pairs in each epoch, it costs more time as indicated by the dotted line in Figure 2(a). Careful readers may also be curious about how the number of selected balancing terms (i.e., K_1 for $CBR(\text{clipping})$ and K_2 for $CBR(\text{sampling})$) influence the recommendation performance. To answer this question, we further conduct experiments by tuning K_1 and K_2 in the range of $\{5, 10, 15, 20, 25, 30\}$. The results are presented in Figure 2(b). We can see that the performance is, in general, better when we use larger K_1 or K_2 , which is as expected, since more balancing requirements have been satisfied. Actually, we have also tried to introduce all the balancing terms like objective (11), but it costs too much time (more than 10 hours per epoch) to obtain the results. By considering all the balancing terms in a feasible adversarial manner, CBR can achieve better performance and higher efficiency than both of $CBR(\text{clipping})$ and $CBR(\text{sampling})$.

The modeling of the latent confounders is also important, which is evidenced by the lowered performance of $CBR(-g)$ comparing with CBR. We speculate that in real-world recommendation datasets, latent confounders can be inevitable, since user personalities are too complex to be totally recorded. Without capturing the latent confounders, the causal relation between the user and item is hard to cut down, which may intensify the distribution shift problem and lowers the recommendation performance.

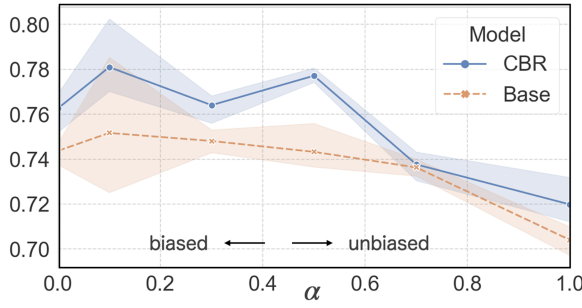


Fig. 3. Performance of our model with different data biases. For each α , we repeat the experiment for three times and report the mean and standard error of the results. In this figure, x-axis is α in Equation (26) and y-axis is AUC score.

5.4 Influence of the Data Bias Severity

In this section, we study how the data bias influences the final recommendation performance. We base this experiment on the simulation dataset, since it allows us to flexibly tune the bias severity by the parameter α . Extremely, $\alpha = 1.0$ means the data is unbiased, and an item is recommended in a completely random manner. If $\alpha = 0.0$, then the data is completely biased by the recommender model. We tune α in the range of $\{0.0, 0.1, 0.3, 0.5, 0.7, 1.0\}$ and set β as 0.5. The model parameters in this experiment are as their optimal values tuned above. Due to the space limitation, we report the results on AUC and MLP, while the conclusions on ACC and GMF are similar and omitted.

The results are presented in Figure 3, from which we can see: Our model can consistently achieve better performance comparing with the base model. This observation manifests that the effectiveness of our model holds for different data biases and suggests that our model can be robust when being applied to different recommendation scenarios. It is interesting to see that the performance gain of our framework is more significant when the data bias is relatively large. The reason can be that, for the base model, the parameters are directly learned based on the biased training sets. They can not perform well on unbiased testing sets. However, by our framework, the learning objective is specially designed to remove the bias effect, which alleviates the distribution shift between the training and testing sets and thus can achieve better performance. In general, the severer the bias is, the larger the performance gap is. Careful readers may find that when $\alpha = 1.0$, there is a small gap between our framework and the base model. The reason can be that, in the synthetic dataset, we have incorporated the latent confounders into the simulation process. In our framework, we explicitly deploy a model to capture such confounders, while in the base model, they are totally ignored, which leads to the lowered performance.

