

CIRS: Bursting Filter Bubbles by Counterfactual Interactive Recommender System

Chongming Gao¹, Wenqiang Lei², Jiawei Chen¹, Shiqi Wang³, Xiangnan He^{1,*},
Shijun Li¹, Biao Li⁴, Yuan Zhang⁴, Peng Jiang⁴

¹University of Science and Technology of China; ²Sichuan University, China;

³Chongqing University, China; ⁴Kuaishou Technology Co., Ltd.

chongming.gao@gmail.com, wenqianglei@gmail.com, cjwustc@ustc.edu.cn, shiqi@cqu.edu.cn, xiangnanhe@gmail.com,
lishijun@mail.ustc.edu.cn, biaoli6@139.com, yuanz.pku@gmail.com, jp2006@139.com,

ABSTRACT

While personalization increases the utility of recommender systems, it also brings the issue of *filter bubbles*. E.g., if the system keeps exposing and recommending the items that the user is interested in, it may also make the user feel bored and less satisfied. Existing work studies filter bubbles in static recommendation, where the effect of overexposure is hard to capture. In contrast, we believe it is more meaningful to study the issue in interactive recommendation and optimize long-term user satisfaction. Nevertheless, it is unrealistic to train the model online due to the high cost. As such, we have to leverage offline training data and disentangle the causal effect on user satisfaction.

To achieve this goal, we propose a counterfactual interactive recommender system (CIRS) that augments offline reinforcement learning (offline RL) with causal inference. The basic idea is to first learn a causal user model on historical data to capture the overexposure effect of items on user satisfaction. It then uses the learned causal user model to help the planning of the RL policy. To conduct evaluation offline, we innovatively create an authentic RL environment (KuaiEnv) based on a real-world fully observed user rating dataset. The experiments show the effectiveness of CIRS in bursting filter bubbles and achieving long-term success in interactive recommendation. The implementation of CIRS is available via <https://github.com/chongminggao/CIRS-codes>.

KEYWORDS

Filter bubble, Interactive recommendation, Causal inference, Offline reinforcement learning

ACM Reference Format:

Chongming Gao¹, Wenqiang Lei², Jiawei Chen¹, Shiqi Wang³, Xiangnan He^{1,*}, Shijun Li¹, Biao Li⁴, Yuan Zhang⁴, Peng Jiang⁴. 2022. CIRS: Bursting Filter Bubbles by Counterfactual Interactive Recommender System. In *Preprint '22*. ACM, New York, NY, USA, 11 pages.

* Corresponding author.

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

Preprint '22, .

© 2022 Association for Computing Machinery.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...\$15.00

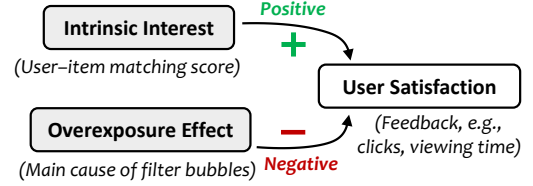


Figure 1: Disentangling causal effects on user satisfaction.

1 INTRODUCTION

“Personalization filters serve up a kind of invisible autopropaganda, indoctrinating us with our own ideas, amplifying our desire for things that are familiar and leaving us oblivious to the dangers lurking in the dark territory of the unknown.”

—Eli Pariser, *The Filter Bubble* [42]

Recommender systems have deeply affected our lives. They change the way of retrieving information from searching strenuously, to obtaining conveniently via the precise personalization. The system usually achieves personalization by learning on the collected behavior data of users and selecting the products that users potentially like [13, 16, 17, 27]. With time evolving and data accumulating, the recommender gradually becomes a mirror reflecting each user’s interest and narrows down the recommendation lists to the items with the maximum user interest. However, this comes at a price. While enjoying the precise personalized recommendations, users have to face the fact that the variety of information shrinks. When users become isolated from the information that varies from their dominant preferences, they are stuck in the so-called “filter bubbles”.

Filter bubbles are common in recommender systems. Recent studies conducted extensive experiments in large-scale recommender systems and found there are two primary causes of filter bubbles [20, 34, 40, 44, 51, 53]. From users’ perspective, those who have less diverse preferences are more liable to be stuck in the bubbles. From the system’s perspective, learning can lead to emphasizing the dominant interest of a user. Moreover, the system usually assumes that user satisfaction equals intrinsic interest — even though the interested items have been overexposed, it assumes that the user satisfaction remains unchanged, which is improper. We believe that such an overexposure effect is the main cause of filter bubbles.

