Addressing Confounding Feature Issue for Causal Recommendation

XIANGNAN HE, University of Science and Technology of China, China

YANG ZHANG*†, University of Science and Technology of China, China

FULI FENG, University of Science and Technology of China, China

CHONGGANG SONG, WeChat, Tencent, China

LINGLING YI, WeChat, Tencent, China

GUOHUI LING, WeChat, Tencent, China

YONGDONG ZHANG, University of Science and Technology of China, China

In recommender system, some feature directly affects whether an interaction would happen, making the happened interactions not necessarily indicate user preference. For instance, short videos are objectively easier to be finished even though the user does not like the video. We term such feature as *confounding feature*, and video length is a confounding feature in video recommendation. If we fit a model on such interaction data, just as done by most data-driven recommender systems, the model will be biased to recommend short videos more, and deviate from user actual requirement.

This work formulates and addresses the problem from the causal perspective. Assuming there are some factors affecting both the confounding feature and other item features, *e.g.*, the video creator, we find the confounding feature opens a backdoor path behind user-item matching and introduces spurious correlation. To remove the effect of backdoor path, we propose a framework named *Deconfounding Causal Recommendation* (DCR), which performs intervened inference with *do-calculus*. Nevertheless, evaluating *do-calculus* requires to sum over the prediction on all possible values of confounding feature, significantly increasing the time cost. To address the efficiency challenge, we further propose a mixture-of-experts (MoE) model architecture, modeling each value of confounding feature with a separate expert module. Through this way, we retain the model expressiveness with few additional costs. We demonstrate DCR on the backbone model of neural factorization machine (NFM), showing that DCR leads to more accurate prediction of user preference with small inference time cost. We release our code at: https://github.com/zyang1580/DCR.

CCS Concepts: • **Information systems** → **Recommender systems**; • **Computing methodologies** → *Machine learning approaches*.

Additional Key Words and Phrases: recommender system, causal inference, causal recommendation, bias, fairness

---

*Corresponding author.

†Partial work done at Tencent.

Authors' addresses: Xiangnan He, University of Science and Technology of China, Hefei, China, 230027, xiangnanhe@gmail.com; Yang Zhang, zy2015@mail.ustc.edu.cn, University of Science and Technology of China, Hefei, China, 230027; Fuli Feng, University of Science and Technology of China, Hefei, China, 230027, fulifeng93@gmail.com; Chonggang Song, WeChat, Tencent, Shenzhen, China, 518054, jerrycgsong@tencent.com; Lingling Yi, WeChat, Tencent, Shenzhen, China, 518054, chrisyi@tencent.com; Guohui Ling, WeChat, Tencent, Shenzhen, China, 518054, randyling@tencent.com; Yongdong Zhang, zhyd73@ustc.edu.cn, University of Science and Technology of China, Hefei, China, 230027.

A: item confounding feature
X: item other features
U: user
Z: the (hidden) factor affecting
     both A and X, *e.g.*, the video creator
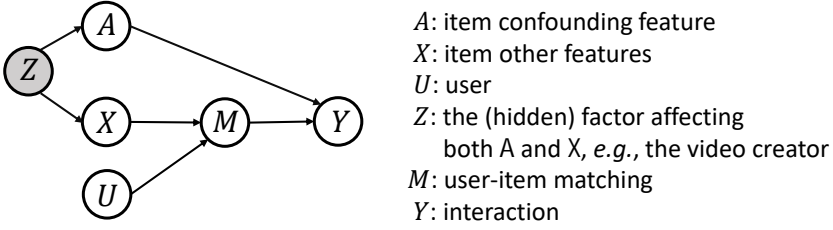M: user-item matching
Y: interaction

Fig. 1. Causal graph to describe the generation process of interactions. $X \longleftarrow Z \longrightarrow A \longrightarrow Y$ is the backdoor path that brings spurious correlations between X and Y.

1 INTRODUCTION

Most recommendation methods assume that the interactions are caused (or captured) by the matching between user reference and item features [9, 34, 49]. However, some item feature could directly affect the happening of an interaction in practice. For instance, videos with short length are easier to be finished, and news articles with attractive title or cover image are easier to be clicked, even though the user does not like the content actually. We name such feature as *confounding feature*, which results in the happened interactions not faithfully reflect user preference. If fitting recommender model on such interaction data, the model will learn the shortcut of the confounding feature, *e.g.,* assigning higher scores for short videos. The biased recommendation is undesired, and even worse, makes the recommender system vulnerable to attack − e.g., the video creator may purposefully upload short videos to make them easier to be recommended.

How to avoid the influence of the confounding feature? An intuitive solution is removing it from the input features, *e.g.,* fitting CTR model on other item features. However, since the interaction data is generated partially due to the confounding feature, the matching model is very likely to learn the effect implicitly (*e.g.,* the item embeddings may encode the semantics of confounding feature). Another solution is training the model with the confounding feature, while removing it during inference to eliminate its effect in the ranking function[1]. The effectiveness of this solution depends on the quality of disentanglement − how well we can separate the effect of confounding feature and other features [45]. Nevertheless, disentangled learning remains to be an open problem which limits the efficacy of this solution.

To explore the root reason of how confounding feature affects, we abstract the interaction generation process as a casual graph, in Figure 1. Let A and X denote the confounding feature and other item features, which affect the interaction (node Y) directly ($A \longrightarrow Y$) and through matching with user preference ($\{U, X\} \longrightarrow M \longrightarrow Y$), respectively. Since A and X describe the same item, they are inevitably affected by some (hidden) factor Z, *e.g.,* the intention of the video creator. When learning recommender models on the interaction data, clearly, the effect of A will be counted in the model prediction. More profoundly, the confounding feature A opens the backdoor path $X \longleftarrow Z \longrightarrow A \longrightarrow Y$, bringing some spurious correlations between X and Y.

---

[1]This solution is originally designed for addressing some existing fairness or biases issues in recommendation [36, 45] instead of the proposed confounding feature issue, but it can be utilized to deal with the proposed issue, intuitively.

To make the recommendation free from the impact of A, we need to estimate the causal effect of X (or equivalently, M) on Y. To this end, we need to cut off the backdoor path through intervention [25]. However, it is hard to conduct interventional experiments on X since item features are usually static and unchangeable. An alternative way is *do-calculus* [25], which can achieve the same effect of intervention on observed data. Particularly, we propose a Deconfounding Causal Recommendation (DCR) framework that approaches user-item matching as $P(Y|U, do(X))$. During training, we estimate the correlation $P(Y|U, X, A)$ to fit the historical interaction data, since it caters to the generation of interactions. Whereas during inference, we use the intervened $P(Y|U, do(X))$ as the ranking function.

Taking one step further, according to the causal graph in Figure 1 and *the backdoor adjustment* [25], $P(Y|U, do(X))$ is equal to $\sum_{a \in \mathcal{A}} P(Y|U, X, a)P(a)$. Which means, we need to iterate over all values of the confounding feature, then conduct a weighted sum on $P(Y|U, X, a)$. This significantly increases the time cost in the inference stage — by $|\mathcal{A}|$ times[2] — making the solution prohibitive for practical use. To address this challenge, we propose a mixture-of-experts (MoE) model architecture for our DCR framework. Specifically, we use a shared backbone model to capture the matching between U and X, the output of which is fed into a separated expert module designed for each value of the confounding feature. Since the U-X matching part is shared for all experts, the time cost of evaluating $P(Y|U, do(X))$ can be reduced largely. Meanwhile, modeling each confounding feature value with a separated expert ensures the modeling fidelity, compared with the approximation method used by [35, 56][3].

The main contributions of our work are summarized as follows:

- We study a new problem of *confounding feature* in recommender system, analyzing its damage effect from the causal perspective.
- We propose a new solution framework DCR to address the problem with intervened inference, which is supplemented with a MoE module to address the efficiency challenge.
- We implement our solution on a well-known feature-based recommender Neural Factorization Machine (NFM) [11] and conduct extensive experiments on two real-world datasets, verifying the effectiveness of our proposal.

2 PROBLEM DEFINITION

We use an uppercase character (*e.g.*, X), calligraphic font (*e.g.*, $\mathcal{X}$) and lowercase character (*e.g.*, x) to denote a random variable, the sample space of the variable, and a specific value of the variable, respectively. Let $\mathcal{D}$ denote historical interactions (*e.g.*, finished playing or click) where the user-item pairs are given binary labels. The target of recommendation is learning a model from $\mathcal{D}$ for predicting to what extent an item will match the user preference. Both user and item could be described by rich side information, *e.g.*, user demographics and item attributes (category, tags, *etc.*). Different from conventional settings [11], we discriminate the *confounding feature* (A) of item from the remaining *content feature* (X).

Conceptually, confounding feature is an item feature that has direct effect on the interaction Y, regardless of user preference. We assume that the confounding feature is a discrete variable of K (= $|\mathcal{A}|$) values. Because most item features are discrete in recommender system, and it is common to discretize continuous features for better modeling and interpretability. Remarkably, there exist techniques [53, 59] for identifying such confounding feature. Thus, this work focuses on the problem that with confounding feature as known (*i.e.,* assuming the confounding feature

---

[2] $\mathcal{A}$ denotes all possible values of the confounding feature, *i.e.*, the sample space of A. $|\mathcal{A}|$ denotes the size of $\mathcal{A}$.
[3] The widely used approximation method is: $P(Y|U, do(X)) \approx P(Y|U, X, \sum_{a \in \mathcal{A}} aP(a))$, which however uses the same parameterization for all confounding feature values.

has been identified), how to eliminate its impact in recommendation. As the first step, this work considers single confounding feature, and leaves the extension to multiple confounding features for future work.

3 METHOD

We first present the causal view for analyzing the impact of the confounding feature on the recommendation process (Section 3.1). Then we introduce our Deconfounding Causal Recommendation (DCR) framework to approach the user-item matching through backdoor adjustment (Section 3.2). Thirdly, we present a MoE model architecture to achieve efficient backdoor adjustment (Section 3.3). Lastly, we discuss the generality of DCR and show it retains the ability of addressing the confounding feature issue when causal relationship changes (Section 3.4).