5.5 Influence of the Confounderness

In real-world recommender systems, there are usually many latent confounders, which are either not recorded in the system (e.g., item promotion) or hard to represent clearly (e.g., user emotions). Such confounders can directly influence the debiasing effect. In this section, we study the influence of latent confounders on the recommendation performance. The experiment is based on the simulation dataset, where β is the parameter to control the confounderness of the dataset. Larger β means the dataset is more severely influenced by the latent confounders. We tune β in the range of $\{0.0, 0.3, 0.5, 0.7, 1.0\}$ and set α as 0.5. For the other settings, we follow the above experiment. The results are presented in Figure 4, from which we can see: When the influence of the latent

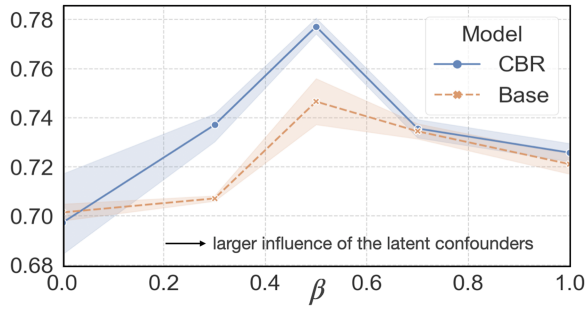


Fig. 4. Performance of our model with different confounderness. For each β , we repeat the experiment for three times and report the mean and standard error of the results. In this figure, x-axis is β in Equation (26) and y-axis is AUC score.

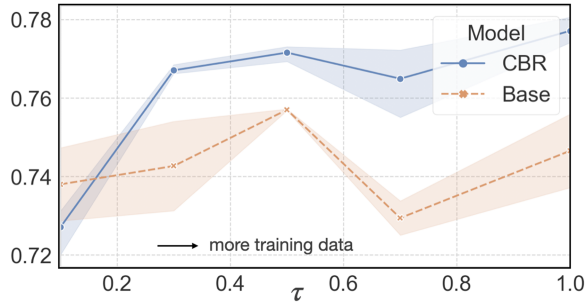


Fig. 5. Performance of our model with different data sparsities based on AUC (y-axis). For each τ , we repeat the experiment for three times and report the mean and standard error of the results.

confounders is very large (e.g., $\beta \geq 0.7$), the performances of our framework and the base model are similar and not satisfied. We speculate that when β is too large, the important **collaborative filtering (CF)** [13] signals are overwhelmed by the latent confounders. Collaborative filtering stands on the key assumption of the recommendation task. Without enough CF signals, neither of the compared models can achieve good performance. When β falls into a reasonable range (e.g., $\beta \in [0.3, 0.5]$), we can observe significant improvement of our framework comparing with the base model. This observation demonstrates the effectiveness of our deconfounder idea and suggests that our framework can be competitive for practical recommendation scenarios, which usually contain latent confounders.

5.6 Influence of the Data Sparsity

In this section, we study the influence of data sparsity on the recommendation performance. The settings of this experiment are similar to the above experiment, and the synthetic dataset is generated by setting α and β as 0.5. We use parts of the original training set for model optimization, and the testing set remains to be unbiased. The ratio τ between the selected and original number of samples is tuned in $\{0.1, 0.3, 0.5, 0.7, 1.0\}$, where $\tau = 1.0$ means incorporating all the training samples. The results are presented in Figure 5, from which we can see: As we use more training samples, the performance of our framework continually goes up, which agrees with the learning theory that more samples can lower the bound between the empirical and expectation loss functions. When the number of samples is relatively small (e.g., $\tau \leq 0.5$), all the models exhibit similar

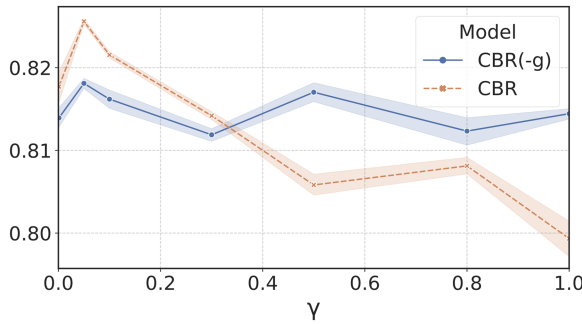


Fig. 6. Performance of our model with different γ 's, where we use AUC (y-axis) as the evaluation metric.

worse performances. As the old saying goes, “even a clever housewife cannot cook a meal without rice.” Without a basic amount of training samples, any model cannot perform well. When the dataset becomes denser, the data bias comes to the major issue. At this time, our framework can achieve much larger performance gains as compared with the base model.