Overexposing items has a pernicious effect on a user’s satisfaction, even though the user is interested in the recommended item. For example, a user who likes dancing can be satisfied when she

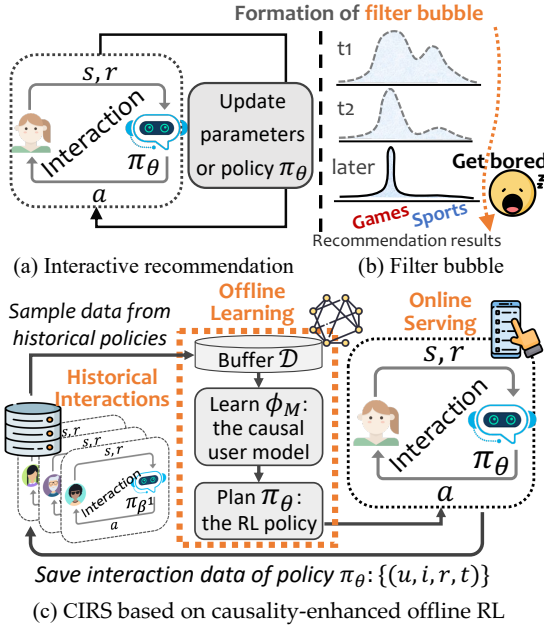


Figure 2: Illustration of interactive recommendation and the formation of the filter bubble.

receives a recommendation about a dancing song. However, she may be bored after dozens of times of incessant recommendations about dancing songs, and thus refuses to choose it. Therefore, it is of great significance to burst filter bubbles for a recommender system for the purpose of maximizing user satisfaction. The key is to disentangle the causal effects on user satisfaction, i.e., modeling how a user’s intrinsic interest together with the overexposure effect affect the user’s final satisfaction (Fig. 1).

Existing methods address filter bubble with two strategies: (1) helping users improve their awareness of diverse social opinions [9, 14], and (2) making the model enhance diversity [1, 35, 54], serendipity [41, 65], or fairness [38] of the recommendation results. However, most of these strategies are heuristic and focus on the static recommendation setting. They cannot fundamentally address the problem since they do not capture the main cause of filter bubble, i.e., the overexposure effect.

In this work, we focus on the interactive recommender system (IRS) in the dynamic environment. An IRS is formulated as a sequential decision making process, which allows the tracking and modeling of the dynamic and real-time overexposure effect. Fig. 2(a) draws an example of interactive recommendation, where a model recommends items (i.e., makes an action a) to a user based on the interaction context (i.e., state s reflecting the user’s information and interaction history), then it receives user feedback (i.e., reward r representing user satisfaction). The interaction process is repeated until the user quits. The model will update its policy π_θ with the goal to maximize the cumulative satisfaction over the whole interaction process. To achieve this goal, the IRS should avoid overexposing items because users will get bored and their satisfaction will drop in filter bubbles (Fig. 2(b)).

Though appealing, this idea faces an inevitable challenge: it is difficult to learn an IRS online with real-time feedback. The reason is twofold: (1) for the model, training the policy online increases the learning time and the deploying complexity [64]; (2) for the user, interacting with a half-baked system can hurt satisfaction [11, 46]. Hence, it is necessary to train the IRS offline on historical logs before serving users online. To this end, we need to conduct causal inference [43] on the offline data to disentangle the causal effect of user interest and overexposure. This provides an opportunity to answer a counterfactual question in the serving phase: “Would the user still be satisfied with the interested item if it had been overexposed?” – If the answer is no, then the filter bubble will occur once recommending the item, so we should not recommend it.

We propose a counterfactual interactive recommender system (CIRS) to achieve this goal. It enhances offline reinforcement learning (offline RL) [28] with causal inference [43]. Fig. 2(c) shows its learning framework, which contains three recurrent steps: 1) learning a causal user model to capture both user interest and item overexposure effect, 2) using the learned user model to provide *counterfactual satisfaction* (i.e., the rewards given by the causal user model instead of immediate users’ feedback) for planning an RL policy, and 3) evaluating the RL policy in terms of the cumulative user satisfaction in real environments.

In addition, we propose an interactive recommendation environment, KuaiEnv, for evaluating the problem. It is created based on the KuaiRec dataset that contains a fully filled user–item matrix (density: 99.6%) collected from the Kuaishou App¹ [12]. In this matrix, IRSs can be evaluated more faithfully since there are no missing values. To effectively reflect the effect of filter bubbles, we further add a “feel bored then quit” exit mechanism in both Virtual-Taobao and KuaiEnv to simulate the reactions of real users. The experiments show that the proposed CIRS can capture the overexposure effect in offline learning and achieve a larger cumulative satisfaction compared with other baselines.

Our contributions are summarized as follows:

- To the best of our knowledge, this is the first work to address filter bubbles in interactive recommendation, where filter bubbles can be more naturally observed and evaluated via modeling the overexposure effect on user satisfaction.
- We solve the problem by integrating causal inference into offline RL. We are the first to combine causal inference and offline RL in interactive recommender systems.
- We conduct empirical studies on Kuaishou App to verify that overexposing items indeed hurts user experience, and we demonstrate that the proposed method can burst filter bubbles and increase the cumulative satisfaction.

