Invariant Preference Learning for General Debiasing in Recommendation

Zimu Wang
Tsinghua University, Beijing, China
14317593@qq.com

Yue He
Tsinghua University, Beijing, China
heyue18@mails.tsinghua.edu.cn

Jiashuo Liu
Tsinghua University, Beijing, China
liujiashuo77@gmail.com

Wenchao Zou
Siemens China, Shanghai, China
wenchao.zou@siemens.com

Philip S. Yu
University of Illinois at Chicago, Chicago, USA
psyu@cs.uic.edu

Peng Cui†
Tsinghua University, Beijing, China
cuip@tsinghua.edu.cn

ABSTRACT

Current recommender systems have achieved great successes in online services, such as E-commerce and social media. However, they still suffer from the performance degradation in real scenarios, because various biases always occur in the generation process of user behaviors. Despite the recent development of addressing some specific type of bias, a variety of data bias, some of which are even unknown, are often mixed up in real applications. Although the uniform (or unbiased) data may help for the purpose of general debiasing, such data can either be hardly available or induce high experimental cost. In this paper, we consider a more practical setting where we aim to conduct general debiasing with the biased observational data alone. We assume that the observational user behaviors are determined by invariant preference (i.e. a user's true preference) and the variant preference (affected by some unobserved confounders). We propose a novel recommendation framework called InvPref which iteratively decomposes the invariant preference and variant preference from biased observational user behaviors by estimating heterogeneous environments corresponding to different types of latent bias. Extensive experiments, including the settings of general debiasing and specific debiasing, verify the advantages of our method.

CCS CONCEPTS

• **Information systems → World Wide Web**.

KEYWORDS

Recommender system, General debiasing, Invariant preference

ACM Reference Format:
Zimu Wang, Yue He, Jiashuo Liu, Wenchao Zou, Philip S. Yu, and Peng Cui†. 2022. Invariant Preference Learning for General Debiasing in Recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3534678.3539439

†Corresponding Author

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08…$15.00
https://doi.org/10.1145/3534678.3539439

1 INTRODUCTION

Recommender systems play a vital role in alleviating information explosion on the web [17, 38]. The research community have proposed plenty of recommendation techniques [33]. In recent years, further efforts are devoted to integrate the powerful expression ability of deep neural networks (DNN) [3] and graph convolution networks (GCN) [16] into recommendation frameworks. Despite their promising performances, real-world recommender systems usually suffer from various types of biases that occur in the generation process of user behaviors. For example, user behaviors are easily impacted by the exposure policy of a recommender system (i.e. exposure bias [20]) or other users' actions (i.e. conformity bias [34]); popular items may have more chances to be clicked (i.e. popularity bias [13]); the demographic, spatial and temporal heterogeneity in population induces the shift of user or item distributions (i.e. out-of-distribution recommendation[10]). The existing recommender systems tend to unconsciously remember the variant patterns from the biased observational data in the training phase, and hence perform poorly when deployed in a testing environment where the learned biased patterns produce negative effects.

In recent years, debiasing has become a research foci in the field of recommendation. A variety of methods are proposed to address a specific type of bias. For example, [28] reweights samples using the frequency of items to balance the popularity; a data imputation method [26] estimates the pseudo-labels for user-item interaction to alleviate the problem of data missing not at random; [40] utilizes expert knowledge to model the generation mechanism of specific bias and remove its effect consequently; etc. However, various types of data biases are often mixed up in real scenarios (e.g. a user's purchase behavior may be influenced by both popularity bias and exposure bias). In addition, some unknown types of bias that are difficult to predefine indeed exist in the complex feedback loop of recommendation. Therefore, a more pragmatic setting is general debiasing, aiming to achieve a model against multiple types of (known or unknown) biases simultaneously.

With the goal of general debiasing, a stream of methods introduces the unbiased uniform data which is collected by a random logging policy, to supervise the training of models on biased data. To name afew, knowledge distillation methods [5, 7, 21] train a teacher model on the uniform dataset and then use it to guide the base model trained on biased dataset; IPS-based methods [28] reweights the samples from biased distribution to the unbiased distribution by the estimated inverse propensity score (IPS) using auxiliary uniform data. However, such uniform data can either be unavailable or induce high cost through experimental interventions. Some recent

research attempt to deal with the general debiasing problem with observational biased data alone [22, 36], but they require that all the information determining user behaviors are observed, which is often infeasible due to the existence of unobserved confounders[1] [29]. For example, marketing policy apparently influences the user preference to an item, but it is usually not encoded in user or item profiles. Hence, it is demanded to address the latent bias induced by unobserved confounders for general debiasing.

In this paper, we provide a generic conceptual framework of data generation in recommendation from a view of causal graph (as shown in Figure 1), where the observational user behaviors are caused by invariant preference (i.e. the true preference) and variant preference affected by latent confounders. Motivated by the recent progress on invariant learning [4, 15, 24], we capture the unobserved confounders through estimating pseudo environments labels as the agents, and propose a novel Invariant Preference Learning (InvPref) framework to implement the conceptual framework. Specifically, we estimate the pseudo labels of heterogeneous environments via clustering in the user-item representation space, and differentiate the invariant preference (i.e. environment-irrelevant preference) and the variant preference (i.e. environment-relevant preference) from observational behaviors in a way of adversarial learning. As the invariant/variant preference learning step and the environment estimation step can promote each other, we design an iterative optimization process in our method. Extensive experimental results from real-world datasets clearly demonstrate that our method outperforms the benchmark models significantly in multiple biased recommendation scenarios. Our implementation can be found at https://github.com/AIflowerQ/InvPref_KDD_2022.

The main contributions of this paper are as follows:

- We investigate the new problem of general debiasing in recommender systems when only the biased training data with latent bias is available.
- We provide a generic conceptual framework of data generation in recommendation from a view of causal graph, where the latent bias are described as latent confounding.
- We propose a novel Invariant Preference Learning (InvPref) method which iteratively differentiate the invariant preference and variant preference from observational biased behaviors.
- We conduct extensive experiments, including general debias and specific debias scenarios, to verify the advantages of our proposal.