5.7 Influence of the Balancing Parameter γ

In our model, γ determines the importance of different optimization targets. In this section, we analyze the influence of γ on the final recommendation performance. More specifically, we tune γ in the range of $[0.001, 0.05, 0.1, 0.3, 0.5, 0.8, 1]$, and the other settings follow the above experiments. We report the performance of CBR(-g) (corresponding to Equation (13)) and CBR (corresponding to Equation (22)) in Figure 6. We can see: In the beginning, the performance rises up as we increase γ within a small range. However, after reaching the optimal points, the performance tends to be lowered by larger γ 's. These results suggest that γ should be specified with a small value, which is not surprising, since the performance is mainly determined by the recommendation loss for modeling user implicit feedback. At the same time, we cannot ignore the balancing term, which is evidenced by the lowered performance when γ is too small (e.g., 0.001).

6 CONCLUSION, LIMITATION, AND FUTURE WORK

In this article, we propose a novel debiased recommender model based on confounder balancing. We begin from the ideal unbiased learning objective and derive its upper bound under the effect of a user representative function. To enhance the training efficiency, we design three methods to handle the large number of item balancing terms. We also model the potential latent confounders, which can lead to more robust optimization.

This article actually opens a new door for debiased recommendation. However, there are still many limitations, which leave much room for improvement. To begin with, we only empirically demonstrate that our implementation inspired by theory 1 works well. However, we do not theoretically check whether the derived upper bound is tight enough. In the future, we would like to develop a more solid theoretical foundation on the upper bound in theory 1. And then, in practical recommender systems, the latent confounders can be complex and structured, and a single embedding can be hard to represent them accurately. As a future work, we plan to use more advanced technique like graph neural network to model the latent confounders. At last, in our model, the dynamic user preferences are ignored. An interesting direction is to extend our model to the time-aware or sequential settings.

ACKNOWLEDGMENT

We appreciate the support from Mindspore⁹, which is a new deep learning computing framework.