2 RELATED WORK

We briefly review the related works from three perspectives: filter bubbles, causality inference, and offline RL in recommendation.

2.1 Filter Bubbles in Recommendation

The term filter bubble was coined by internet activist Eli Pariser [42] and used for describing the state of intellectual isolation that

¹<https://www.kuaishou.com/cn>

results from algorithm’s personalization process. When a user is stuck in a filter bubble, the algorithm will show the user limited information that conforms with their preexisting belief or interest.

In recommendation, many works systematically analysed the causes of filter bubbles. There are in general two causes. The first cause may be users’ personal features [20, 34, 44, 51]. For example, Liu et al. [34] conducted simulation studies on news recommendations and found that users with more extreme preferences are more liable to be stuck in filter bubbles. Additionally, Hussein et al. [20] studied misinformation filter bubbles on YouTube, the most popular video-sharing platform, and found that personalized information, e.g., age, gender, and watch history, can indeed have a significant effect on amplifying misinformation filter bubbles.

The second cause is the overexposure effect. Recent empirical studies verified that the formation of filter bubbles is mainly due to that the model emphasizes certain items with dominant user interest [34, 53]. This kind of filter bubbles can have a pernicious effect on user satisfaction. In this work, we focus on modeling and alleviating the overexposure effect to address the filter bubble issue.

Existing methods that aim to combat filter bubbles focus on improving users awareness of diverse social opinions [9, 14], enhancing models’ diversity [1, 35, 54], serendipity [41, 65], fairness [38] of recommendation results, correcting model behavior by watching debunking content [53]. However, these strategies focus on the static recommendation setting, where the dynamic nature of filter bubbles is hard to model. In contrast, we capture the overexposure effect, i.e., the main cause of filter bubbles, and solve the problem in interactive recommendation.

2.2 Causality-enhanced Recommendation

Recently, causal inference (CI) has drawn a lot of attention in neural language processing (NLP) [10], computer vision (CV) [36, 57] and recommender system (RS) [4, 30, 47, 60, 66, 69]. Instead of exploiting the correlation relationships between input and output by feeding data to the black-box neural networks, CI explicitly models the causal mechanism among variables [18, 43]. In recommender systems, CI can be a powerful tool for addressing various biases in the data or the learning process, e.g., selection bias and popularity bias [5]. For example, many studies tried to estimate the unbiased user preference based on inverse propensity scoring (IPS) [45, 47], which is an effective CI-based method. Intuitively, when a user has a lower probability of seeing an item, we should assign increased importance to this sample and vice versa. However, IPS-based causal methods suffer from the high variance issue and it is difficult to estimate the propensity score [45].

Recently, researchers have followed Pearl’s causal inference framework [43] in RS [58, 63, 66, 69]. We generally summarize the procedure of these studies as three steps: 1) Constructing a causal graph, i.e., a representation of the structure causal model (SCM) that describes the causal relationship among the related variables. 2) In the learning stage, fitting an unbiased model (e.g., implemented as a neural network) on the training dataset based on the proposed causal graph. (3) In the inference stage, actively changing certain variables (called *intervention*) according to the certain requirement, then predicting the unbiased result of the target variable. In this work, we use this framework because it enables

Table 1: Six Types of Recommender Systems

	User Model Debiasing RL-based			Publications
Static RS	✓			[16, 17, 27, 67]
Unbiased static RS	✓	✓		[29, 31, 45, 47, 58, 69, 73, 75]
Traditional IRS			✓	[32, 33, 35, 62, 68, 74, 76, 78]
Model-based IRS	✓		✓	[3, 8, 59, 71, 72, 77]
OPE-based IRS		✓	✓	[6, 21–23, 37, 39, 52, 61]
Unbiased model-based IRS	✓	✓	✓	[7, 19] CIRS (Ours)

the model to remove the effect of confounding factors [58, 63], or answer the counterfactual question [69] in recommendation.

2.3 Offline RL for Recommendation

Recommender systems are designed to enhance user satisfaction or increase sales when serving online users. We usually consider recommendation as a sequential decision making process, where the recommender policy decides to make recommendations according to previous user feedback. Common studies on static recommendation models (e.g., DeepFM [16] and LightGCN [17]) only pay attention to improving the performance in single-round recommendation. To directly gain the long-term success in the multi-round decision making problem, we can adopt the interactive recommender system (IRS) based on reinforcement learning (RL) [11, 33, 35, 56, 68, 74, 76, 78]. Though effective, learning an IRS policy with online users is impractical as it is too slow and will hurt user experience [15]. On the other hand, the recorded recommender logs and offline user feedback are easier to obtain [25]. Therefore, it is natural to think of learning a policy on the offline data, which is the core idea of offline RL [28].