2 RELATED WORK

2.1 Debiasing in Recommender

As academia and industry gradually pay attention to the bias problem in recommendation scenarios, more and more debiasing methods have been proposed. These methods can be divided into two streams. The first stream of methods is designed for specified biases based on expert knowledge: 1) IPS based method[27, 39]. This class of methods performs debiasing by reweighting the samples. For example, [13] reweights each sample based on the popularity of the items, thereby mitigating the problem of popularity bias; [27] derives an unbiased estimate of binary feedback modeling for recommender systems, based on the idea of positive unlabeled learning, so that it can reduce the popularity and exposure biases. 2) Data Imputation[11, 25]. These methods estimate pseudo-labels for missing interactions to solve the Missing-Not-At-Random problem. [11] based on the assumption that the probability of users' selection on items depends on users' rating values for that item, jointly modeling rating and missing data mechanism to alleviate the selection bias. 3) Generative modeling[37, 40]. Such methods model the mechanism of specific biases generation based on expert knowledge and eliminate the influence of bias in the inference stage. For example, [32] models the causes of click bias in news recommendation scenarios. In real production, there are often multiple or even unknown biases in the training data, which makes it impractical to design a dedicated debiasing algorithm for each bias. To make matters worse, different types of biases can be mixed. This makes this stream of methods potentially difficult to deal with real-world scenarios.

The second stream of methods is called unbiased data augmentation methods. This stream of methods introduces an unbiased uniform dataset (collected by a random logging policy)[28] to supervise the training process of the model on biased data. There are currently two main research lines for this type of method: 1)Knowledge Distillation [5, 7, 21]. This line of methods train a teacher model on an unbiased uniform dataset to guide the training process on biased data. For example, [7] uses a uniform data set to optimize the debiasing parameters by solving the double-layer optimization problem with meta-learning technique; [21] and [5] train a teacher model same with the base model on the uniform dataset to guide the training of the base model. 2) IPS-based[27, 28]. This line of methods study how to make the biased distribution of the training data approximate the unbiased distribution of the uniform data by sample reweighting. For example, [28] estimates the IPS for each sample based on Naive Bayes using a small amount of uniform data. However, the strategy of random recommendation can seriously affect the user experience and may require large costs, and sufficient scale and reliable unified data are often inaccessible.

At present, there are some researches on using Information Bottleneck[31] to solve the problem of General Debiasing without uniform data[22, 36]. These methods model the biases introduced by the recommender policy itself based on causal graph, and use Information Bottleneck technique to learn latent vector without the information of recommender policy. These methods lack modeling the influence of side information (e.g. season and public opinion) on user preference. However, in real scenarios, there are various sources of biases, and factors such as seasons and public opinion will cause large biases in the user's preference.

2.2 Invariant Learning

Invariant learning[4, 6, 24] assumes that there is heterogeneity in observed data, that is, observed data originate from multiple different environments. There are differences in the distribution of data in environments. The goal of Invariant learning is to capture representations with invariant predictive ability across environments. [18] theoretically and systematically analyzes the assumption strength of existing invariant learning methods, and theoretically relaxes the

[1] A confounder can be regarded as a variable corresponding to a source of bias.

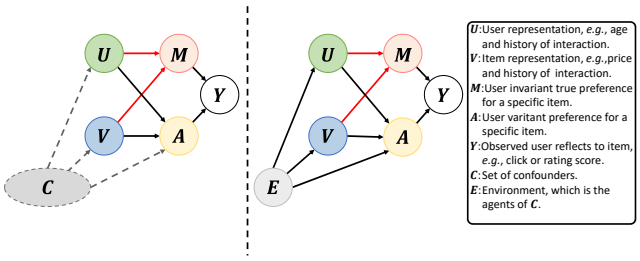


Figure 1: Causal Graph View of Recommendation Bias

previous invariance assumption and proposes a corresponding solution. [2] prove that a form of the information bottleneck constraint along with invariance helps address key failures of IRM[4]-based methods, and propose a solution combining invariant learning and information bottleneck. [23] propose a practical and easy-to-implement weighting method to capture invariance for better generalization. [15] heuristically analyzes IRM failures and proposes an alternative invariance penalty by revisiting the Gramian matrix of the data representation.

3 METHOD

In this section, we first analyze the data generation in recommendation scenarios where bias occurs owing to the confounders, from a view of causal graph. Then we will introduce the details of our proposed framework Invariant Preference Learning, which utilizes the environment as the agents of confounders.

3.1 Causal View of Bias in Recommendation

To study the general biasing problem, we define a causal graph to describe the generation of observational user behaviors in Figure 1. Note that we use capital letters (eg, **U**), lowercase letters (eg, u), and calligraphic font letters (eg, $\mathcal{U}$) to denote a variable, its specific value, and sample space, respectively. In particular,

- **U** represents the user node, which contains the user's behavior history and profile (e.g. age and occupation).
- **V** represents the item node, which contains the history of feedback from users and the profile (e.g. price and category).
- **M** represents the invariant true preference of user for item, and the preference is a representation.
- **A** represents the variant user preference for the item affected by latent confounders.
- **Y** represents the user feedback on the item in the observational data.
- **C** represents the set of confounders that may cause biases in the observational data.
- **E** represents the environment which is the agents for **C**.

The distribution of observational data $P(Y,U,V) = P(Y|U,V) \cdot P(U,V)$, where $P(Y|U,V)$ denotes the generation mechanism of users' feedback Y. The estimation of Y is always biased because $P(Y|U,V)$ is inevitably affected by the confounders C in recommendation scenario, e.g. the marketing policy or recommender system itself. However, the regular patterns in user behaviors imply that it naturally assumes users' feedback is determined by an invariant truth preference M (i.e. $M \perp\!\!\!\perp C|U,V$), and another variant preference A reflecting his/her received disturbance from confounders. As a result, we can generally formulate the observational user behaviors as in Figure 1. Then the confounders C will jointly influence user U, item V, and variant preference A. To address the confounders, especially the unobserved ones, it is needed to employ agents of them from observational data. Here, we make the Assumption 1 as follows.

Table 1: Notation and Definitions

Notation	Annotation
D	The training dataset.
L	Size of embedding.
u, v, e	The label of user, item and environment.
$\mathbf{s}, \mathbf{t}, \mathbf{q} \in \mathbb{R}^L$	The embedding of user, item and environment.
$\mathbf{S}, \mathbf{T}, \mathbf{Q}$	The embedding set of user, item and environment.
$y_{u,v}$	The truth feedback in observed data.
$\mathbf{m}, \mathbf{a} \in \mathbb{R}^L$	The invariant and variant preference.
$P_e(Y\|U,V)$	The condition distribution of environment e.