3.1 Causal View of Confounding Feature

Conceptually, causal graph [25] is a directed acyclic graph, in which a node represents a variable and a directed edge denotes the causal relation between two connected nodes. Functionally speaking, causal graph is an abstract of the data generation process, and widely used to guide the model design [36, 56]. Figure 1 shows the causal graph for interaction generation when the confounding feature exists. Next, we explain the semantics of the causal graph.

- Node U denotes the user, specifically, user features, including ID.
- Node A denotes the given confounding feature.
- Node X denotes other content features of item, including ID, that are associated with user preference.
- Node Z represents some hidden factors affecting both confounding feature A and content feature X. Such factors are caused by the production of items, which might be unobservable, *e.g.,* the intention of the video creator.
- Node M represents the user-item matching, reflecting to what extent the content features match user preference.
- Node Y denotes the interaction label, indicating whether the interaction behavior (*e.g.,* finished playing and click) is happened.
- Edges $\{X, U\} \longrightarrow M$ denote that the user-item matching is determined by the user features U and item content features X.
- Edges $\{A, M\} \longrightarrow Y$ represent that the interaction is determined both by the level of user-item matching M and the confounding feature A. The edge $A \longrightarrow Y$ corresponds to the phenomenon that the confounding feature affects the probability of an interaction, but not reflects true user preference.
- The edges $A \longleftarrow Z \longrightarrow X$ denote that hidden factor Z affects both confounding feature A and content features X. For instance, to create a wedding video, the creator (Z) will make relatively longer video (A) and choose touching background musics (X). As such, A and X will exhibit correlation in the observed data due to the hidden common cause Z.

Given the target of learning a ranking function from historical interactions $\mathcal{D}$, existing methods usually estimate the correlation $P(Y|U, X, A)$ or $P(Y|U, X)$. However, both choices are problematic:

- **Modeling** $P(Y|U, X, A)$. The models based on $P(Y|U, X, A)$ will account for the direct effect of $A \longrightarrow Y$. Consequently, the predictions are biased towards some special A, *e.g.,* assigning higher scores for short videos.
- **Modeling** $P(Y|U, X)$. From the causal graph, we recognize a backdoor path between X and Y, *i.e.,* $X \longleftarrow Z \longrightarrow A \longrightarrow Y$, wherein A is a confounder between X and Y. Therefore, ignoring

> A in the input will make the model learn the spurious correlation between X and Y, which also leads to biased recommendation.

In this light, the key to eliminate the impact of A lies in cutting off both the direct path $A \longrightarrow Y$ and backdoor path $X \longleftarrow Z \longrightarrow A \longrightarrow Y$ in the model prediction. Accordingly, *making recommendation with the causal effect of X on Y* can achieve the target.

3.2 Deconfounding Causal Recommendation

We now consider how to obtain a recommender model based on the causal effect of X on Y.

3.2.1 Causal Intervention. By definition, the causal effect of X on Y is the changes of Y when forcibly changing the value of X from a reference value to a target value. Therefore, the key to estimate causal effect lies in obtaining the outcome Y after the intervention on X. In practice, a *de facto* standard to obtain the outcome of causal intervention is conducting randomized controlled trial [23]. However, such experiments are very expensive in recommendation and not practical for the confounding feature issue, since the items are typically created by a third party. Therefore, we have to estimate the intervention outcome from the observational data.

Intervention with *do-calculus*. Fortunately, the *do-calculus* in causal science [25] provides an alternative solution to estimate $P(Y|U, do(X))$. Considering that there is a backdoor path between X and Y, we take *the backdoor adjustment* [25] to identify the target causal effect. Formally, we have,

$$P(Y|U, do(X)) = \sum_{a \in \mathcal{A}} P(Y|U, X, A = a) P(A = a), \tag{1}$$

where $P(Y|U, X, A = a)$ and $P(A = a)$ are both identifiable conditional probability distributions. Conceptually, $P(Y|U, do(X))$ blocks the backdoor path by conditioning on the confounder A. From the view of controlled experiment, it means that we select a group of candidate user-item pairs such that A has the fixed distribution $P(A)$ across different groups, and set their X as the target value; and then observe the outcome distribution $P(Y|U, X, A = a)$ over the candidates.

Given the target value x and reference value x^*, the difference between $P(Y|U, do(X = x))$ and $P(Y|U, do(X = x^*))$ is equal to the difference between the controlled experiments over two randomized groups since the marginal distribution $P(A)$ is invariant. In this way, the obtained causal effect on Y of changing the value of X from the reference value x^* to the target value x, is freed from the influence of the confounding feature A. Remarkably, we can directly regard $P(Y|U, do(X = x))$ as the causal effect of X on Y, *i.e.,* discarding the reference status, since the reference status is the same for all items and thus has no influence on item ranking. Theoretically, we can also conduct the backdoor adjustment over Z as $P(Y|U, do(X)) = \sum_{z \in Z} P(Y|U, X, Z = z) P(Z = z)$. Nevertheless, this is impractical since Z is a hidden confounder that cannot be observed.

3.2.2 Estimating $P(Y|U, do(X))$. To estimate $P(Y|U, do(X))$, as the procedure in PD [56], we need to: 1) model the two probability distributions in Equation (1), *i.e.,* $P(Y|U, X, A)$ and $P(A)$, through historical interaction data $\mathcal{D}$; and 2) infer the expectation of $P(Y|U, X, A)$ over $P(A)$.

Estimating $P(A)$. Recall that the confounding feature A is discrete with K possible values. We have $K \ll |\mathcal{D}|$, where $|\mathcal{D}|$ is the size of dataset $\mathcal{D}$. Therefore, we can directly approximate $P(A = a)$ with the ratio of samples with $A = a$, *i.e.,*

$$P(A = a) = \frac{|\{(U, X, A, Y)|A = a\}|}{|\mathcal{D}|}. \tag{2}$$

Estimating $P(Y|U, X, A)$. Apparently, it is infeasible to directly observe all probabilities from $\mathcal{D}$ due to the data sparsity in recommendation. To resolve this issue, we resort to a machine learning
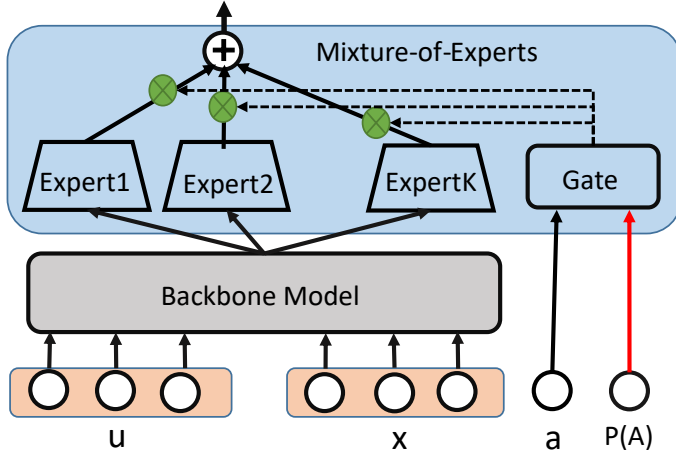
Fig. 2. Illustration of the MoE model architecture for Deconfounding Causal Recommendation. In the gate module, the inputs with black arrow and red arrow are used for model training and model inference, respectively.

model to learn the distribution. In line with existing recommendation work [5, 29], we assume that $P(Y|U, X, A)$ follows a Bernoulli distribution. Given a specific condition $U = u$, $X = x$, and $A = a$, we learn a mapping function $f(u, x, a)$ to calculate the interaction probability[4] (i.e., $Y = 1$) by minimizing its negative log-likelihood over $\mathcal{D}$. Formally,

$$\min \sum_{(u,x,a,y) \in \mathcal{D}} - y \log \left(f(u, x, a) \right) \tag{3}$$
$$- (1 - y) \log \left(1 - f(u, x, a) \right).$$

The L_2 regularization is used over the parameters of the mapping function to control overfitting but is not shown for briefness. We can implement the mapping function $f(u, x, a)$ by any feature-aware recommender model, like FM [27] and NFM [11].

Inference. Once the model is trained, we can evaluate $P(Y|U, do(X))$ according to Equation (1) for recommendation scoring. Given a user-item pair, the recommendation score is calculated as:

$$P(y = 1|u, do(x)) = \sum_{a \in \mathcal{A}} P(a) \cdot f(u, x, a). \tag{4}$$

To summarize, we first train a model according to Equation (3) to learn $P(Y|U, X, A)$. In the inference stage, we calculate $P(y = 1|u, do(x))$ according to Equation (4) to free the recommendation from the impact of the confounding feature A. We term this general framework as *Deconfounding Causal Recommendation* (DCR).

3.3 Mixture-of-Experts Model Architecture

As above, a direct way to evaluate $P(y = 1|u, do(x))$ is enumerating each value of A, conducting the model inference $f(u, x, a)$ for K times (note that $K = |\mathcal{A}|$). Apparently, it will significantly increase the time cost in recommendation inference. As such, we need to consider how to accelerate the inference.

---

[4]Note that the output can be seen as the parameter of the Bernoulli distribution.

NWGM approximation. Note that $P(y = 1|u, do(x))$ is indeed the expectation of $f(u, x, a)$ over the distribution $P(A)$. We can thus achieve the target with the NWGM approximation [50], which has been widely used to approximate the expectation of functions [26, 35, 56]. Formally,

$$P(y = 1|u, do(x)) \approx f\left(u, x, \sum_{a \in \mathcal{A}} a * P(a)\right). \tag{5}$$

Nevertheless, this approximation will sacrifice the estimation precision especially when the function $f(\cdot)$ is nonlinear.