Commonly used offline RL strategies include off-policy evaluation (OPE) learning [52] and model-based RL methods [8, 77]. However, the former one has the high-variance problem and the latter suffers from bias issues [28]. As mentioned above, CI is a powerful tool to address the bias problem in recommender system. Hence, we combine the model-based offline RL and the CI technique to develop an unbiased IRS that can recognize and address the filter bubble problem on the offline data. Here, we briefly summarized in Table 1 six types of recommenders with respect to three dimensions: (1) whether the system explicitly builds a user model trying to capture real user preference, (2) whether the system considers debiasing, and (3) whether the system has an RL-based policy. Note that there are two other works lie in the same category as our CIRS, but they are not designed for the filter bubble problem.

3 PREREQUISITES

In this section, we introduce the problem definition and the empirical analyses of the real-world data from Kuaishou.

3.1 Problem Definition

We denote the user set as \mathcal{U} and item set as \mathcal{I} . The set of all interaction sequences of a user $u \in \mathcal{U}$ can be denoted as $\mathcal{D}_u = \{\mathcal{S}_u^1, \mathcal{S}_u^2, \dots, \mathcal{S}_u^{|\mathcal{D}_u|}\}$. Each $\mathcal{S}_u^k \in \mathcal{D}_u$ is the k -th interaction sequence (i.e., trajectory) recording a complete interaction process:

$S_u^k = \{(u, i_l, t_l)\}_{1 \leq l \leq |S_u^k|}$, where the user u begins to interact with the system at time t_1 and quits at time $t_{|S_u^k|}$, and $i_l \in \mathcal{I}$ is the recommended item at time t_l . Let $\mathbf{e}_u \in \mathbb{R}^{d_u}$ and $\mathbf{e}_i \in \mathbb{R}^{d_i}$ be the feature representation vectors of user u and item i , respectively. For the system, the task is to recommend items to users based on their preferences and interaction history. This process can be cast as a reinforcement learning problem, whose key components are summarized as follows:

- **State.** The system maintains a state $\mathbf{s}_t \in \mathbb{R}^{d_s}$ at time t is regarded as a vector representing information of all historical interactions between user u and the system prior to t . In this paper, we obtain the \mathbf{s}_t by using the Transformer model [55].
- **Action.** The system makes an action a_t at time t is to recommend items to user u . Let $\mathbf{e}_{a_t} \in \mathbb{R}^{d_a}$ denote the representation vector of action a_t . In this paper, each action a_t recommends only one item i . Hence, we have $\mathbf{e}_{a_t} = \mathbf{e}_i$.
- **Reward.** The user u returns feedback as a reward score r_t reflecting its satisfaction after receiving a recommended item i . The reward can also be the *counterfactual satisfaction* predicted by the causal user model ϕ_M instead of real users' feedback.
- **Policy network.** The policy network optimizes the target policy $\pi_\theta = \pi_\theta(a_t | \mathbf{s}_t)$ that represents the probability of making an action a_t conditioned on the state \mathbf{s}_t . It decides how to generate an item i to recommend and is implemented as a fully-connected neural network.

To make full use of the historical data, we base our method CIRS on the offline RL framework. Learning CIRS requires three recurrent steps (as illustrated by three color blocks in Fig. 4):

- (1) Training a causal user model ϕ_M on historical interaction data $\{(u, i, r)\}$ to estimate not only user interest but also the effect of item overexposure.
- (2) Using the learned causal user model ϕ_M (instead of real users) to train policy π_θ . In each interaction loop, ϕ_M samples a user u to interact with π_θ . When π_θ makes an action a_t (recommends i), ϕ_M provides a *counterfactual satisfaction* as the reward r . Intuitively, if π_θ has made similar recommendations before, ϕ_M shrinks the reward r .
- (3) Serving the learned policy π_θ to real users and evaluating the results in the interactive environment. When the interaction ends, we save the log $\{u, i, r, t\}$ to historical data for the purpose of future learning.

The three steps can be conducted repeatedly to continuously improve ϕ_M and π_θ .

3.2 Field Study of Overexposure Effect on User Satisfaction

As mentioned above, besides users' responsibility, the cause of filter bubbles is that the model incessantly recommends similar items to users. We suppose this is pernicious as users may not feel comfortable under such circumstances. Thus, we present a hypothesis as follows:

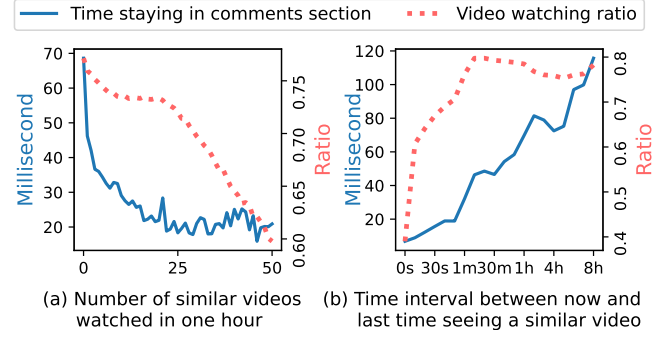


Figure 3: Empirical studies of exposure effect on Kuaishou. Statistics of two metrics vary with the exposure effect.