Assumption 1. *The training data $D_e := \{(y_j^e, u_j^e, v_j^e)\}_{j=1}^{n_e}$ is collected from heterogeneous environments $e \in \mathcal{E}$. The sample $\{(y_j^e, u_j^e, v_j^e)\}$ from the environment e follows the distribution $P_e(Y,U,V)$ of environment e. $\forall e_1, e_2 \in \mathcal{E}$, $e_1 \neq e_2 \Leftrightarrow P_{e_1}(Y,U,V) \neq P_{e_2}(Y,U,V)$. And for any environment $e \in \mathcal{E}$, environment e contains bias $\Leftrightarrow$ $P_e(Y,U,V) \neq P_\mu(Y,U,V)$, where $P_\mu(Y,U,V)$ is the ideal unbiased data distribution.*

The heterogeneity of environment can regared to be caused by confounders. As a result, we can directly use it as the agents of confounders, and make some modifications to the original graph. We changed the original C to the E for the environment. As a result, we propose a novel Invariant Preference Learning framework according to the causal graph we defined, to solve the general debiasing problem.

Problem 1 (General Debiasing Problem). *Given heterogeneous training data $D = \{D_e\}$ collected from multiple environments $e \in \mathcal{E}$ without explicit labels, the task is to exploit the latent heterogeneity inside data and capture the invariant preference for general debiasing.*

3.2 Framework

In order to address the General Debiasing problem in recommendation, we propose a general debiasing framework named InvPref, which consists of two interactive stages as follows:

- **Environment Inference** In order to exploit the latent heterogeneity inside data, InvPref proposes to generate environments with a clustering algorithm based on $p_e(y|u,v)$. Specifically, given a (u,v) pair, we infer its feedback $\hat{y}_{u,v,e}$ under each environment e and select the environment $\hat{e}_{u,v}$ corresponding to the result closest to $y_{u,v}$, which is the true feedback of (u,v).
- **Invariant Preference Learning** Given the learned environments, we capture invariant and variant preference via adversarial learning. Specifically, we use different embeddings to capture the invariant preference, variant preference

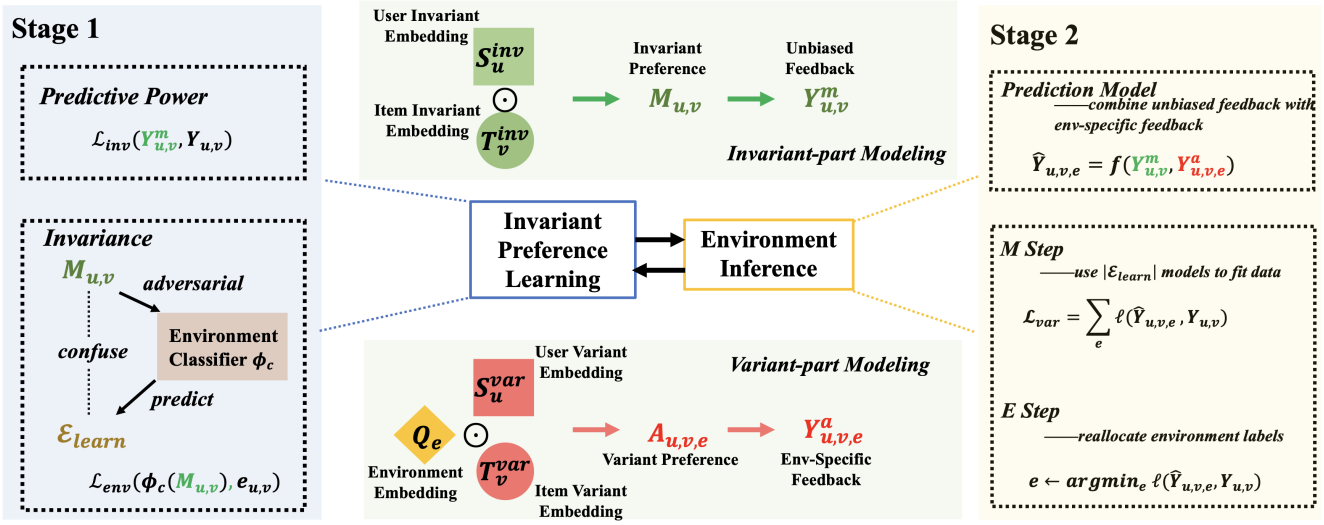


Figure 2: The Framework of InvPref.

and latent environments respectively. To learn discriminative and invariant true user preference across multiple environments, InvPref jointly optimizes the recommendation task and an environment classifier which is only used in the training phase.

InvPref alternates the above two processes, making them reciprocally resonate. In testing, we only use the learned invariant preference to make prediction, which can greatly resist the influence brought by potential biases. For clarity, we will introduce the Invariant Preference Learning stage and Environment Inference stage respectively in the following sections.

3.3 Invariant Preference Learning

We use $\mathbf{s}^{inv}_u \in \mathbb{R}^L$ and $\mathbf{t}^{inv}_v \in \mathbb{R}^L$ to denote the invariant embeddings of user u and item v, respectively. Given u and v as inputs, the invariant preference $\mathbf{m}_{u,v}$ of user u for item v is modeled as the Hadamard product of user's invariant preference embedding $\mathbf{s}^{inv}_u$ and item's invariant embedding $\mathbf{t}^{inv}_v$:

$$\mathbf{m}_{u,v} = \mathbf{s}^{inv}_u \odot \mathbf{t}^{inv}_v \tag{1}$$

which characterizes the invariant interaction between the user u and item v. With the invariant preference $\mathbf{m}_{u,v}$, we can calculate the feedback $y^m_{u,v}$ of user u to item v, which only depends on the invariant preference:

$$y^m_{u,v} = \phi_r(\mathbf{m}_{u,v}), \tag{2}$$

where $\phi_r(\cdot)$ denotes the feedback predicting function and takes different forms in explicit and implicit feedback:

$$\phi_r(\mathbf{x}) = \begin{cases} \mathbf{x}^T \mathbf{1}_L & \text{for explicit feedback;} \\ \text{Sigmoid}(\mathbf{x}^T \mathbf{1}_L) & \text{for implicit feedback.} \end{cases} \tag{3}$$

To capture discriminative and invariant true user preference across multiple environments, we need to simultaneously maximize the predictive power of $\mathbf{M}$ to $\mathbf{Y}$ and minimize the environment information in $\mathbf{M}$.

(1) In order to optimize the predictive ability of $\mathbf{M}$ to $\mathbf{Y}$ and facilitate the use of $\mathbf{Y}_m$ for specific recommendation tasks (e.g. rank and rate predict), InvPref uses the observed feedback $\mathbf{Y}$ as supervision information to optimize the loss term

$$\mathcal{L}_{inv} = \frac{1}{|D|} \sum_{(u,v)\in D} \ell_{rec}(y^m_{u,v}, y_{u,v}), \tag{4}$$

where ℓ_{rec} denotes the loss function in recommendation task, and we use mean square error (MSE) for explicit feedback and binary cross entropy loss (BCE) for implicit feedback.