Noticing that directly adjusting the inference strategy results in the dilemma between efficiency and accuracy, we now consider to address the issue by adjusting the model architecture. We propose a MoE model framework with two considerations: 1) calculating the causal effect with one time of model inference; and 2) preserving sufficient modeling fidelity. The key lies in simultaneously calculating $f(u, x, a)$ for all values of A in one model inference. In this light, the MoE framework handles each value of A with a specific expert[5]. We then let the experts share the same backbone for handling the matching between U and X. This can largely save the computation cost since the U-X matching, especially when considering high-order feature interactions [18], is the most computation-intensive part of $f(u, x, a)$.

Concretely, the MoE framework (*cf.* Figure 2) includes:

- *A Backbone Recommender Model*, which aims to learn a representation for the matching between item features x and u:

$$\boldsymbol{m} = f_\Theta(u, x), \tag{6}$$

  where $\boldsymbol{m}$ is a latent representation that denotes the matching signal, and Θ denotes the model parameters.

- *K Experts*, where each expert corresponds to a confounding feature value a, mapping the matching representation $\boldsymbol{m}$ and a to the interaction probability. We implement each expert as a multi-layer perceptron with two hidden layers. Formally,

$$f(u, x, a) = f_{\phi_a}(\boldsymbol{m}|a), \tag{7}$$

  where ϕ_a denotes the parameters of the expert for a. Across the experts, we use a gate with a one-hot input to select the expert for each training sample. In the inference stage, we feed $P(A)$ into the gate to calculate the weighted sum over the experts, which evaluates the causal effect $P(y = 1|u, do(x))$.

We term the DCR implemented with the MoE model architecture as DCR-MoE. Algorithm 1 shows the procedure of training (line 2-8) and inference (line 10-14) of DCR-MoE. In training, after the MoE has been initialized (line 2), we will iteratively update the MoE model in a mini-batch manner. Each iteration has three key steps: compute the matching $\boldsymbol{m}$ between item feature x and u with the same backbone recommender f_Θ for all samples (line 5); then compute the output of the expert corresponded to confounding value a, *i.e.*, $f_{\phi_a}(\boldsymbol{m}|a)$, for each sample sample (u, x, a) (line 6); next update all model parameters by minimizing the loss in equation (3) with $f_{\phi_a}(\boldsymbol{m}|a)$ as the prediction for each sample (line 7). Note that although all experts are expected to be updated in each iteration, each training sample will only be fed into and used to update an expert that corresponds to the value of its confounding feature. During inference, we will first compute the matching $\boldsymbol{m}$ for each candidate user-item pair (u, x) (line 11), then compute the outputs of all experts based on the $\boldsymbol{m}$ (line 12). Last, the prediction is generated by summing over all experts' outputs with $P(A)$ as weights (line 13). Different from the training, all experts will be utilized to generate the prediction for each candidate.

---

[5]Each expert models a stratification [8] $P(Y|U, X, A)$ *w.r.t.* the value of A.

Algorithm 1: DCR-MoE

Input: Estimated $P(A)$ with equation (2), training dataset $\mathcal{D}$, testing dataset $\mathcal{D}_{testing}$, and the number of experts K

Output: Predictions to user-item candidates in the testing dataset

1 // start training;
2 Initialize the the backbone recommender model f_Θ and the K experts $\{f_{\phi_a}\}_{a\in\mathcal{A}}$ of the MoE;
3 **for** *stop condition is not reached* **do**
4   Randomly sample a batch of data from $\mathcal{D}$;
5   Compute $\boldsymbol{m} = f_\Theta(u, x)$ for each training sample (u, x, a) according to equation (6);
6   For each sample (u, x, a), compute the output of **the expert for** a, *i.e.,* $f_{\phi_a}(\boldsymbol{m}|a)$ in equation (7);
7   Update the backbone recommender f_Θ and experts $\{f_{\phi_a}\}_{a\in\mathcal{A}}$ by minimizing the loss in equation (3) with $f(u, x, a) = f_{\phi_a}(\boldsymbol{m}|a)$;
8 **end**
9 //start inference;
10 **for** *each candidate user-item pair (u, x) in $\mathcal{D}_{testing}$* **do**
11   Compute $\boldsymbol{m} = f_\Theta(u, x)$;
12   Compute the outputs of **all experts**, getting $\{f_{\phi_a}(\boldsymbol{m}|a)\}_{a\in\mathcal{A}}$;
13   Compute $\sum_{a\in\mathcal{A}} f_{\phi_a}(\boldsymbol{m}|a)P(a)$ as the prediction;
14 **end**
15 return all predictions for testing dataset;

3.4 Generality of DCR

It is worth mentioning that DCR presents a general solution for addressing the confounding feature issue. It works not only for the case that a confounder exists between A and X (as shown in Figure 1), but also for other cases that the causal relation between A and X is different. Considering the direct causal relation (i.e., no other variable like mediator or confounder) between A and X, there are three possible cases:

- $A \longrightarrow X$. As shown in Figure 3(a), A has direct causal effect on X. In this case, A is still a confounder between X and Y. Our intervention with *do-calculus* (abbreviated as *do-intervention*) in Equation (1) still gives the causal effect of X on Y.
- $X \longrightarrow A$. As shown in Figure 3(b), there is a directed edge from X to A. In this case, $X \longrightarrow A \longrightarrow Y$ is also a causal path, *i.e.,* A becomes one of the mediators for the causal effect of X on Y. It is thus harder to block the undesired direct effect of A on Y, which cannot be achieved by estimating the total causal effect of X on Y. This is because $P(Y|U, do(X))$ contains the effect of A on Y. To achieve the goal, we should dive into the path-specific causal effect of X on Y through the path $X \longrightarrow M \longrightarrow Y$. According to [25], we define such causal effect as:

$$P(Y|U, X = x, A_{x^*}) - P(Y|U, X = x^*, A_{x^*})$$
$$= \sum_{a\in\mathcal{A}} \left(P(Y|U, x, a) - P(Y|U, x^*, a) \right) P(a|x^*), \tag{8}$$

where x and x^* represent the target and reference values of X respectively, and A_{x^*} denotes whatever possible values of A if let $X = x^*$. All possible values of x can take the same reference value x^* to estimate their causal effects. The second term in Equation (8) will thus not affect the ranking of items when using this path-specific causal effect as a recommendation policy.
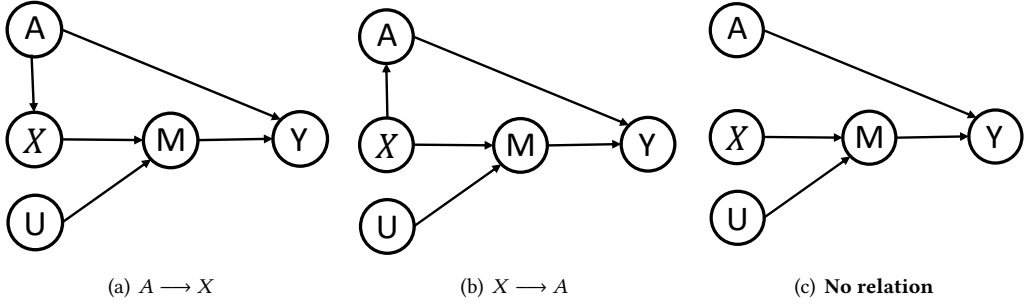
(a) $A \longrightarrow X$ (b) $X \longrightarrow A$ (c) **No relation**

Fig. 3. Three possibilities of direct causal relations between A and X. Our DCR solution can also handle the three cases.

Therefore, our target is to estimate:

$$\sum_{a \in \mathcal{A}} P(Y|U, x, a)P(a|x^*). \tag{9}$$

Remarkably, we can take any value of X as the reference status. As long as there exists a x^* satisfying $P(a|x^*) = P(a)$, we have,

$$\sum_{a \in \mathcal{A}} P(Y|U, x, a)P(a|x^*) \propto \sum_{a \in \mathcal{A}} P(Y|U, x, a)P(a). \tag{10}$$

It means our do-intervention can to some extent approximate the path-specific causal effect under this case.

- No relation. As shown in Figure 3(c), A and X are independent item features. In this case, performing intervention equals directly inferring the plain correlation. Formally,

$$P(Y|U, do(X)) = \sum_{a \in \mathcal{A}} P(Y|U, X, a)P(a) = P(Y|U, X). \tag{11}$$

As we consider the average causal effect instead of the individual causal effect, our do-intervention still captures the causal effect of X on Y. That is, the do-intervention still focuses on the user-item matching, eliminating the direct effect of A on Y.

To summarize, the proposed do-intervention can eliminate the direct (and confounding) effects of the identified item feature A and recognize the true user-item matching whatever the (direct) causal relations between A and X[6]. It means that DCR (including MoE) is a general framework for tackling the confounding feature issue not restricted to the causal graph in Figure 1. Moreover, these analyses indicate the potential of taking DCR as a uniform framework to deal with other issues in recommendation that can be abstracted as the causal graphs in Figure 1 and 3. For example, the popularity bias issue [56] and position bias issue [10] can be described by Figure 3(a) and Figure 3(b), respectively.

---

[6]Note that although we only discuss the cases regarding the direct causal relations between A and X. Indeed, the discussed three cases can represent many other complicated cases: 1) $A \longrightarrow X$ can represent the case that there are paths from A to X; 2) $X \longrightarrow A$ can represent the case that there are paths from X to A; 3) no relation can also represent the case that the path between A and X contains colliders, since the colliders indeed block this path, making A and X independent. And the corresponding discussions can be directly applied to these complicated cases.

4 RELATED WORK

In this section, we discuss the work on recommendation fairness and bias, which are both relevant with our work in terms of eliminating the impact of some predefined factors. We also review causal recommendation methods from the technical perspective.