HYPOTHESIS 1. *Users' satisfaction will drop if the recommended items (or similar items) are repeatedly exposed and recommended to them in a short time.*

To verify this hypothesis, we conduct empirical studies on real data from Kuaishou, a video-sharing mobile App. We chose 7, 176 users from the video-watching user pool of the platform. We filter and collect their interaction history from August 5th, 2020 to August 11st, 2020. There are 34, 215, 294 views in total and 3, 110, 886 videos were watched. Each item is tagged with at least one and at most four category tags, and there are 31 category tags in total.

During watching a video, users can choose to quit watching at anytime by scrolling down to the next video or leaving the video-playing interface. Each video has a comments section where users can enter by clicking the "comment" button. If users are interested in a video, they will stay watching it for a longer time or enter its comments section. Therefore, we design two key metric indicators to reflect user satisfaction: One is the time duration staying in comments section, and the other is the video watching ratio, which is the ratio of viewing time to the total video length.

To show how item exposure affects user satisfaction, we study how the two indicators change with the degree of overexposure effect. Specifically, we group all collected views with respect to (a) the number of videos with the same tag watched in one hour, or (b) the time interval between now (watching this video) and the last time viewing a video with the same tag. Then we compute the average values of the mentioned indicators in every group. The results are shown in Fig. 3. Two observations are found:

OBSERVATION 1. *User satisfaction towards a recommended item drops when the system increases the number of similar items in recent recommendations.*

OBSERVATION 2. *User satisfaction towards a recommended item drops as the time interval between two similar items is shortened.*

The result is statistically significant since even the smallest group contains enough points: 31, 277 points for the group at $x = 50$ in Fig. 3 (a) and 76, 303 points for the group at $x = (7h \sim 8h)$ in Fig. 3 (b). Therefore, HYPOTHESIS 1 is empirically proved. This suggests that the filter bubble, i.e., the overexposure effect of the recommendation algorithm, does have a pernicious impact on user satisfaction. Consequently, we can design a "feel bored then quit"

exit mechanism in the constructed environments as a reflection of user behavior, which enables us to simulate the effect of filter bubbles effectively. We will illustrate it in Section 5.

4 PROPOSED METHODS

In this section, based on the observations obtained in the field studies, we propose a counterfactual interactive recommender system (CIRS) that leveraging causal inference in offline RL. We first introduce CIRS's three main modules in the offline RL framework, then we describe how to leverage causal inference to disentangle the causal effects on user satisfaction.

4.1 Offline RL-based Framework

We base our CIRS model on offline RL, where we can utilize the large amount of offline data to train the interactive recommender system. The framework of CIRS is illustrated in Fig. 4. It contains three stages (shown in three color blocks): pre-learning stage, RL planning stage, and the RL evaluation stage. The functions of these three stages correspond to the three: (1) pre-learning the user model ϕ_M via supervised learning, (2) using the learned user model ϕ_M to learn RL policy π_θ by providing *counterfactual satisfaction* as the reward, and (3) evaluating policy π_θ in the real environment. Specially, the real environment can either be the real-world online environment or the simulation environment that can reflect real user behavior. Next, we separately introduce the three main components in CIRS: causal user model ϕ_M , the state tracker module, and RL-based interactive policy π_θ .

4.1.1 Causal User Model. The causal user model ϕ_M learns user interest based on the historical data and provide the counterfactual satisfaction r for the RL planning stage. It aims to explicitly disentangle the causal effect by correctly modeling the effect of item overexposure on user satisfaction. There are two sub-modules in the user model: an interest estimation module designed for computing the user's intrinsic interest y , and a counterfactual satisfaction estimation module capturing how the overexposure effect affects user satisfaction r . The details of this component and will be reported in Section 4.2, where we will formally introduce the causal view of the recommender.

4.1.2 Transformer-based State Tracker. The state s_t in the interactive recommendation should include all critical information at time t for policy π_θ to make decisions. That includes: the feature vector \mathbf{e}_u of the user u , the feature vectors of the recently recommended items, i.e., actions of the system in the interaction loop $\{\mathbf{e}_{a_1}, \dots, \mathbf{e}_{a_t}\}$, and the user's feedback towards them $\{r_1, \dots, r_t\}$. In order to automatically extract key information from these vectors, we use the Transformer model [55] to derive s_t as illustrated in Fig. 4. Transformer is a state-of-the-art sequence-to-sequence model with an attention mechanism that can capture the dependence between current input and previous input sequences [70]. In this work, we use only a two-layer encoder of Transformer. Since the input is generated sequentially, we need to add a mask to prevent future information leaking into each state of the sequence.