(2) In order to guarantee the invariance across environments, we need to filter environment information. In other words, invariant preference should be able to confuse the classifier so that a well-trained classifier cannot predict the environment label of the sample. Inspired by Domain Generalization methods[8, 9, 19], we adopt the domain adversarial learning.

We define a classifier ϕ_c parameterized by θ_c that identifies sample environment labels using invariant preference and uses cross entropy loss which is denoted as CE to measure the quality of the classification.

$$\mathcal{L}_{env} = \frac{1}{|D|} \sum_{(u,v)\in D} \text{CE}(\phi_c(\mathbf{m}_{u,v}), e_{u,v}) \tag{5}$$

The goal of the classifier is to identify the environment labels of the samples, i.e., to minimize the $\mathcal{L}_{env}$, while the goal of $\mathbf{m}_{u,v}$ is to confuse the classifier, i.e., to maximize the $\mathcal{L}_{env}$. In summary, the goal of adversarial learning is

$$\min_{\phi_c} \max_{\mathbf{S}^{inv}, \mathbf{T}^{inv}} \mathcal{L}_{env} \tag{6}$$

where we maximize the $\mathcal{L}_{env}$ with respect to $\mathbf{S}^{inv}$ and $\mathbf{T}^{inv}$ since $\mathbf{M} = \mathbf{S}^{inv} \odot \mathbf{T}^{inv}$.

In summary, the overall objective function of our invariant preference learning stage is:

$$\mathcal{L}_{pref} = \mathcal{L}_{inv} - \alpha \cdot \mathcal{L}_{env} \tag{7}$$

Algorithm 1 Invariant Preference Learning (InvPref)

Input: Training dataset D consist of (u, v) pairs.
Output: Invariant user embeddings $\mathbf{S}^{inv}$ and item embeddings $\mathbf{T}^{inv}$.
Initialize $\mathbf{S}^{inv}, \mathbf{T}^{inv}, \mathbf{S}^{var}, \mathbf{T}^{var}, \mathbf{Q}, \theta_c$.
Random assign environment label for each (u, v).
while not converged **do**
  Stage 1: Invariant Preference Learning: Update $\mathbf{S}^{inv}$, $\mathbf{T}^{inv}$, θ_c according to Equation10 $\sim$ 12.
  Stage 2: Environment Inference:
    M Step: Fit models according to Equation 16, update $\mathbf{S}^{inv}, \mathbf{T}^{inv}, \mathbf{S}^{var}, \mathbf{T}^{var}, \mathbf{Q}$.
    E Step·: Reallocate environment labels according to Equation 17.
end while
return: $\mathbf{S}^{inv}$and $\mathbf{T}^{inv}$.

where α is the hyper-parameter to balance the two loss. We aim to find the saddle point $\theta_c, \mathbf{S}, \mathbf{T}$ such that

$$(\mathbf{S}, \mathbf{T}) = \arg\min\{\mathcal{L}_{inv} - \alpha \cdot \mathcal{L}_{env}\}, \tag{8}$$

$$\theta_c = \arg\min \mathcal{L}_{env}. \tag{9}$$

which can be obtained with the following gradient updates:

$$\mathbf{S} \leftarrow \mathbf{S} - \mu\left(\frac{\partial \mathcal{L}_{inv}}{\partial \mathbf{S}} - \alpha \frac{\partial \mathcal{L}_{env}}{\partial \mathbf{S}}\right), \tag{10}$$

$$\mathbf{T} \leftarrow \mathbf{T} - \mu\left(\frac{\partial \mathcal{L}_{inv}}{\partial \mathbf{T}} - \alpha \frac{\partial \mathcal{L}_{env}}{\partial \mathbf{T}}\right), \tag{11}$$

$$\theta_c \leftarrow \theta_c - \mu\alpha \frac{\partial \mathcal{L}_{env}}{\partial \theta_c}, \tag{12}$$

where μ is the learning rate.

In wide scenarios, it is convenient to update stationary points with the help of gradient descent optimizers which are provided by many mature machine learning frameworks. However, it is infeasible to directly update Equations(10-12) by gradient descent algorithms. There is a little but important difference between them and jointly optimizing a recommender model and an environment classifier by gradient descent, that the gradients for recommendation and domain prediction are subtracted, not summed.

Inspired by [9], we introduce the gradient reversal layer(GRL) between $\mathbf{M}$ and ϕ_c to make it is possible to update Equations(10-12) by using gradient descent optimizers.

3.4 Environment Inference

In order to exploit the latent heterogeneity, InvPref proposes to generate environments based on $p_e(y|u, v)$, for which we design an EM-based clustering algorithm. The clustering algorithm consists of two steps. In M step, we use multiple models to fit the data from the corresponding environment, and then in E step, we reallocate the environment labels according to learned models. For simplicity, the environment label $e_{u,v}$ of sample (u, v) is abbreviated as e.

(1) Prediction Model

We use $\mathbf{s}_u^{var} \in \mathbb{R}^L$ and $\mathbf{t}_v^{var} \in \mathbb{R}^L$ to denote the variant embeddings of user u and item v respectively, for capturing variant preferences which affected by environment. For environment e we use $\mathbf{q}_e \in \mathbb{R}^L$ to denote its corresponding embedding. Given u and v as inputs, the variant preference $\mathbf{a}_{u,v,e}$ of user u for item v in environment e is defined as the Hadamard product of $\mathbf{s}_u^{var}$, $\mathbf{t}_v^{var}$ and $\mathbf{q}_e$:

$$\mathbf{a}_{u,v,e} = \mathbf{s}_u^{var} \odot \mathbf{t}_v^{var} \odot \mathbf{q}_e \tag{13}$$

which models the variant interaction between user, item and environment.