REFERENCES

- [1] Ioana Bica, Ahmed M. Alaa, and Mihaela van der Schaar. 2020. Time series deconfounder: Estimating treatment effects over time in the presence of hidden confounders. In *ICML (Proceedings of Machine Learning Research)*, Vol. 119. PMLR, 884–895. Retrieved from <http://proceedings.mlr.press/v119/bica20a.html>.
- [2] Jiawei Chen, Hande Dong, Yang Qiu, Xiangnan He, Xin Xin, Liang Chen, Guli Lin, and Keping Yang. 2021. AutoDebias: Learning to debias for recommendation. *arXiv preprint arXiv:2105.04170* (2021).
- [3] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and debias in recommender system: A survey and future directions. *arXiv preprint arXiv:2010.03240* (2020).
- [4] Long Chen, Hanwang Zhang, Jun Xiao, Xiangnan He, Shiliang Pu, and Shih-Fu Chang. 2019. Counterfactual critic multi-agent training for scene graph generation. In *ICCV*. 4613–4623.
- [5] Quanyu Dai, Zhenhua Dong, and Xu Chen. 2022. Debiased recommendation with neural stratification. *AI Open* 3 (2022), 213–217.
- [6] Jingtao Ding, Yuhua Quan, Xiangnan He, Yong Li, and Depeng Jin. 2019. Reinforced negative sampling for recommendation with exposure data. In *IJCAI*. 2230–2236.
- [7] Sihao Ding, Fuli Feng, Xiangnan He, Yong Liao, Jun Shi, and Yongdong Zhang. 2021. Causal incremental graph convolution for recommender system retraining. *arXiv preprint arXiv:2108.06889* (2021).
- [8] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly robust policy evaluation and learning. *arXiv preprint arXiv:1103.4601* (2011).
- [9] Bent Fuglede and Flemming Topsøe. 2004. Jensen-Shannon divergence and Hilbert space embedding. In *ISIT*. IEEE.
- [10] Xiang Gao, Meera Sitharam, and Adrian E. Roitberg. 2017. Bounds on the Jensen gap, and implications for mean-concentrated distributions. *arXiv preprint arXiv:1712.05267* (2017).
- [11] Asela Gunawardana and Guy Shani. 2009. A survey of accuracy evaluation metrics of recommendation tasks. *J. Mach. Learn. Res.* 10, 12 (2009).
- [12] Ruocheng Guo, Jundong Li, and Huan Liu. 2020. Learning individual causal effects from networked observational data. In *WSDM*. 232–240.
- [13] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer product-based neural collaborative filtering. *arXiv preprint arXiv:1808.03912* (2018).
- [14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.
- [15] Fredrik Johansson, Uri Shalit, and David Sontag. 2016. Learning representations for counterfactual inference. In *ICML*. PMLR, 3020–3029.
- [16] Fredrik D. Johansson, Uri Shalit, Nathan Kallus, and David A. Sontag. 2020. Generalization bounds and representation learning for estimation of potential outcomes and causal effects. *CoRR* abs/2001.07426 (2020).
- [17] Fredrik D. Johansson, Uri Shalit, and David A. Sontag. 2016. Learning representations for counterfactual inference. In *ICML (JMLR Workshop and Conference Proceedings)*, Vol. 48. JMLR.org, 3020–3029. Retrieved from <http://proceedings.mlr.press/v48/johansson16.html>.
- [18] Carolin Lawrence, Artem Sokolov, and Stefan Riezler. 2017. Counterfactual learning from bandit feedback under deterministic logging: A case study in statistical machine translation. In *EMNLP*.
- [19] Byung-Jun Lee, Jongmin Lee, and Kee-Eung Kim. 2021. Representation balancing offline model-based reinforcement learning. In *ICLR*. OpenReview.net. Retrieved from https://openreview.net/forum?id=QpNz8r_Ri2Y.
- [20] Yao Liu, Omer Gottesman, Aniruddh Raghu, Matthieu Komorowski, Aldo A. Faisal, Finale Doshi-Velez, and Emma Brunskill. 2018. Representation balancing MDPs for off-policy policy evaluation. In *NIPS*. 2649–2658. Retrieved from <https://proceedings.neurips.cc/paper/2018/hash/980ecd059122ce2e50136bda65c25e07-Abstract.html>.
- [21] Jing Ma, Ruocheng Guo, Chen Chen, Aidong Zhang, and Jundong Li. 2021. Deconfounding with networked observational data in a dynamic environment. In *WSDM*. 166–174.
- [22] Benjamin M. Marlin, Richard S. Zemel, Sam T. Roweis, and Malcolm Slaney. 2012. Collaborative filtering and the missing at random assumption. *CoRR* abs/1206.5267 (2012).
- [23] Yusuke Narita, Shota Yasui, and Kohei Yata. 2019. Efficient counterfactual learning from bandit feedback. In *AAAI*. AAAI Press, 4634–4641. DOI : <https://doi.org/10.1609/aaai.v33i01.33014634>
- [24] Judea Pearl. 2009. *Causality*. Cambridge University Press.

⁹<https://www.mindspore.cn>.