We further use a gate mechanism to filter information from the action \mathbf{e}_{a_t} and user feedback r_t . Hence, the input for Transformer at time t is $\mathbf{e}'_{a_t} := \mathbf{g}_t \odot \mathbf{e}_{a_t}$. Where ' \odot ' denotes the element-wise

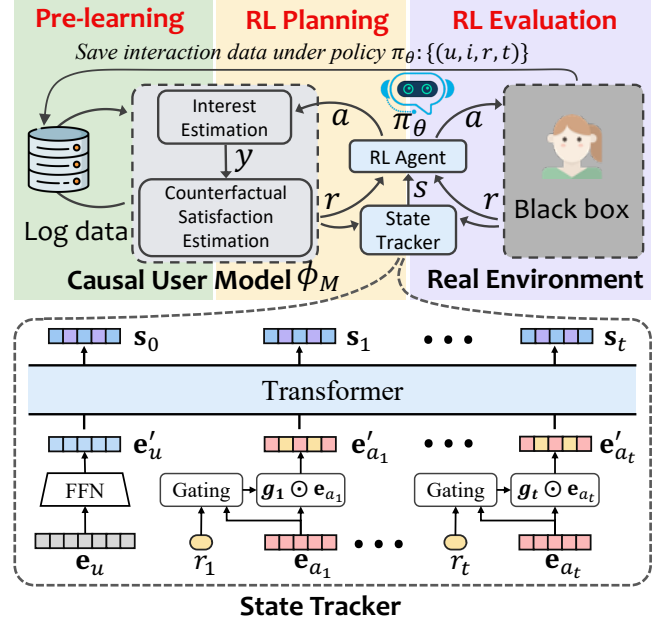


Figure 4: Learning Framework of Counterfactual IRS.

product, and the gating vector \mathbf{g}_t is computed as:

$$\mathbf{g}_t = \sigma(\mathbf{W} \cdot \text{Concat}(\mathbf{r}_t, \mathbf{e}_{a_t}) + \mathbf{b}), \quad (1)$$

where $\mathbf{W} \in \mathbb{R}^{d_s \times (1+d_a)}$ and $\mathbf{b} \in \mathbb{R}^{d_s}$ refers to the weight matrix and bias vector, respectively. $\text{Concat}(\cdot)$ is the vector concatenation operator. Besides, we use a feed-forward network (FFN) to transform the representation vector of user u from $\mathbf{e}_u \in \mathbb{R}^{d_u}$ to $\mathbf{e}'_u \in \mathbb{R}^{d_s}$ to let it lie in the same space of \mathbf{e}'_{a_t} .

The feature vectors, the parameters in Transformer and the gate are initialized randomly and trained in the end-to-end manner using the gradients passed from the RL model.

4.1.3 RL-based Interactive Recommendation Policy. We implement our interactive recommendation policy π_θ as the PPO algorithm [49]. PPO is a powerful on-policy reinforcement learning algorithm based on actor-critic framework [26]. It can work in both discrete and continuous state space and action space. We aim to maximize the cumulative user satisfaction of the long-term interaction, which can be realized by maximizing the objective function of PPO:

$$\mathbb{E}_t \left[\min \left(\frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \hat{A}_t, \text{clip} \left(\frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)}, 1 - \epsilon, 1 + \epsilon \right) \hat{A}_t \right) \right], \quad (2)$$

where ϵ is the hyperparameter that controls the maximum percentage of change that can be updated at one time. The function $\text{clip}(x, a, b)$ clips the variable x in the range of $[a, b]$. θ_{old} is the policy before updating, i.e., the interaction data are generated under policy θ_{old} . And the advantage function \hat{A}_t is implemented as the generalized advantage estimator (GAE) [48] which can be generally understood as a quantity that is proportional the normalized cumulative rewards of a sequence $\sum_{l=0}^{|S|} \gamma^l r_{t+l}$. For more details, we refer the audiences to the original PPO algorithm [49].

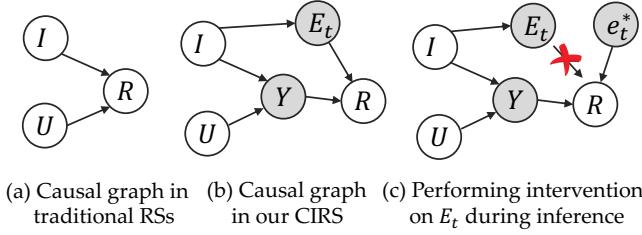


Figure 5: Causal graphs of traditional IRS models (a) and the CIRS model (b-c). U : a user, I : an item, R : user satisfaction (quantified by feedback such as clicks or viewing time), Y : intrinsic interest, E_t and e_t^* : the overexposure effect. Latent variables are shaded.