Similar to the calculation of $y_{u,v}^m$ which is the feedback decided by invariant preference $m_{u,v}$, we can calculate the user's feedback $y_{u,v,e}^a$ which is only decided by variant preference $\mathbf{a}_{u,v,e}$.

$$y_{u,v,e}^a = \phi_r(\mathbf{a}_{u,v,e}) = \phi_r(\mathbf{s}_u^{var} \odot \mathbf{t}_v^{var} \odot \mathbf{q}_e). \tag{14}$$

Since the feedback $y_{u,v,e}$ in the observed data is affected by both $M_{u,v}$ and $A_{u,v,e}$, we construct a fusion function $f(\cdot, \cdot)$ to predict $\hat{y}_{u,v,e}$ by $y_{u,v}^m$ and $y_{u,v,e}^a$

$$\hat{y}_{u,v,e} = f(y_{u,v}^m, y_{u,v,e}^a) = \begin{cases} y_{u,v}^m + y_{u,v,e}^a & \text{explicit feedback;} \\ y_{u,v}^m \times y_{u,v,e}^a & \text{implicit feedback.} \end{cases} \tag{15}$$

(2) M Step

In M step, we fit the models to the user's behavior across environments for environment clustering, which gives that:

$$\mathcal{L}_{var} = \sum_{e \in \mathcal{E}_{learn}} \frac{1}{|D_e|} \sum_{(u,v) \in D_e} \ell_{rec}(\hat{y}_{u,v,e}, y_{u,v}) \tag{16}$$

where $\mathcal{E}_{learn}$ denotes the environments learned in E step, D_e denotes the data from environment e and ℓ_{rec} is the loss function for the recommendation task. In M step, we minimize the $\mathcal{L}_{var}$ with respect to the embeddings of users, items and environments to better fit the data from each environments respectively.

(3) E Step

In E step, given $|\mathcal{E}_{learn}|$ models which jointly fit the training data, we reallocate the environment label of each data point according to their loss with respect to each model (which can be viewed as the distance between the data point and the centre of the cluster):

$$e_{u,v} \leftarrow \arg\min_{e \in \mathcal{E}_{learn}} \{\ell_{rec}(\hat{y}_{u,v,e}, y_{u,v})\} \tag{17}$$

In summary, as for the environment inference, we iteratively run M step and E step to obtain the learned environments $\mathcal{E}_{learn}$ as well as better embeddings of users, items and environments.

Remark 1 (Joint Optimization of the Whole Framework). *For clarity, we introduce the two stages of the proposed InvPref framework respectively, while these two stages are jointly optimized. The overall objective function of InvPref is given as:*

$$\mathcal{L}_{major} = \mathcal{L}_{pref} + \beta \cdot \mathcal{L}_{var} \tag{18}$$

As for the optimization of embeddings of users, items and environments, we directly perform gradient descent with $\mathcal{L}_{major}$, since the clustering loss $\mathcal{L}_{var}$ also affects the embeddings. As for the environment inference, we generate new environments according to the above mentioned EM-based clustering algorithm. The pseudo-code of InvPref is shown in Algorithm 1.

Remark 2 (Testing phase). *Although we model both the invariant preference and the variant preference in our framework, in testing phase, we only use the learned invariant preference to make predictions, which we think is unbiased and can resist the potential distributional shifts.*

Table 2: Implicit feedback without Uniform Data. InvPref achieves the best performance with a remarkable improvement.

Dataset	Yahoo						Coat					
Top-K	NDCG			Recall			NDCG			Recall		
	top3	top5	top7	top3	top5	top7	top3	top5	top7	top3	top5	top7
MF	0.519	0.589	0.641	0.554	0.728	0.857	0.338	0.372	0.409	0.252	0.373	0.484
IPS-B	0.528	0.601	0.653	0.561	0.741	0.871	0.357	0.384	0.422	0.270	0.381	0.492
SNIPS-B	0.531	0.608	0.658	0.562	0.752	0.877	0.356	0.385	0.425	0.268	0.384	0.498
Rel-B	0.537	0.608	0.656	0.584	0.762	0.880	0.353	0.397	0.430	0.275	0.419	0.520
ATMF	0.535	0.605	0.656	0.572	0.746	0.874	0.350	0.381	0.418	0.263	0.382	0.485
DR-B	0.535	0.605	0.656	0.569	0.744	0.869	0.360	0.386	0.424	0.271	0.381	0.493
DRJL-B	0.537	0.610	0.661	0.572	0.752	0.878	0.327	0.376	0.404	0.248	0.403	0.495
CVIB	0.540	0.614	0.660	0.572	0.755	0.869	0.356	0.389	0.426	0.269	0.391	0.497
InvPref	**0.588**	**0.654**	**0.697**	**0.622**	**0.790**	**0.896**	**0.410**	**0.452**	**0.488**	**0.322**	**0.470**	**0.576**

4 EXPERIMENT

We conduct experiments in real-world datasets, including both general and specific bias, to evaluate the performance of InvPref, in comparison with benchmark methods. Our experiments aim to answer the following questions.

- **RQ1**: Does InvPref outperform other debias methods?
- **RQ2**: Does InvPref address various specific biases?
- **RQ3**: Does the environment act as the agents of confounders?
- **RQ4**: How does the environment inference influence invariant preference learning?

4.1 Experimental Setup

In this section we detail datasets used and baselines compared.

4.1.1 Dataset.

- **Yahoo**[2] & **Coat**[3]. Both datasets consist of a biased dataset of normal user interactions, and an unbiased uniform dataset collected by a random logging strategy. The interaction between user and item is that user rates item (rate 1-5). We randomly sample 10% of the uniform dataset for use by methods that require uniform data, and the remaining 90% as test data (In the comparison with methods that do not require data for training, we use all uniform data for test). In implicit feedback, we treat interactions with scores ≥ 4 as positive samples, and negative samples are collected from all possible user-item pairs (regardless of whether user-item pairs are observed or not) by a random sampling strategy. In explicit feedback, we predict the user's rating (1-5) for the item.
- **MovieLens-1M**[4]. This dataset contains user ratings (1-5) for movies. We treat interactions with ratings ≥ 4 as positive samples to construct implicit feedback, a more common scenario in reality. We construct test data without popularity bias according to the method of [37], and its sample size accounts for 20% of the total data. The rest of the data is used for training.
- **MIND**[5]. It is a widely used news dataset. It records which news is exposed to users and which news users click on. In the test phase, we use the set of news which exposed to users as item pool. Thus, we eliminate exposure bias in testing. For detail, we randomly sample 20% of the exposed items clicked by users as test data and the rest as training data.

4.1.2 Evaluation metrics. We use the following metrics to measure the performance of the model under implicit feedback.

NDCG@K measures the quality of recommendation through discounted importance based on position.

$$DCG_u@K = \sum_{(u,v)\in D_{test}} \frac{I(\hat{z}_{u,v} \leq K)}{log(\hat{z}_{u,v}+1)}$$

$$NDCG@K = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} \frac{DCG_u@K}{IDCG_u@K},$$

where $IDCG_u@K$ is the ideal $DCG_u@K$, $\mathcal{U}$ is the space of users, D_{test} is the test data, and $\hat{z}_{u,v}$ is the position of item v in the recommended rank for user u.