4.1 Fairness in Recommendation

In recommendation, fairness [42] is usually defined on some sensitive features such as user gender [13] and user race [61]. Towards the fairness goal, most efforts [2, 13, 16, 17, 47] try to remove the information of sensitive features from making recommendations. A line of research achieves this target through post-processing [14, 16], which forcibly re-ranks the items with heuristically defined fairness policies. [17, 46, 47, 52, 61] try to remove the sensitive information by adding the fairness-aware loss into the training objectives, *e.g.,* [61] constraints the representations of sensitive and nonsensitive features to be orthogonal; [52] adds the loss defined based on fairness metrics such as the non-parity. Among them, [15, 17, 46, 47] adopt adversarial training to optimize the recommendation loss and fairness loss.

In summary, these fairness-oriented methods try to remove all information related to the sensitive feature regardless its usefulness for estimating user preference, which may reduce the modeling fidelity. In contrast, we aim to make the recommendation free from the impact of the confounding feature, including its direct and confounding effects, without sacrifice of the information of other item features (X in Figure 1). What's more, the above methods are at the level of correlation instead of causality.

4.2 Bias in Recommendation

Recommender systems also face various bias issues generated by different reasons [4]. The position bias [10] and clickbait [36] issues are most related to the confounding feature issue. Position bias happens as users tend to click items in the front positions of recommendation lists. Differently, the considered confounding feature is one type of item features instead of context features, and the underlying causal mechanisms are also different. The clickbait issue [36] considers the bias generated by the effect of the item's cover or title on click. Differently, we do not restrict the type of interaction. Moreover, its causal mechanism is different from us, *e.g.,* it does not consider confounders. To handle bias issues, the most widely considered method is IPW-based methods[7] [10, 12, 28, 30, 48], which adjust the training distribution by reweighting training samples with propensity scores. However, the propensities are difficult to set properly, e.g., causing the high variance issues etc [4, 56]. [33, 58] respectively take a cross-pairwise loss and a constructive loss to achieve unbiased recommendation. They indeed implicitly reweight training samples by controlling the sampling of negative samples in the loss. Utilizing unbiased data (collected from random exposure) [3, 19, 39] to guide the model learning is another type of methods, however such unbiased data is at the expense of user experience and risky to collect. Post-hoc re-ranking or model regularization methods are also designed [1, 60], but they are heuristically designed and lack theoretical foundations for effectiveness [56]. Different from these methods, we take causal adjustment — the backdoor adjustment — to remove the undesired effects of the confounding features.

4.3 Causal Methods for Recommendation

Recently, some efforts try to introduce causal inference to recommendation. One line of research considers the confounding issue in recommendation [6, 29, 31, 35, 41, 56]. Among them, [35, 38, 56]

---

[7]Please note that IPW is also one type of causal methods. The doubly robust (DR) based method [48] is also based on IPW but adds an imputation model to improve model robustness.

also adopt causal intervention, where PD [56] aims at solving the popularity bias problem, and DecRS [35] considers the bias amplification problem. Technically, PD takes a similar two-step procedure to us for causal effect estimation, but assumes the form of the effect of confounders and designs a partially linear model to accelerate the inference. Differently, the proposed DCR does not need this assumption and takes MoE to accelerate the inference, keeping the model's expressiveness. DecRS takes a different procedure for causal effect estimation with the NWGM approximation, which may sacrifice the estimation precision. DCR directly estimates causal effects with MoE, ensuring both model fidelity and efficiency. [38] adopts the backdoor adjustment to eliminate the bias introduced by heterogeneous information. But their method is specially designed based on heterogeneous information networks and the backdoor adjustment is used for useful information selection instead of directly generating recommendations. Besides, [41] learns substitutes for unobserved confounders by fitting exposure data, to solve the unobserved confounder problem. [43] learns substitutes for unobserved confounders based on user historical interactions in sequential recommendation, and achieves deconfounding via IPW. [6, 29] adjust data distribution to estimate causal effect with IPW methods. [20] solves the confounding problem with information bottleneck [20]. These methods do not perform intervention with *do-calculus*.

Another type is the counterfactual-based method. Work [36, 45] tries to utilize the counterfactual inference to estimate their target causal effects. [36] is to solve the clickbait issue, and [45] is to eliminate the popularity bias. Both of them need to estimate the total effect and then remove the undesired effect by comparing the factual and counterfactual worlds. [37] focuses on out-of-distribution recommendation. It first learns the interaction generation process and uses counterfactual inference to mitigate the effect of out-of-date interactions. Different from these methods, we take causal intervention instead of counterfactual inference to estimate the target causal effect. Other work [22, 32, 44, 54, 55] is less related to us. [22, 44, 51, 54] try to generate some counterfactual data and take these data to improve the model performance, *e.g.,* [51] generates counterfactual data by simulating the recommendation process and takes the generated data to train a ranking model. [55] proposes a counterfactual importance sampling for recommendation based on the bandit. [32] proposes a framework for finding counterfactual explanations for neural recommenders. Though these works consider the recommendation from the causal perspective, they make use of the counterfactual inference to generate explanations or interaction data. We take intervention with *do-calculus* to estimate causal effects and generate recommendations based on the estimated results.

5 EXPERIMENTS

In this section, we conduct experiments to answer the following three questions:

RQ1: How is the performance of the proposed framework DCR implemented with MoE (denoted as DCR-MoE) compared with existing methods?

RQ2: How do the design choices affect the effectiveness and efficiency of DCR-MoE?

RQ3: Has the proposed method effectively eliminated the impact of the confounding feature?

5.1 Experimental Setting

5.1.1 Datasets. To evaluate the effectiveness of addressing the confounding feature issue, we require datasets with: 1) biased training data affected directly by the confounding feature; and 2) unbiased testing data *w.r.t.* the confounding feature, which can reflect true user interests. In this light, we select two video recommendation datasets, treating the finished playing and a post-playing feedback as the training label and testing label of each sample (user-item pair), respectively. Remarkably, taking a label that is different from the training label as the testing label is a reasonable setting to evaluate debiasing results [36]. As aforementioned, video length is a confounding feature

Table 1. Dataset Information. A denotes the confounding feature. $\rho_1(or\rho_2)$ denotes the Pearson correlation coefficient between the confounding feature A and the training (or testing) label.

Dataset	#users	#items	#samples	A	ρ_1	ρ_2	$|\rho_1/\rho_2|$
Kwai	18,019	422,144	19,429,358	video length	-0.085	0.014	6.07
Wechat	20,000	96,539	7,195,486	video length	-0.131	-0.015	8.73

in video recommendation. Note that each sample has both training and testing labels, which are both binary. For each dataset, we split all samples into training/validation/testing sets with a ratio of 6 : 2 : 2. To avoid data leakage, we make sure each sample regardless of label types only occurs in either training or testing/validation sets.

1) **Kwai**: It is a short video recommendation dataset released in the Kuaishou User Interest Modeling Challenge[8]. It is a relatively large dataset in research, having $19,429,358$ interacted samples between $18,019$ users and $422,144$ videos. It has two types of interaction labels: finished playing and liking. The finished playing is taken as the training label, and the other is taken as the testing label. We discretize the confounding feature, *i.e.*, the video length, such that it has 6 possible values. We take other items' discrete features that can be one-hot encoded as the other item feature X, indeed, only id information can be utilized.

2) **Wechat**: The dataset is released in the WeChat Big Data Challenge[9], which records user behaviors on short videos. It has $7,195,486$ interacted samples between $20,000$ users and $96,539$ videos. It contains many types of interaction labels, such as finished playing, liking, and read-comment. We take the finished playing as the training label, and the read-comment as the testing label. We also discretize the confounding feature (video length) such that it has 6 possible values. We take some other inherent and discrete features such as *bgm_song_id* as the other item feature X.

To show that the selected testing label is more free from the impact of the confounding feature compared with the training label, we compute the Pearson correlation coefficients between the confounding feature and the two labels, respectively. The Pearson correlation coefficient is a measure of linear correlation between two variables[10]. The correlation coefficient ranges from -1 to 1, and a higher absolute value implies a stronger linear correlation of the variables. We denote the correlation coefficient between the confounding feature and the training label as ρ_1, and the correlation coefficient between the confounding feature and the testing label as ρ_2. The computed correlation coefficients and other statistics of datasets are summarized in Table 1. We can find that for both the two datasets, the absolute value of ρ_2 (*i.e.*, $|\rho_2|$) is far small than that of ρ_1 (*i.e.*, $|\rho_1|$), and ρ_2 is very close to zero. These results show that the testing label has weaker linear correlations to the confounding feature, implying the testing label is more free from the impact of the confounding feature to some degree. Therefore, we think taking the selected labels as testing labels is appropriate.

5.1.2 Compared Methods. We implement the DCR framework based on the MoE architecture, *i.e.*, DCR-MoE. Specially, we implement the backbone recommender model of MoE as the bottom layer of Neural Factorization Machines (NFM) [11], including embedding and the bi-interaction (EB) layers, and implement each expert of MoE as the deep part of NFM, including hidden and prediction

---

[8]https://www.kuaishou.com/activity/uimc
[9]https://algo.weixin.qq.com/
[10]The measure can only reflect a linear correlation of variables, and ignores many other types of relationships or correlations. Anyway, it is one of the most widely used measurements.

layers. Then, we compare it with the following correlation-based, IPW-based, fairness-oriented, and counterfactual inference based methods:

-NFM-WA [11]. This method trains NFM with the confounding feature as the input, *i.e.,* estimating the correlation $P(Y|U, X, A)$ as the user-item matching.

-NFM-WOA. This method trains NFM without the confounding feature as the input, *i.e.,* estimating the correlation $P(Y|U, X)$ as the user-item matching.

-IPW [28], refers to the conventional inverse propensity weighting method. It tries to capture true user preference from biased data by re-weighting training samples. Following the previous work [28], 1) we assume the interaction probability is equal to $P(Y = 1|A)P(R = 1|U, I)$, where R represents the relevance between user U and item I; 2) the propensity weight is defined as the inverse of $(\frac{P(Y=1|A=a)}{\max_{a'} P(Y=1|A=a')})^{0.5}$ for positive samples with $A = a$. The weights for negative samples are defined in a similar way. We also implement IPW with NFM.