At last, to learn a good policy, we need the counterfactual satisfaction given by ϕ_M to be as correct and constructive as possible. Next, we will introduce how to build a causal user model ϕ_M to capture the overexposure effect thus avoiding the filter bubble.

4.2 Causal Inference-based User Satisfaction Disentanglement

We illustrate the causal graphs of the traditional recommender systems and our CIRS model in Fig. 5.

- Node U represents a certain user u , e.g., an ID or the profile feature that can represent the user.
- Node I represents an item i that is recommended to user u .
- Node R represents user u 's real-time satisfaction for the recommended item i . It is the feedback such as a click or the video watching ratio.
- Node Y represents the user's intrinsic interest that is static regardless of item overexposure.
- Node E_t and e_t^* represent the overexposure effect of item i on user u . E_t is a random variable and e_t^* is the value of E_t computed in the inference stage (i.e., the RL planning stage).

Traditional recommender systems will fit user satisfaction R based on solely the feature information of user U and item I (Fig. 5 (a)). This assumes that user satisfaction equals to intrinsic interest, which is improper as stated before.

Our proposed CIRS model innovatively takes into account the overexposure effect E_t , and disentangles the causal effect on user satisfaction R . Concretely, R is generated from two causal paths:

- (1) $(U, I) \rightarrow Y \rightarrow R$: This path projects user u and item i to their corresponding intrinsic interest y_{ui} . Then user satisfaction r_{ui} is proportional to y_{ui} .
- (2) $I \rightarrow E_t \rightarrow R$: This path captures the real-time overexposure effect $e_t(u, i)$ of an item i on user u 's satisfaction r .

Intrinsic Interest Estimation. We estimate the user's intrinsic interest y_{ui} as $\hat{y}_{ui} = f_{\theta}(u, i)$. The estimation model $f_{\theta}(u, i)$ can be implemented by various established recommendation models, such as DeepFM [16] used in this work. We illustrated this part in the interest estimation module in Fig. 4.

Overexposure Effect Definition Considering that overexposure effect negatively affects user satisfaction, we define the overexposure effect e_t of recommending an item i to user u at time t as:

$$e_t := e_t(u, i) := \alpha_u \beta_i \sum_{(u, i_l, t_l) \in S_u^k, t_l < t} \exp\left(-\frac{t - t_l}{\tau} \times \text{dist}(i, i_l)\right), \quad (3)$$

where $\text{dist}(i, i_l)$ is distance between two items i and i_l . α_u represents the *sensitivity* of user u to the overexposure effect, e.g., a user with a large α_u is more likely to feel bored when overexposed to similar content. Likewise, β_i represents the *unendurableness* of item i . For example, a classical music might be more endurable than a pop song, so β_i of the classical music is smaller. τ is a temperature hyperparameter.

User Satisfaction Estimation. Generally, a similar item i_l (i.e., with smaller $\text{dist}(i, i_l)$) that was recommended recently (i.e., with smaller $t - t_l$) contributes a larger overexposure effect to item i . And e_t is the sum of the effect of all items recommended to the user u before time t . After obtaining the overexposure effect e_t via Eq. (3) and intrinsic interest \hat{y}_{ui} via DeepFM, we estimate user u 's satisfaction on item i as:

$$\hat{r}_{ui}^t := \hat{r}_{ui}^t = \frac{\hat{y}_{ui}}{1 + e_t(u, i)}. \quad (4)$$

Therefore, a large $e_t(u, i)$ will diminish user satisfaction \hat{r}_{ui}^t even with unchanged intrinsic interest y_{ui} . In the training stage of causal user model ϕ_M , we minimize the objective function in common recommendation models. In experiments, we use the MSE loss for the VirtualTaobao and BPR loss for KuaiEnv:

$$L_{\text{MSE}} = \sum_{(u, i, t) \in \mathcal{D}} (\hat{r}_{ui}^t - r_{ui}^t)^2, \quad L_{\text{BPR}} = - \sum_{(u, i, t) \in \mathcal{D}, j \sim p_n} \log\left(\sigma\left(\hat{r}_{ui}^t - \hat{r}_{uj}^t\right)\right). \quad (5)$$

Where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the Sigmoid function. The item j is a negative instance sampled from the distribution p_n .