Recall@K measures how many items recommended to user will be interacted.

$$Recall_u@K = \frac{\sum_{(u,v)\in D_{test}} I(\hat{z}_{u,v} \leq K)}{|D^u_{test}|}$$

$$Recall@K = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} Recall_u@K,$$

where D^u_{test} is the set of all interactions of the user u in test data D_{test}.

In explicit feedback, we use two commonly used distance evaluation metrics including Mean Squared Error (MSE) and Mean Absolute Error (MAE).

$$MSE = \frac{1}{|D_{test}|}\sum_{(u,v)\in D_{test}} (\hat{y}_{u,v} - y_{u,v})^2.$$

$$MAE = \frac{1}{|D_{test}|}\sum_{(u,v)\in D_{test}} |\hat{y}_{u,v} - y_{u,v}|.$$

[2]https://webscope.sandbox.yahoo.com/catalog.php?datatype=r&did=3
[3]https://www.cs.cornell.edu/~schnabts/mnar/
[4]https://grouplens.org/datasets/movielens/1m/
[5]https://paperswithcode.com/dataset/mind

Table 3: Implicit Feedback with Uniform Data. InvPref outperforms all compared baselines.

Dataset	Yahoo						Coat					
Top-K	NDCG			Recall			NDCG			Recall		
	top3	top5	top7	top3	top5	top7	top3	top5	top7	top3	top5	top7
MF	0.513	0.587	0.637	0.557	0.739	0.862	0.331	0.360	0.402	0.258	0.375	0.485
IPS-U	0.524	0.603	0.650	0.564	0.756	0.874	0.308	0.348	0.384	0.243	0.370	0.470
SNIPS-U	0.528	0.606	0.654	0.567	0.757	0.875	0.326	0.353	0.389	0.255	0.354	0.467
Rel-U	0.531	0.607	0.654	0.591	0.768	0.882	0.350	0.394	0.428	0.287	0.426	0.524
DR-U	0.532	0.609	0.655	0.571	0.760	0.874	0.342	0.373	0.409	0.267	0.385	0.486
DRJL-U	0.529	0.607	0.652	0.571	0.769	0.874	0.328	0.376	0.403	0.258	0.392	0.488
CausE	0.527	0.601	0.652	0.572	0.753	0.880	0.338	0.366	0.405	0.275	0.379	0.483
KD	0.536	0.610	0.657	0.579	0.760	0.877	0.339	0.372	0.401	0.274	0.389	0.473
AD	0.577	0.646	0.690	0.615	0.782	0.893	0.362	0.414	0.451	0.288	0.452	0.553
InvPref	**0.587**	**0.651**	**0.697**	**0.619**	**0.788**	**0.896**	**0.383**	**0.432**	**0.469**	**0.310**	**0.465**	**0.568**

Table 4: Explicit Feedback without Uniform Data.

Dataset	Yahoo		Coat	
Metric	MSE	MAE	MSE	MAE
MF	1.420	0.867	1.252	0.852
IPS-B	1.019	0.778	1.083	0.811
SNIPS-B	0.989	0.782	1.071	0.811
ATMF	0.999	0.802	1.152	0.830
DR-B	1.011	0.793	1.182	0.865
DRJL-B	1.007	0.772	1.072	0.808
CVIB	0.987	0.784	1.171	0.854
InvPref	**0.949**	**0.765**	**0.998**	**0.783**

Table 5: Explicit Feedback with Uniform Data.

Dataset	Yahoo		Coat	
Metric	MSE	MAE	MSE	MAE
MF	1.415	0.926	1.255	0.853
IPS-U	0.981	0.768	1.208	0.822
SNIPS-U	0.982	0.781	1.202	0.828
DR-U	0.993	0.775	1.215	0.815
DRJL-U	0.981	0.776	1.178	0.830
CausE	0.996	0.797	1.127	0.819
KD	0.988	0.783	1.205	0.834
AD	0.988	0.769	1.010	0.788
InvPref	**0.947**	**0.764**	**0.998**	**0.783**

4.1.3 Baselines. We compare mainstream methods in recommendation debiasing, including both methods designed for special bias and methods oriented toward general debiasing.

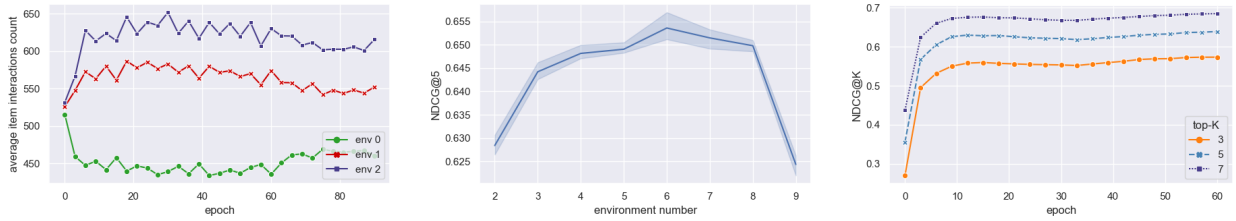
- **Special Bias Method** Methods used to reduce special bias: IPS-B[28], SNIPS-B[30], Rel-B[27](designed only for implicit feedback), DR-B[14], DRJL-B[35], ATMF[26], Fair[1], MACR[37], WMF[12] and EXMF[20].
- **General Debias using uniform data Method** Methods that rely on uniform data.: IPS-U[28], SNIPS-U[30], Rel-U[27](designed only for implicit feedback), DR-U[14], DRJL-U[35], CausE[5] , KD[21] and AD[7].
- **General Debias without uniform data Method**: CVIB[36].

Limited to the space, we detail the baselines in the appendix A. **Note that InvPref does not use the uniform data in all settings.**

4.2 RQ1: Does InvPref outperform other debias methods?

In this section, we first compare the performances of methods in general debiasing setting. We conduct experiments on Yahoo and Coat datasets, including explicit feedback and implicit feedback. According to the results in table 2-5, we can observe that:

- Compared to other baselines, InvPref achieves the best performance with a remarkable improvement for all the metrics in each case, demonstrating the superiority of our model.
- Compared to MF, the methods (e.g. IPS-B, Rel-B) that address specific biases without leveraging uniform data can still bring about some benefits in general debiasing setting, implying various biases are mixed in real applications.
- However, the inability of solving other bias types leads to their relatively poor performances. Even the doubly-robust methods (e.g. DR-B, DRJL-B) cannot achieve the promised performance, showing their limitations.
- The attempt of introducing the uniform data into methods targeting specific biases (e.g. SNIPS-U, DRJL-U) does not achieve desired performance, indicating rough utilization of uniform data even produces negative effect. It is because the size of uniform data is always quite small, easily inducing the overfitting in this dataset.
- The methods can take advantage of auxiliary uniform data if carefully designed. For example, CausE and KD train another teacher model; AD further avoids the overfitting problem through meta-learning technology and reaches competitive results. However, the scarcity of uniform data restrains the power of this kind of method.
- The CVIB performs effectively in implicit feedback setting, but its lack of modeling the unobserved confounders makes its performance far inferior to our proposal, verifying the necessity of considering unobserved confounders that widely exist in real scenarios.