-FairGo [47]. This method is proposed for dealing with unfairness probelm in recommendation. To achieve fair recommendation, it attempts to remove the effects of sensitive features by adversarial training. Here, we take it to eliminate the impact of the confounding feature, *i.e.,* taking the confounding feature as the sensitive feature. We implement it based on NFM for a fair comparison. The hyper-parameter λ to control removing the effects of the sensitive feature is tuned in the range of $\{1e\text{-}3, 1e\text{-}2, 1e\text{-}2, 0.1, 0.2, 10, 50\}$.

-CR [36]. This is a counterfactual method for clickbait issue [36]. It trains the model with the exposure feature but removes the undesired effect of exposure features at inference with counter-factual reasoning. We replace the exposure feature with the confounding feature A to eliminate the impact of A. We implement CR based on NFM, and take the *SUM-tanh* strategy (showing the best results in [36]) to fuse the effects of exposure features (A) and other features (X) on interactions. The hyper-parameter α to control the influences of exposure feature is searched in the range of $\{0.1, 0.25, 0.5, 0.75, 1, 2, 3, 4, 5\}$.

5.1.3 Hyper-parameters. For a fair comparison, all methods are optimized with the binary cross-entropy (BCE) loss and tuned on the validation set. We optimize all models with adagrad [7] optimizer with default mini-batch size of 1024. Following previous work [9] to set a small embedding size for FM-based methods, we fix the embedding size to 16 for all compared methods. The deep part of NFM (including hidden layers and a prediction layer) and the experts of our MoE are implemented as MLPs having two hidden layers with sizes 256 and 128, respectively. We search the learning rate in the range of $\{0.01, 0.001\}$. For all methods, we have two different L_2 regularization coefficients: one for embedding layers and the other for other model parameters, which are both searched in the range of $\{1e\text{-}1, 1e\text{-}2, \dots, 1e\text{-}6, 0\}$. The best hyper-parameters for reported results are found by the grid-search with a popular hyper-parameter tuning tool – ray[11] [24], and the patience for early stopping is set as 10 epochs.

5.1.4 Metrics. To measure the recommendation performance, we adopt three widely-used evaluation metrics: Recall, Mean Average Precision (MAP), which consider whether the relevant items are retrieved within the top-N positions, and NDCG that measures the relative orders among positive and negative items in the top-N list. We generate recommendation lists just in the observed data, *i.e.,* ranking all the items appearing in the validation or testing sets, similar to [3]. Since the average number of interacted items by a user in Kwai is significantly bigger than that in Wechat, we report the results of top-10 recommendation and top-20 recommendation for Kwai, and the results of top-3 recommendation and top-5 recommendation for Wechat.

---

[11]The document can be found at https://docs.ray.io/en/master/tune/index.html. And we take its ASHAScheduler to perform the grid-search process.

Table 2. The overall top-N recommendation performance of different methods on Kwai and Wechat. Metric@N (Recall@10) denotes the corresponding top-N (top-10) recommendation performance on this metric (Recall). For each dataset, bold scores denote the best in each column, while the underlined scores denote the best baseline. "RI" refers to the relative improvement of DCR-MoE over the corresponding baseline, averaged on the three metrics. For all metrics, higher results are better.
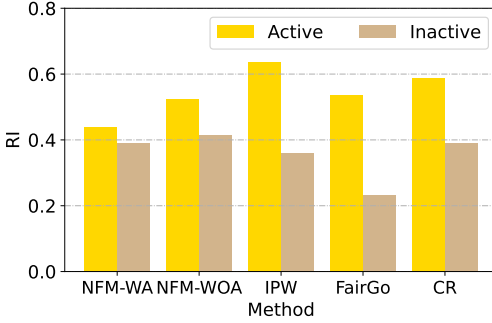
	Kwai							
Methods	Recall@10	MAP@10	NDCG@10	RI	Recall@20	MAP@20	NDCG@20	RI
NFM-WA	0.0778	0.0247	0.0458	40.4%	0.1530	0.0310	0.0694	30.7%
NFM-WOA	0.0759	0.0228	0.0439	47.6%	0.1494	0.0289	0.0672	36.4%
IPW	0.0775	0.0250	0.0464	39.5%	0.1474	0.0308	0.0684	33.2%
FairGo	0.0876	0.0269	0.0506	26.9%	0.1651	0.0332	0.0749	21.4%
CR	0.0764	0.0244	0.0458	41.8%	0.1492	0.0305	0.0687	33.0%
DCR-MoE	**0.1089**	**0.0353**	**0.0634**	-	**0.1936**	**0.0423**	**0.0896**	-
	Wechat							
Methods	Recall@3	MAP@3	NDCG@3	RI	Recall@5	MAP@5	NDCG@5	RI
NFM-WA	0.1275	0.0910	0.1198	6.5%	0.1584	0.0906	0.1341	4.2%
NFM-WOA	0.1324	0.0950	0.1243	2.4%	0.1609	0.0930	0.1370	2.0%
IPW	0.1326	0.0941	0.1240	2.8%	0.1600	0.0926	0.1366	2.5%
FairGo	0.1291	0.0924	0.1213	5.0%	0.1598	0.0813	0.1351	7.6%
CR	0.1282	0.0915	0.1205	5.9%	0.1584	0.0905	0.1345	4.1%
DCR-MoE	**0.1355**	**0.0976**	**0.1271**	-	**0.1646**	**0.0947**	**0.1396**	-

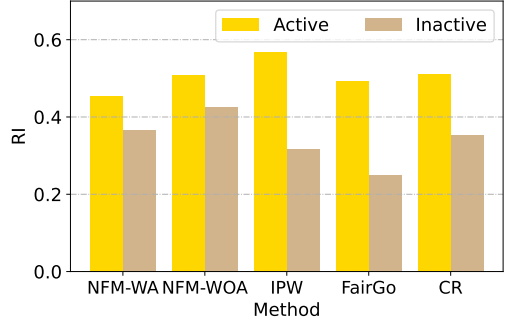## 5.2 RQ1: Performance Comparison

In this subsection, we study the recommendation performance of our DCR-MoE over all users as well as in different user subgroups.

5.2.1 Overall Performances. The overall performance comparison is summarized in Table 2 regarding the top-N recommendation. From the table, we have the following observations:
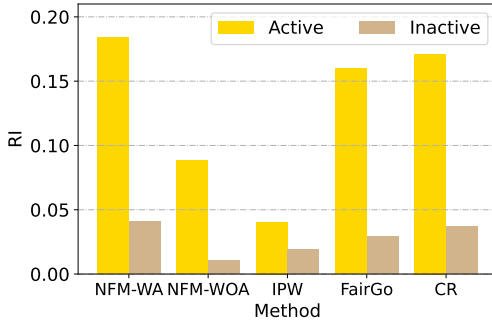
- The proposed method DCR-MoE achieves the best performance on both Kwai and Wechat datasets. This verifies the effectiveness of our deconfounding causal recommendation framework. The improvements can be attributed to the intervention performed at inference, *i.e.,* estimating the casual effect $P(Y|U, do(X))$ as user-item matching, making the recommendations free from the impact of the confounding feature.

- DCR-MoE consistently outperforms NFM-WOA and NFM-WA. NFM-WA models the correlation $P(Y|U, X, A)$, thus the direct effect of the confounding feature on interaction will be captured, misleading the model towards items with dominant values of the confounding feature (*cf.* Figure 7). The unsatisfactory results of NFM-WOA, which models the correlation $P(Y|U, X)$, demonstrate that simply removing the confounding feature during model training cannot solve the confounding feature issue. This is because the spurious correlation brought by the backdoor path will mislead the results. Summarily, when the confounding feature appears, it is better to model the user-item matching from the causal effect perspective instead of mere correlation.

- IPW and CR are both causal methods. However, neither of them demonstrates the power to deal with the confounding feature issue due to two drawbacks. 1) IPW relies on accurate estimation of propensities, which is non-trivial due to high variance [4, 56]. Moreover, its way to define propensity might not be suitable for the confounding feature. 2) To remove the undesired effect of the confounding feature, CR needs to precisely disentangle the effects of the
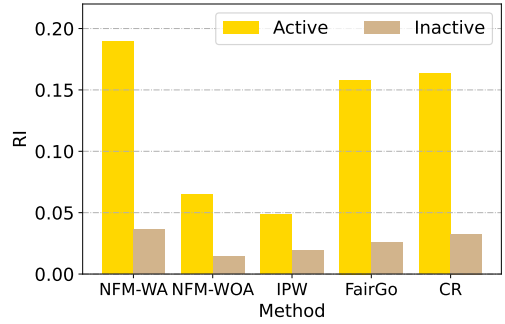
(a) **Recall@10 on Kwai**

(b) **NDCG@10 on Kwai**

(c) **Recall@3 on Wechat**

(d) **NDCG@3 on Wechat**

Fig. 4. Relative Improvements(RI) of DCR-MoE over different baselines on the active and inactive user groups, respectively. For Kwai, the results of Recall@10 and NFCG@10 are reported. For Wechat, the results of Recall@3 and NDCG@3 are reported. We omit the results of other metrics which show similar phenomena.

confounding feature and other features. However, disentangling is hard without supervision or inductive bias [21, 57]. Meanwhile, the fusion strategy of CR may be insufficient to represent the real fusion mechanism of the above two types of effects.

- The results of FairGo are also unsatisfactory where we postulate the reason to be information loss. The adversarial learning in FairGo tries to remove any information related to the confounding feature, which indeed removes partial information of X due to the relations $X \longleftarrow Z \longrightarrow A$. However, the removed information of X may be useful for user-item matching. In contrast, DCR-MoE keeps all information of X and merely eliminates the impact brought by the confounding feature.

- DCR-MoE achieves different degrees of improvements on Kwai and Wechat. We think the difference is due to different properties of the two datasets: 1) the interactions in Wechat are influenced by the social networks that are not given[12]; and 2) Wechat is more sparse regarding users.