- [25] Rajesh Ranganath and Adler Perotte. 2018. Multiple causal inference with latent confounding. *arXiv preprint arXiv:1805.08273* (2018).
- [26] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).
- [27] Yuta Saito. 2020. Asymmetric tri-training for debiasing missing-not-at-random explicit feedback. In *SIGIR*. 309–318.
- [28] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased recommender learning from missing-not-at-random implicit feedback. In *WSDM*. 501–509.
- [29] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. In *ICML (JMLR Workshop and Conference Proceedings)*, Vol. 48. JMLR.org, 1670–1679. Retrieved from <http://proceedings.mlr.press/v48/schnabel16.html>.
- [30] Uri Shalit, Fredrik D. Johansson, and David Sontag. 2017. Estimating individual treatment effect: Generalization bounds and algorithms. In *ICML*. PMLR, 3076–3085.
- [31] Bharath Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, and Gert Lanckriet. 2009. On integral probability metrics, divergences and binary classification. *arXiv preprint arXiv:0901.2698* (2009).
- [32] Adith Swaminathan and Thorsten Joachims. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *J. Mach. Learn. Res.* 16 (2015), 1731–1755. Retrieved from <http://dl.acm.org/citation.cfm?id=2886805>.
- [33] Adith Swaminathan and Thorsten Joachims. 2015. Counterfactual risk minimization: Learning from logged bandit feedback. *CoRR* abs/1502.02362 (2015).
- [34] Adith Swaminathan and Thorsten Joachims. 2015. The self-normalized estimator for counterfactual learning. In *NIPS*. 3231–3239. Retrieved from <https://proceedings.neurips.cc/paper/2015/hash/39027dfad5138c9ca0c474d71db915c3-Abstract.html>.
- [35] Wenjie Wang, Fuli Feng, Xiangnan He, Xiang Wang, and Tat-Seng Chua. 2021. Deconfounded recommendation for alleviating bias amplification. *arXiv preprint arXiv:2105.10648* (2021).
- [36] Wenjie Wang, Fuli Feng, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2021. Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue. In *SIGIR*. 1288–1297.
- [37] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position bias estimation for unbiased learning to rank in personal search. In *WSDM*. ACM, 610–618. DOI : <https://doi.org/10.1145/3159652.3159732>
- [38] Yixin Wang and David M. Blei. 2019. The blessings of multiple causes. *J. Amer. Statist. Assoc.* 114, 528 (2019), 1574–1596.
- [39] Yu Wang, Xin Xin, Zaiqiao Meng, Xiangnan He, Joemon Jose, and Fuli Feng. 2021. Probabilistic and variational recommendation denoising. *arXiv preprint arXiv:2105.09605* (2021).
- [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. In *NeurIPS*. Retrieved from <https://proceedings.neurips.cc/paper/2020/hash/13f3cf8c531952d72e5847c4183e6910-Abstract.html>.
- [41] Zhenlei Wang, Shiqi Shen, Zhipeng Wang, Bo Chen, Xu Chen, and Ji-Rong Wen. 2022. Unbiased sequential recommendation with latent confounders. In *WWW*. 2195–2204.
- [42] 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 *SIGIR*. 1791–1800.
- [43] Chen Xu, Jun Xu, Xu Chen, Zhenghua Dong, and Ji-Rong Wen. 2022. Dually enhanced propensity score estimation in sequential recommendation. In *CIKM*. 2260–2269.
- [44] Da Xu, Chuanwei Ruan, Evren Korpoglu, Sushant Kumar, and Kannan Achan. 2020. Adversarial counterfactual learning and evaluation for recommender system. *Adv. Neural Inf. Process. Syst.* 33 (2020), 13515–13526.
- [45] Liuyi Yao, Zhixuan Chu, Sheng Li, Yaliang Li, Jing Gao, and Aidong Zhang. 2020. A survey on causal inference. *arXiv preprint arXiv:2002.02770* (2020).
- [46] Bo-Wen 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 *CIKM*. ACM, 329–338. DOI : <https://doi.org/10.1145/3357384.3358058>
- [47] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. 2021. Causal intervention for leveraging popularity bias in recommendation. *arXiv preprint arXiv:2105.06067* (2021).
- [48] Zihao Zhao, Jiawei Chen, Sheng Zhou, Xiangnan He, Xuezhi Cao, Fuzheng Zhang, and Wei Wu. 2021. Popularity bias is not always evil: Disentangling benign and harmful bias for recommendation. *arXiv preprint arXiv:2109.07946* (2021).
- [49] 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 *WWW*. 2980–2991.

- [50] Hao Zou, Kun Kuang, Boqi Chen, Peixuan Chen, and Peng Cui. 2019. Focused context balancing for robust offline policy evaluation. In *KDD*. ACM, 696–704. DOI : <https://doi.org/10.1145/3292500.3330852>
- [51] Hao Zou, Kun Kuang, Boqi Chen, Peixuan Chen, and Peng Cui. 2019. Focused context balancing for robust offline policy evaluation. In *SIGIR*. 696–704.

Received 27 February 2022; revised 9 October 2022; accepted 6 January 2023