(a) Average frequency of items in the sample sets from each environment.

(b) Environment number influences performance.

(c) The performance gradually improves along training, proving the promotion of our two steps

Figure 3: Exploratory experiments. The first is conducted on the MovieLens-1M dataset and the others are on the Yahoo dataset.

Table 6: Performance of Dealing with Popularity Bias on MovieLens-1M dataset.

Top-K	NDCG			Recall		
	top20	top30	top40	top20	top30	top40
MF	0.079	0.085	0.092	0.071	0.097	0.119
Fair	0.090	0.097	0.104	0.083	0.112	0.137
MACR	**0.101**	**0.108**	**0.114**	0.093	0.123	0.148
InvPref	0.098	0.106	0.113	**0.094**	**0.124**	**0.150**

Table 7: Performance of Dealing with Exposure Bias on MIND dataset.

Top-K	NDCG			Recall		
	top10	top20	top30	top10	top20	top30
MF	0.222	0.260	0.283	0.362	0.499	0.591
WMF	0.235	0.276	**0.300**	0.379	0.522	0.618
EXMF	0.228	0.267	0.289	0.362	0.497	0.588
InvPref	**0.236**	**0.278**	**0.300**	**0.386**	**0.532**	**0.624**

4.3 RQ2: Does InvPref address various specific biases

In this section, we choose two common bias scenarios, exposure bias and popularity bias, to verify the ability of InvPref to deal with various specific biases. For each bias type, we compare with the typical approaches designed for it respectively. From the results reported in Tables 6-7, InvPref achieves the best performance on both NDCG and Recall under exposure bias, while obtains the best performance on Recall and 2nd best on NDCG under popularity bias, proving our framework well describes types of biases that occur in generation procedure of user behaviors.

4.4 RQ3: Does the environment act as the agents of confounders?

In this section, we verify whether it is reasonable to utilize the environment as the agents of confounders. Because we are not exactly aware of which are the confounders in real applications, we predefine the item popularity as the confounder in popularity scenario. From the results in Figure 3(a), it is apparently to see: the samples are clearly clustered into distinct environments in the training phase according to the degree of item popularity, implying the validity of introducing heterogeneous environments.

4.5 RQ4: How does the environment inferring influence invariant preference learning?

In this section, we study the role of environment inferring in our proposed framework. According to the Figure 3(b), we can find that:

- The performance of InvPref becomes better as the number of environments increases. It suggests that more environments are needed to present the heterogeneity in data.
- However, it would hurt the learning process of InvPref owing to the over sparse samples in each environment, if the number of environments is too large.
- After rounds of iterative optimization, the environment inferring and invariant preference learning promote each other until both of them converge finally.

5 CONCLUSION

In this paper, we investigate the problem of general debiasing in recommendation when only the biased observational data is available. We first provide a generic conceptual framework of data generation in recommendation from a view of causal graph. It regards the observational user behaviors are determined by invariant preference and the variant preference. Further, we introduce the environment as the agents of confounders and propose a novel Invariant Preference Learning(InvPref) method. We conduct extensive experiments to support our contributions. The backbone of InvPref is MF, which is not an advanced recommendation model. How to perform General Debiasing on SOTA recommenders without uniform data is still unsolved. Another interesting research direction can be dealing with the challenge brought by implicit feedback in General Debiasing setting without uniform data.

ACKNOWLEDGMENTS

This work was supported in part by National Key R&D Program of China (No. 2018AAA0102004, No. 2020AAA0106300), National Natural Science Foundation of China (No. U1936219, 62141607), Beijing Academy of Artificial Intelligence (BAAI), NSF under grants III-1763325, III-1909323, III-2106758, SaTC-1930941, and Tsinghua University-Siemens Joint Research Center (JCIIOT).

REFERENCES

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling popularity bias in learning-to-rank recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*. 42–46.

[2] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Jean-Christophe Gagnon-Audet, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. 2021. Invariance principle meets information bottleneck for out-of-distribution generalization. *Advances in Neural Information Processing Systems* 34 (2021).

[3] Rashad Al-Jawfi. 2009. Handwriting Arabic character recognition LeNet using neural network. *Int. Arab J. Inf. Technol.* 6, 3 (2009), 304–309.

[4] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893* (2019).

[5] Stephen Bonner and Flavian Vasile. 2018. Causal embeddings for recommendation. In *Proceedings of the 12th ACM conference on recommender systems*. 104–112.

[6] Peter Bühlmann. 2020. Invariance, causality and robustness. *Statist. Sci.* 35, 3 (2020), 404–426.

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

[8] Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*. PMLR, 1180–1189.

[9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research* 17, 1 (2016), 2096–2030.

[10] Yue He, Zimu Wang, Peng Cui, Hao Zou, Yafeng Zhang, Qiang Cui, and Yong Jiang. 2022. CausPref: Causal Preference Learning for Out-of-Distribution Recommendation. (2022). https://doi.org/10.1145/3485447.3511969 arXiv:arXiv:2202.03984

[11] José Miguel Hernández-Lobato, Neil Houlsby, and Zoubin Ghahramani. 2014. Probabilistic matrix factorization with non-random missing data. In *International Conference on Machine Learning*. PMLR, 1512–1520.

[12] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining*. Ieee, 263–272.

[13] Jin Huang, Harrie Oosterhuis, Maarten De Rijke, and Herke Van Hoof. 2020. Keeping dataset biases out of the simulation: A debiased simulator for reinforcement learning based recommender systems. In *Fourteenth ACM conference on recommender systems*. 190–199.

[14] Nan Jiang and Lihong Li. 2016. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*. PMLR, 652–661.

[15] Kia Khezeli, Arno Blaas, Frank Soboczenski, Nicholas Chia, and John Kalantari. 2021. On Invariance Penalties for Risk Minimization. *arXiv preprint arXiv:2106.09777* (2021).