---

[12]The dataset Wechat is a short video recommendation dataset collected from the China popular social media WeChat. Friends will influence each other to watch videos.

Table 3. The performance comparison between NFM-WA, MoE, and DCR-MoE. Both NFM-WA and MoE estimate the correlation P(Y|U,X,A), but MoE takes the proposed MoE model architecture. MoE and DCR-MoE have the same model size.

Datasets	Kwai			Wechat		
Methods	Recall@10	MAP@10	NDCG@10	Recall@3	MAP@3	NDCG@3
NFM-WA	0.0778	0.0247	0.0458	0.1275	0.0910	0.1198
MoE	0.0757	0.0240	0.0448	0.1286	0.0924	0.1215
DCR-MoE	0.1089	0.0353	0.0634	0.1355	0.0976	0.1271

5.2.2 Improvements in Active and Inactive User Groups. We have partially attributed the performance improvement differences of DCR-MoE on Kwai and Wechat to the sparsity of the datasets on the user side. We then investigate whether user activeness affects the performance of our DCR-MoE. For this goal, we first split users into active and inactive groups according to the number of user's positive testing label (*i.e.,* the positive post-playing feedback) and the number of user's training samples. The active group is the intersection of the following two user sets: 1) users with the top-ranked number of training samples (top 40% for Kwai and top 20% Wechat) and 2) users with the most positive testing labels (top 50% for Kwai[13] and top 20% for Wechat). Finally, the 20% and 6% of users are selected as active users in Kwai and Wechat, respectively. Users that are not categorized as active users are all in the inactive group. Then we evaluate different methods on the two groups respectively and compute the relative improvements (RI) of DCR-MoE over different methods. The results on Recall@10 and NDCG@10 for Kwai, on Recall@3 and NDCG@3 for Wechat, are shown in Figure 4. The results of other metrics are omitted since they show similar phenomena to the reported metrics.

According to the results in Figure 4, DCR-MoE can get improvements in both active and inactive groups. Moreover, DCR-MoE always achieves larger improvements in the active group. Particularly, the improvements in the active group of Wechat are at least twice as large as that in the inactive group. These results reflect the influences of data sparsity on estimating causal effect: with more data, it is more possible that there are more diverse combinations of X and A, thus $P(Y|U, X, A)$ can be estimated better on different values of X and A for user U; meanwhile, computing $P(Y|U, do(X))$ needs to enumerate all possible values of A; thus the estimation of $P(Y|U, do(X))$ regarding active users can be more accurate[14]. Second, there are more inactive users than active users in both datasets, ecpecially for Wechat. The total relative improvements are dominated by the inactive users since the metrics are averaged over all users. This also explains why the overall performance gain on Wechat dataset (*cf.* Table 2) is relatively marginal. Besides, we believe bringing more improvements for active users is also meaningful since the consumption of items is mostly generated by active users on real-world recommendation platforms.

5.3 RQ2: Ablation Studies

DCR-MoE has two main designs – the intervention at inference and the MoE model architecture. In this subsection, we conduct experiments to verify the function of these designs.

5.3.1 The Effectiveness of Intervention at Inference. Recall that the most important operation in our DCR framework is to perform intervention at the inference stage with *do-calculus*. We have shown DCR-MoE can achieve better recommendation performances. We question that the improvements of DCR-MoE may come from its more parameters instead of the intervention, because it has

---

[13]In Kwai, each user in the top 50% has great than 700 training samples.
[14]Indeed, data sparsity is related to the *overlap* assumption [40] for precise causal effect estimation.

Table 4. Inference cost comparisons between DCR-NFM, DCR-NFM-A, and DCR-MoE (the number of experts K=6). Results shown in EB/MLP column give the number of times that different methods need to run EB layers/MLPs in NFM or MoE. The Time column refers to actual time cost (seconds) at inference.

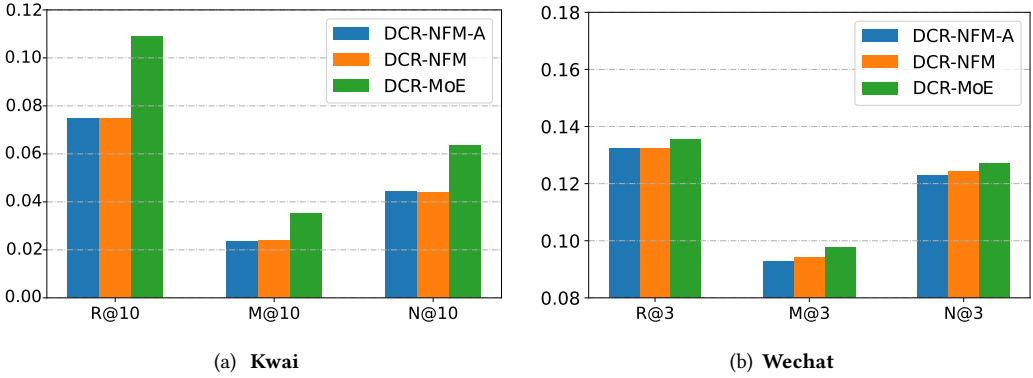| Datasets	Kwai			Wechat		
Methods	EB	MLP	Time(s)	EB	MLP	Time(s)
DCR-NFM	6	6	13.6	6	6	2.9
DCR-NFM-A	1	1	2.5	1	1	0.6
DCR-MoE	1	6	5.4	1	6	1.1

(a) **Kwai**

(b) **Wechat**

Fig. 5. Recommendation performance comparisons between different implementations for our DCR framework. Here, R, M and N denote Recall, MAP and NDCG, respectively.

multiple experts. In this light, we further evaluate a variant of DCR-MoE, named MoE, which directly takes the expert-specific output of the MoE model, *i.e.,* still using a one-hot vector to select one expert at inference, as the recommendation score. Obviously, MoE models the correlation $P(Y|U, X, A)$ as the user-item matching. Then we compare MoE with DCR-MoE, and NFM-WA that also models $P(Y|U, X, A)$ but with fewer model parameters. The comparison is shown in Table 3. We see the performance of MoE is not as good as that of DCR-MoE which conducts intervention at inference. And MoE has similar performances as NFM-WA on the both Kwai and Wechat, since they both estimate $P(Y|U, X, A)$ as user-item matching. These results verify that our improvements are coming from the intervention instead of the extra model parameters.

5.3.2 The Importance of the MoE Model Architecture. Recall that one motivation for designing MoE is to speed up the inference of the DCR. We compare DCR-MoE with: 1) the DCR-NFM that implements DCR based on NFM, *i.e.,* implementing DCR without the MoE architecture, and 2) the DCR-NFM-A that speeds up the inference by NWGM approximation (Equation 5), which can be seen as applying PD [56] to handle the confounding feature. We compare both the inference efficiency and the recommendation performance. Regarding the computation cost, we theoretically compare the number of times to calculate the Embedding-and-Bi-interaction (EB) layers and MLPs of NFM (or MoE) as well as report the actual running time for inference. To fairly compare the actual running time, we run all models on the same machine with an NVIDIA RTX 3060 GPU, an Intel i7-9700K CPU, and 16 GB of memory. Table 4 and Figure 5 show the comparisons regarding time cost and recommendation performance, respectively. From the table and figure, we have the following observations:
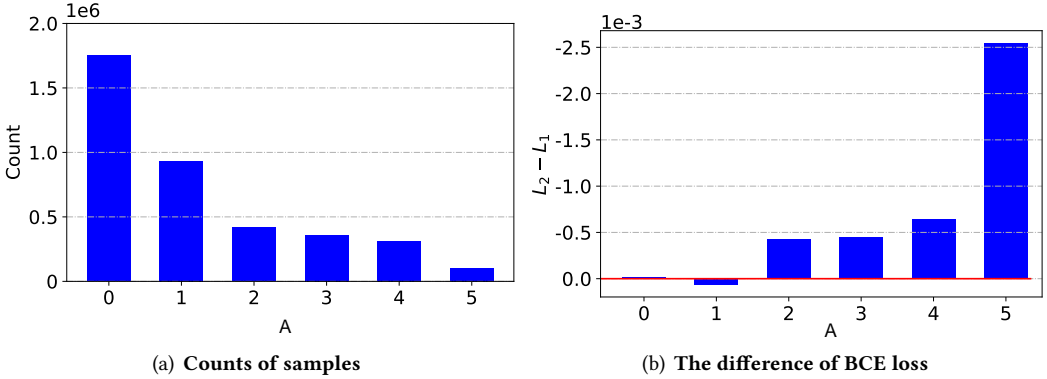
(a) **Counts of samples**

(b) **The difference of BCE loss**

Fig. 6. The advantages of MoE for fitting $P(Y|U, X, A)$ on different values of confounding feature A. (a) the counts of samples for different values of A (sorted according to the counts). (b) The difference $(L_2 - L_1)$ between the BCE loss of MoE (L_2) and the BCE loss of NFM-WA (L_1) on different A. Please note that a higher bar in (b) means a more negative value of $L_2 - L_1$.

- From Table 4, we find: 1) DCR-MoE is sub-optimal regarding the theoretical running cost. While DCR-MoE only requires $\frac{1}{K}$ (K=6) of DCR-NFM's EB layer calculations, DCR-MoE has more cost on running the MLPs than DCR-NFM-A. 2) The actual running time exhibits similar trend, which justifies the ability of MoE to speed up the inference of DCR. Note that MoE can keep using simple experts when using more complex backbone recommender models (*e.g.,* modeling high order feature interactions). We thus believe that the accelerating of MoE for inference will be more significant as compared to DCR-NFM when using more complex backbone. Meanwhile, the gap to DCR-NFM-A will be reduced in such cases.
- Although DCR-NFM-A achieves the best inference efficiency, its recommendation performance decreases on most metrics compared with DCR-NFM, as shown in Figure 5. The result can be attributed to the fact that DCR-NFM-A achieves the inference speedup by approximating computation, sacrificing the modeling fidelity. On the contrary, DCR-MoE can also increase the recommendation performance compared with DCF-NFM, ensuring the modeling fidelity.

The above two observations show that MoE can not only improve the efficiency of DCR but also improve its effectiveness. Note that our DCR-MoE has the similar training time cost to the single model methods, *e.g.,* NFM-WA and DCR-NFM, since each sample is only fed into one expert and used to update one expert. We do not study the training time cost here. With the training and inference efficiency and recommendation performance considerations, we believe MoE is an important design for DCR.

5.3.3 Why Does MoE Bring Improvements? We then investigate why the MoE architecture of DCR-MoE brings performance gains, especially on the dataset Kwai. Note that Kwai has an extremely unbalanced distribution of A, as shown in Figure 6(a). We postulate that MoE enhances the estimation of $P(Y|U, X, A)$ on different values of A, and further, a better estimation for the $P(Y|U, do(X))$, under an unbalanced distribution of A. In a single model (NFM-WA or DCR-NFM), the model parameters are dominated by the samples with the head (*i.e.,* more frequent) values of A since the training loss is dominated by them, making samples with the tail values of A cannot be well expressed. That means the learning of $P(Y|U, X, A)$ on the head values of A will disturb the

Table 5. The relative improvements (RI) of DCR-MoE to baselines NFM-WA and NFM-WOA in recommendation performance, when these models (including DCR-MoE and the baselines) are performed on confounding or non-confounding features. For Kwai and Wechat, the results for the top-20 recommendation and top-5 recommendation are reported, respectively.

Datasets	Kwai(top-20)		Wechat(top-5)	
Feature type	RI to NFM-WA	RI to NFM-WOA	RI to NFM-WA	RI to NFM-WOA
confounding feature	30.7%	36.4%	4.2%	2.0%
non-confounding feature	0.4%	3.7%	1.0%	-0.5%

learning of that on the tail values. However, our MoE takes different experts to represent different values of A, making the $P(Y|U,X,A)$ on tail values of A influenced by the head values less. Thus, MoE enhances the estimation of $P(Y|U,X,A)$.

To verify the above intuition, we compare the average BCE loss of MoE (denoted as L_2, MoE is the one defined in Section 5.3.1) and the average BCE loss of NFM-WA (denoted as L_1) over samples with the same values of A. Both MoE and NFM-WA are modeling the correlation $P(Y|U,X,A)$, so the smaller BCE loss means a better estimation for $P(Y|U,X,A)$. The results are shown in Figure 6(b). Together with Figure 6(a), we can find that MoE and NFM-WA have larger loss differences on the samples with more tail values of A ($A = 2, 3, 4, 5$), and the loss of NFM-WA is larger than the loss of MoE. Meanwhile, MoE has similar losses to NFM-WA on the head values ($A = 0, 1$). This shows the MoE model architecture can estimate $P(Y|U,X,A)$ on tail values of A better and keep the accuracy of $P(Y|U,X,A)$ on other head values. Due to the better estimation of $P(Y|U,X,A)$ on tail values of the confounding feature with MoE architecture, DCR-MoE can get a better estimation of the $P(Y|U, do(X))$, leading to the better recommendation performance.

5.3.4 Does DCR-MoE Really Address the Confounding Feature Issue? We further verify that the success of DCR-MoE comes from addressing the confounding feature issue, from another perspective. Specially, we study the improvements of DCR-MoE to basic baselines (NFM-WA and NFM-WOA) in recommendation performance, forcibly performing DCR-MoE and the baselines on non-confounding features[15]. If the improvements are still large, compared to the normal case where all models are performed on confounding features, the DCR-MoE may not address the confounding feature issue. Thus, we next compare the relative improvements (RI, defined in Table 2) of DCR-MoE to the baselines in the above two cases. The results are summarized in Table 5. From the table, we find that: DCR-MoE performed on non-confounding features shows far fewer (even negative) relative improvements to the corresponding NFM-WA and NFM-WOA, compared to the case where models are performed on confounding features. This verifies that the performance improvement of DCR-MoE (in the normal case) is brought by effectively addressing the confounding feature issue.

5.4 RQ3: Prediction Analyses

The performance study has shown the superiority of DCR-MoE regarding overall recommendation accuracy. In this subsection, we further study whether our DCR-MoE truly eliminates the impact of the confounding feature by conducting an analysis of the model predictions. Specifically, we first divide items into different groups according to the values of A, such that items with the same value of the confounding feature belong to a group. Then we average the prediction scores from DCR, NFM-WA, NFM-WOA, IPW, and CR over the samples that belong to the same group. Meanwhile,

---

[15]We manually select a feature having similar correlation coefficients to the training label and the testing label as the non-confounding feature, *i.e.,* $|\rho_1/\rho_2|$ (*cf.* Table 1) is near to 1. To perform models on the non-confounding feature, we forcibly replace the confounding feature A with the non-confounding feature.

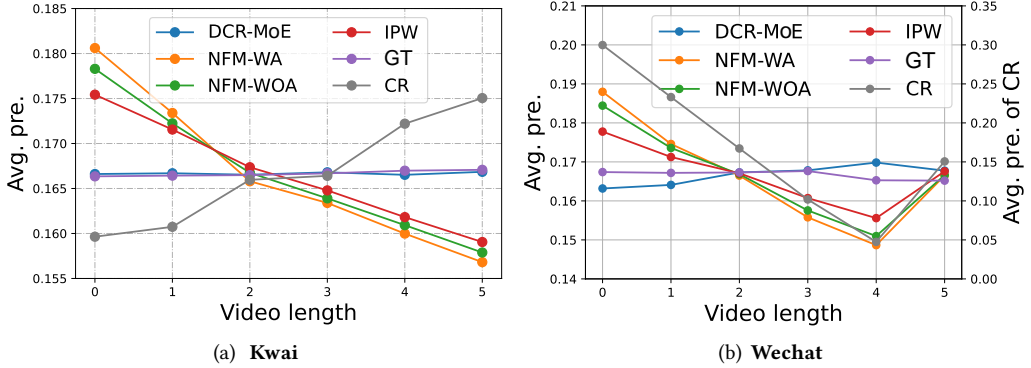(a) **Kwai**   (b) **Wechat**

Fig. 7. The average prediction (Avg. pre.) scores of different methods over samples in the same group, where the group is defined according to the value of the confounding feature, on Kwai and Wechat. "GT" is used to represent the ground-truth interest of the population to different groups.

we average the evaluation interaction label (*i.e.,* testing label) over the samples that belong to the same group, which can represent the ground-truth interest of the population to different item groups. The later result is denoted as GT. The comparison[16] between different methods is shown in Figure 7, where we have the following observations:

- The lines of GT are almost flat, which shows the true user-item matching is less related to the confounding feature, *i.e.,* video length. Meanwhile, the lines of DCR-MoE are most similar to the lines of GT that are taken as the ground truth of user interests, meaning DCR-MoE captures the true user-item matching. Indeed, for all user-item candidates, DCR-MoE forcibly changes their confounding feature to whatever possible values according to the same P(A) at inference, making the recommendations invariant with respect to the value of the confounding feature A.
- NFM-WOA and NFM-WA are both highly related to the confounding feature (since their lines are not flat). They are both biased towards some special groups, *e.g.,* giving higher scores to the group with the shortest video length. It is natural for NFM-WA since it captures the direct effect of A on Y for estimating user-item matching by modeling $P(Y|U,X,A)$. The biased results of NFM-WOA which models $P(Y|U,X)$ can be attributed to the spurious correlations brought by the backdoor path, and show that the impact of the confounding feature A will still be captured even if A is removed from the inputs.
- The lines of IPW are slightly flatter, compared with NFM-WA and NFM-WOA, but are still far from the lines of GT. This shows IPW can only slightly mitigate the biased recommendation, which can be attributed to the fact the propensity weights are not easy to be estimated well.
- The phenomenon of CR is rather strange. For the shortest video, CR has the lowest average prediction on Kwai, while it has the largest average prediction on Wechat. We think this is another evidence that CR cannot truly disentangle the effects of the confounding feature (undesired) and other features, leading to incorrectly removing effects.

Here we do not compare FairGo since it blindly removes all information related to the confounding feature and is straightforward to get flat lines by tuning hyper-parameters to remove more information.

---

[16]Since the scales of average prediction scores are different for different methods. We adopt the softmax function to deal with the results to better reflect the shape of distributions.

6 CONCLUSION

This work studies a new problem for dealing with the impact of item confounding feature on recommendation. We analyzed the problem from a causal view and recognized a backdoor path through the confounding feature, which results in spurious correlations between the remaining content features and the interaction behavior and biased recommendation. To remove the backdoor path, we proposed a DCR framework to estimate the causal effect of content features on the interaction, with a MoE architecture to speed up the inference. We prove in theory that DCR has good generality to deal with more cases of different causal relations between confounding feature and other features. We conduct experiments on two real-world datasets, providing insightful analyses on the rationality and effectiveness of our proposal.

This work shows the importance of causal modeling for item features in recommendation, but focuses on one confounding feature. In future, we will extend the proposed framework in the following directions: 1) handling multiple confounding features; 2) diving into the detailed causal relations among content features; and 3) incorporating the causal relations among user features. Moreover, we would like to explore the causal discovery problem in recommendation to get rid of the labor cost on causal graph construction.

ACKNOWLEDGMENTS

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*. ACM, 42–46.

[2] Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H Chi, et al. 2019. Fairness in Recommendation Ranking through Pairwise Comparisons. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2212–2220.

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

[4] 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).

[5] Jiawei Chen, Yan Feng, Martin Ester, Sheng Zhou, Chun Chen, and Can Wang. 2018. Modeling Users' Exposure with Social Knowledge Influence and Consumption Influence for Recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 953–962.

[6] Konstantina Christakopoulou, Madeleine Traverse, Trevor Potter, Emma Marriott, Daniel Li, Chris Haulk, Ed H Chi, and Minmin Chen. 2020. Deconfounding User Satisfaction Estimation from Response Rate Bias. In *Fourteenth ACM Conference on Recommender Systems*. ACM, 450–455.