[16] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

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

[18] Masanori Koyama and Shoichiro Yamaguchi. 2021. When is invariance useful in an Out-of-Distribution Generalization problem ? arXiv:2008.01883 [stat.ML]

[19] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. 2018. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5400–5409.

[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] 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.

[22] Dugang Liu, Pengxiang Cheng, Hong Zhu, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2021. Mitigating Confounding Bias in Recommendation via Information Bottleneck. In *Fifteenth ACM Conference on Recommender Systems*. 351–360.

[23] Evan Zheran Liu, Behzad Haghgoo, Annie S. Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021. Just Train Twice: Improving Group Robustness without Training Group Information. arXiv:2107.09044 [cs.LG]

[24] Jiashuo Liu, Zheyuan Hu, Peng Cui, Bo Li, and Zheyan Shen. 2021. Heterogeneous risk minimization. In *International Conference on Machine Learning*. PMLR, 6804–6814.

[25] Benjamin Marlin, Richard S Zemel, Sam Roweis, and Malcolm Slaney. 2012. Collaborative filtering and the missing at random assumption. *arXiv preprint arXiv:1206.5267* (2012).

[26] 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.

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

[28] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. In *international conference on machine learning*. PMLR, 1670–1679.

[29] Zheyan Shen, Peng Cui, Kun Kuang, Bo Li, and Peixuan Chen. 2018. Causally regularized learning with agnostic data selection bias. In *Proceedings of the 26th ACM international conference on Multimedia*. 411–419.

[30] Adith Swaminathan and Thorsten Joachims. 2015. The self-normalized estimator for counterfactual learning. *advances in neural information processing systems* 28 (2015).

[31] Naftali Tishby, Fernando C Pereira, and William Bialek. 2000. The information bottleneck method. *arXiv preprint physics/0004057* (2000).

[32] Wenjie Wang, Fuli Feng, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2021. Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1288–1297.

[33] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 950–958.

[34] Xin Wang, Steven CH Hoi, Martin Ester, Jiajun Bu, and Chun Chen. 2017. Learning personalized preference of strong and weak ties for social recommendation. In *Proceedings of the 26th International Conference on World Wide Web*. 1601–1610.

[35] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2019. Doubly robust joint learning for recommendation on data missing not at random. In *International Conference on Machine Learning*. PMLR, 6638–6647.

[36] Zifeng Wang, Xi Chen, Rui Wen, Shao-Lun Huang, Ercan Kuruoglu, and Yefeng Zheng. 2020. Information theoretic counterfactual learning from missing-not-at-random feedback. *Advances in Neural Information Processing Systems* 33 (2020), 1854–1864.

[37] Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. 2021. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1791–1800.

[38] Le Wu, Xiangnan He, Xiang Wang, Kun Zhang, and Meng Wang. 2021. A survey on neural recommendation: From collaborative filtering to content and context enriched recommendation. *arXiv preprint arXiv:2104.13030* (2021).

[39] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin. 2018. Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *Proceedings of the 12th ACM conference on recommender systems*. 279–287.

[40] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling user interest and conformity for recommendation with causal embedding. In *Proceedings of the Web Conference 2021*. 2980–2991.

APPENDIX

A BASELINE DETAILS

In this section we detail all baselines compared as follow:

- MF. This is one of the most basic recommendation models. The embeddings of users and items is learned by decomposing the user-item interaction matrix, and their inner product is used to predict the feedback.
- IPS-B/IPS-U. This is the most basic IPS-based debiasing recommendation algorithm. This method uses the inverse of the estimated propensity score as the weight of the observed sample to adjust for the biased distribution of the observed data.
- SNIPS-B/SNIPS-U. In order to solve the problem of large variance of traditional IPS-based methods, the academic community proposes self-normalized IPS. The objective function of this method is as follows,

$$\mathcal{L}_{SNIPS} = \frac{\sum_{(u,v)\in D} \frac{\ell_{rec}(\hat{y}_{u,v}, y_{u,v})}{p_{u,v}}}{\sum_{(u,v)\in D} \frac{1}{p_{u,v}}}$$

  where $p_{u,v}$ is the propensity score and ℓ_{rec} is a recommendation loss.
- Rel-B/Rel-U. Based on the idea of positive unlabeled learning, an unbiased estimate of binary feedback modeling for recommender systems is derived. Its objective function is as follows

$$\mathcal{L}_{Rel} = \frac{1}{|D|} \sum_{(u,v)\in D} \left[\frac{y_{u,v}}{p_{u,v}} \ell^1_{u,v} + (1 - \frac{y_{u,v}}{p_{u,v}}) \ell^0_{u,v} \right]$$

  where $\ell^1_{u,v}$ and $\ell^0_{u,v}$ means recommendation loss to positive feed back and negative feedback separately.
- DR-B/DR-U. This method combines the basic data imputation method (which assigns predefined scores to unobserved user-item interactions) with the basic IPS method. As long as one of the two components is accurate, unbiased learning can be achieved.
- DRJL-B/DRJL-U. The method jointly learns an imputation model for pseudo-label generation, which is then combined with the basic IPS method.
- ATMF. This method debiases the recommendation model through an asymmetric tri-training method. Specifically, two pre-models are trained on observed data and then used to generate pseudo-labels. According to them, the base model and pre-models are continuously trained until convergence.
- CVIB. By separating the task-aware mutual information term in the original information bottleneck Lagrangian into factual and counterfactual parts, we derive a contrastive information loss and an additional output confidence penalty, which facilitates balanced learning between the factual and counterfactual domains.
- CausE. Train a teacher model on uniform data and extract unbiased information using the alignment term between the base-model and teacher-model embeddings. Specifically, it minimizes the distance(e.g. l_2) between the base-model and the teacher-model's embedding.
- KD. Train a separate teacher model on the uniform data, and then transfers the model s knowledge to the normal training on biased data.
- AD. The state-of-the-art method that utilize the unbiased information. leverages another (small) set of uniform data to optimize the debiasing parameters by solving the bi-level optimization problem with meta-learning.
- EXMF. The method introduces exposure variables to model the probability of exposure. Exposure probabilities are then converted into confidence weights to reduce exposure bias.
- WMF. The method is a heuristic that reduces exposure bias by assigning lower weights to unobserved samples in implicit feedback.
- Fair. In the training phase, the method uses a regular term to constrain the scores of products with different popularity to be as close as possible to reduce the popularity bias.
- MACR. This method models the generation mechanism of popularity bias based on a pre-defined causal graph. When inferring, it adopts counterfactual reasoning to eliminate the influence of popularity bias.