[7] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research* 12, 7 (2011).

[8] Constantine E Frangakis and Donald B Rubin. 2002. Principal Stratification in Causal Inference. *Biometrics* 58, 1 (2002), 21–29.

[9] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. 1725–1731.

[10] Ruocheng Guo, Xiaoting Zhao, Adam Henderson, Liangjie Hong, and Huan Liu. 2020. Debiasing Grid-based Product Search in E-commerce. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2852–2860.

[11] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 355–364.

[12] Jin Huang, Harrie Oosterhuis, and Maarten de Rijke. 2022. It Is Different When Items Are Older: Debiasing Recommendations When Selection Bias and User Preferences Are Dynamic. In *Proceedings of the Fifteenth ACM International*

Conference on Web Search and Data Mining. ACM, 381–389.

[13] Rashidul Islam, Kamrun Naher Keya, Ziqian Zeng, Shimei Pan, and James Foulds. 2021. Debiasing Career Recommendations with Neural Fair Collaborative Filtering. In *Proceedings of the Web Conference 2021*. ACM / IW3C2, 3779–3790.

[14] Jurek Leonhardt, Avishek Anand, and Megha Khosla. 2018. User Fairness in Recommender Systems. In *Companion Proceedings of the The Web Conference 2018*. ACM, 101–102.

[15] Jie Li, Yongli Ren, and Ke Deng. 2022. FairGAN: GANs-based Fairness-aware Learning for Recommendations with Implicit Feedback. In *Proceedings of the ACM Web Conference 2022*. ACM, 297–307.

[16] Yunqi Li, Hanxiong Chen, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2021. User-oriented Fairness in Recommendation. In *Proceedings of the Web Conference 2021*. ACM / IW3C2, 624–632.

[17] Yunqi Li, Hanxiong Chen, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. 2021. Towards Personalized Fairness Based on Causal Notion. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1054–1063.

[18] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1754–1763.

[19] Dugang Liu, Pengxiang Cheng, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2020. A General Knowledge Distillation Framework for Counterfactual Recommendation via Uniform Data. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 831–840.

[20] 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*. ACM, 351–360.

[21] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. 2019. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. In *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 4114–4124.

[22] Rishabh Mehrotra, Prasanta Bhattacharya, and Mounia Lalmas. 2020. Inferring the Causal Impact of New Track Releases on Music Recommendation Platforms through Counterfactual Predictions. In *Fourteenth ACM Conference on Recommender Systems*. ACM, 687–691.

[23] Marcia L Meldrum. 2000. A Brief History of the Randomized Controlled Trial: From Oranges and Lemons to the Gold Standard. *Hematology/oncology Clinics of North America* 14, 4 (2000), 745–760.

[24] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. 2018. Ray: A Distributed Framework for Emerging AI Applications. In *13th USENIX Symposium on Operating Systems Design and Implementation*. USENIX Association, 561–577.

[25] Judea Pearl. 2009. *Causality*. Cambridge University Press.

[26] Jiaxin Qi, Yulei Niu, Jianqiang Huang, and Hanwang Zhang. 2020. Two Causal Principles for Improving Visual Dialog. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 10860–10869.

[27] Steffen Rendle. 2010. Factorization Machines. In *2010 IEEE International Conference on Data Mining*. IEEE Computer Society, 995–1000.

[28] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased Recommender Learning from Missing-not-at-random Implicit Feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. ACM, 501–509.

[29] Masahiro Sato, Sho Takemori, Janmajay Singh, and Tomoko Ohkuma. 2020. Unbiased Learning for the Causal Effect of Recommendation. In *Fourteenth ACM Conference on Recommender Systems*. ACM, 378–387.

[30] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as Treatments: Debiasing Learning and Evaluation. In *Proceedings of the 33nd International Conference on Machine Learning*. PMLR, 1670–1679.

[31] Wenjie Shang, Yang Yu, Qingyang Li, Zhiwei Qin, Yiping Meng, and Jieping Ye. 2019. Environment Reconstruction with Hidden Confounders for Reinforcement Learning based Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 566–576.

[32] Khanh Hiep Tran, Azin Ghazimatin, and Rishiraj Saha Roy. 2021. Counterfactual Explanations for Neural Recommenders. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1627–1631.

[33] Qi Wan, Xiangnan He, Xiang Wang, Jiancan Wu, Wei Guo, and Ruiming Tang. 2022. Cross Pairwise Ranking for Unbiased Item Recommendation. In *Proceedings of the ACM Web Conference 2022*. ACM, 2370–2378.

[34] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale Commodity Embedding for E-commerce Recommendation in Alibaba. In *Proceedings of the 24th ACM SIGKDD International*

Conference on Knowledge Discovery & Data Mining. ACM, 839–848.

[35] Wenjie Wang, Fuli Feng, Xiangnan He, Xiang Wang, and Tat-Seng Chua. 2021. Deconfounded Recommendation for Alleviating Bias Amplification. In The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. ACM, 1717–1725.

[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 Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 1288–1297.

[37] Wenjie Wang, Xinyu Lin, Fuli Feng, Xiangnan He, Min Lin, and Tat-Seng Chua. 2022. Causal Representation Learning for Out-of-Distribution Recommendation. In Proceedings of the ACM Web Conference 2022. ACM, 3562–3571.

[38] Xiangmeng Wang, Qian Li, Dianer Yu, Peng Cui, Zhichao Wang, and Guandong Xu. 2022. Causal Disentanglement for Semantics-Aware Intent Learning in Recommendation. IEEE Transactions on Knowledge and Data Engineering (2022).

[39] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2021. Combating Selection Biases in Recommender Systems with a Few Unbiased Ratings. In Proceedings of the 14th ACM International Conference on Web Search and Data Mining. ACM, 427–435.

[40] Yixin Wang and David M Blei. 2019. The Blessings of Multiple Causes. J. Amer. Statist. Assoc. 114, 528 (2019), 1574–1596.

[41] Yixin Wang, Dawen Liang, Laurent Charlin, and David M Blei. 2020. Causal Inference for Recommender Systems. In Fourteenth ACM Conference on Recommender Systems. ACM, 426–431.

[42] Yifan Wang, Weizhi Ma, Min Zhang, Yiqun Liu, and Shaoping Ma. 2022. A Survey on the Fairness of Recommender Systems. ACM Trans. Inf. Syst. (2022). Just Accepted.

[43] Zhenlei Wang, Shiqi Shen, Zhipeng Wang, Bo Chen, Xu Chen, and Ji-Rong Wen. 2022. Unbiased Sequential Recommendation with Latent Confounders. In Proceedings of the ACM Web Conference 2022. ACM, 2195–2204.

[44] Zhenlei Wang, Jingsen Zhang, Hongteng Xu, Xu Chen, Yongfeng Zhang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Counterfactual Data-augmented Sequential Recommendation. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 347–356.

[45] 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. ACM, 1791–1800.

[46] Chuhan Wu, Fangzhao Wu, Xiting Wang, Yongfeng Huang, and Xing Xie. 2021. Fairness-aware News Recommendation with Decomposed Adversarial Learning. In Proceedings of the AAAI Conference on Artificial Intelligence. AAAI Press, 4462–4469.

[47] Le Wu, Lei Chen, Pengyang Shao, Richang Hong, Xiting Wang, and Meng Wang. 2021. Learning Fair Representations for Recommendation: A Graph-based Perspective. In Proceedings of the Web Conference 2021. ACM / IW3C2, 2198–2208.

[48] Teng Xiao and Suhang Wang. 2022. Towards Unbiased and Robust Causal Ranking for Recommender Systems. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. ACM, 1158–1167.

[49] Jun Xu, Xiangnan He, and Hang Li. 2020. Deep Learning for Matching in Search and Recommendation. Found. Trends Inf. Retr. 14, 2-3 (2020), 102–288.

[50] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In Proceedings of the 32nd International Conference on Machine Learning. JMLR, 2048–2057.

[51] Mengyue Yang, Quanyu Dai, Zhenhua Dong, Xu Chen, Xiuqiang He, and Jun Wang. 2021. Top-N Recommendation with Counterfactual User Preference Simulation. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management. ACM, 2342–2351.

[52] Sirui Yao and Bert Huang. 2017. Beyond Parity: Fairness Objectives for Collaborative Filtering. In the Thirty-first Conference on Neural Information Processing Systems. 2921–2930.

[53] Kun Zhang and Aapo Hyvärinen. 2009. On the Identifiability of the Post-Nonlinear Causal Model. In Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. AUAI Press, 647–655.

[54] Shengyu Zhang, Dong Yao, Zhou Zhao, Tat-Seng Chua, and Fei Wu. 2021. Causerec: Counterfactual User Sequence Synthesis for Sequential Recommendation. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 367–377.

[55] Xiao Zhang, Haonan Jia, Hanjing Su, Wenhan Wang, Jun Xu, and Ji-Rong Wen. 2021. Counterfactual Reward Modification for Streaming Recommendation with Delayed Feedback. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 41–50.

[56] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. 2021. Causal Intervention for Leveraging Popularity Bias in Recommendation. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 11–20.

[57] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling User Interest and Conformity for Recommendation with Causal Embedding. In Proceedings of the Web Conference 2021. ACM / IW3C2, 2980–2991.

[58] Chang Zhou, Jianxin Ma, Jianwei Zhang, Jingren Zhou, and Hongxia Yang. 2021. Contrastive Learning for Debiased Candidate Generation in Large-scale Recommender Systems. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. ACM, 3985–3995.

[59] Shengyu Zhu, Ignavier Ng, and Zhitang Chen. 2020. Causal Discovery with Reinforcement Learning. In *8th International Conference on Learning Representations*.

[60] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. 2021. Popularity-opportunity Bias in Collaborative Filtering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM, 85–93.

[61] Ziwei Zhu, Xia Hu, and James Caverlee. 2018. Fairness-aware Tensor-based Recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1153–1162.