Counterfactual Satisfaction Estimation. In the RL planning stage, when the learned causal user model ϕ_M interacts with the policy π_{θ} , the overexposure effect e_t^* now is different to e_t in the pre-learning stage. Therefore, we perform the causal intervention $do(E_t = e_t^*)$ [43] by cutting off the path $I \rightarrow E_t \rightarrow R$ as shown in Fig. 5(c). Unlike traditional causal methods aiming to remove the effect of confounders [57, 58], we still need to model the correct overexposure effect e_t^* in this stage. Note that we use the asterisk to mark out all values in this intervention stage. We compute e_t^* as:

$$e_t^*(u, i) = \gamma^* \cdot \alpha_u \beta_i \sum_{(u, i_l^*, t_l^*) \in S_u^*, t_l^* < t} \exp\left(-\frac{t - t_l^*}{\tau^*} \times \text{dist}(i, i_l^*)\right), \quad (6)$$

where S_u^* is the new interaction trajectory produced in the RL planning stage. γ^* is a hyper parameter introduced to adjust the scale of the overexposure effect. We fix γ to be 10 throughout experiments. τ^* is the temperature hyperparameter in the intervention stage and can have a different value with τ in Eq. (3). We estimate the *counterfactual satisfaction* as:

$$\hat{r}_{ui}^{t*} = \frac{\hat{y}_{ui}}{1 + e_t^*(u, i)}. \quad (7)$$

By now, we can use the estimated *counterfactual satisfaction* as the reward signal to update the RL policy by optimizing Eq. (2). The

whole process is illustrated in the pre-learning and RL planning stage in Fig. 4. We use the Adam optimizer [24] in learning the causal user model ϕ_M , the policy π_θ , and the state tracker.

At last, by innovatively enhancing the offline RL framework with causal inference, we obtain a policy that can guarantee large user satisfaction by preventing filter bubbles, i.e., overexposing items.

5 EXPERIMENTS

In this section, we conduct experiments to evaluate the IRS. We aim to investigate the following research questions: **(RQ1)** How does CIRS perform compared with SOTA static recommendation methods and RL-based interactive recommendation policies? **(RQ2)** How does CIRS perform in a limited number of interactive rounds? **(RQ3)** How does CIRS perform in different environments with varying user tolerance of filter bubble? **(RQ4)** What is the effect of the key parameters in CIRS?

5.1 Experimental Setup

We introduce the experimental settings with regards to the environments, evaluation metrics, and state-of-the-art methods.

5.1.1 Recommendation Environments. Traditional recommendation datasets are too sparse to evaluate the interactive recommender systems. We use two recommendation environments, VirtualTaobao² [50] and KuaiEnv. The two environments can play the same role as the online real users. For the recommenders, an environment is like a black box as shown in the upper right corner of Fig. 4.

VirtualTaobao is a benchmark RL environment for recommendation. It is created by simulating the behaviors of real users on Taobao, one of the largest online retail platforms. In VirtualTaobao, a user is represented as an 88-dimensional vector $\mathbf{e}_u \in \{0, 1\}^{88}$, and a recommendation is represented as a 27-dimensional vector $\mathbf{e}_i \in \mathbb{R}^{27}, 0 \leq \mathbf{e}_i \leq 1$. When a model makes a recommendation \mathbf{e}_i , the environment will immediately return a reward signal representing user interest, i.e., a scalar $r \in \{0, 1, \dots, 10\}$.

It provides 100,000 logged interactions for training the offline RL policy. In VirtualTaobao, we use Euclidean distance for the $\text{dist}(i, i_j)$ term in Eq. (3) and Eq. (6).

KuaiEnv is created by us on the KuaiRec dataset³ [12]. KuaiRec is a real-world dataset that contains a fully observed user-item interaction matrix, which means each user has viewed each video and then left feedback. Therefore, unlike VirtualTaobao that simulates real users by training a model on Taobao data, KuaiEnv uses real user historical feedback. We define the reward signal as the video watching ratio which is the ratio of viewing time to the total video length. Without loss of generality, we use this floating point number to indicate users' intrinsic interest. We use the fully-observed matrix, i.e., *small matrix*, to evaluate the policy π_θ . For pre-learning the user model ϕ_M , we use the additional sparse user-video interactions in the *big matrix*. For the details of the data, please refer to the KuaiRec dataset [12].

Each video in KuaiRec has at least one and no more than four categorical attributes, e.g., Food or Sports. Hence we use the Hamming distance for computing the $\text{dist}(i, i_j)$ term in Eq. (3) and Eq. (6).

Exit Mechanism. By now, VirtualTaobao and KuaiEnv can provide users' intrinsic interest as the reward signal. However, they cannot reflect users' responses to the overexposure effect. To this end, we introduce the "feel bored then quit" mechanism in two environments to penalize filter bubbles. Concretely, in VirtualTaobao, we compute the Euclidean distance between recommended target and the most recent N recommended items. If any of them is lower than the threshold d_Q , the environment will quit the interaction process as the real users can feel bored and quit under such the monotonous recommendation. In KuaiEnv, similarly, for the most recent N recommended items, if there are more than n_Q items in the N items have at least one attribute of the current recommended target, then the environment ends the interaction process.

5.1.2 Evaluation Metrics. We aim to evaluate the model performance with regard to cumulative satisfaction over the whole interaction trajectory \mathcal{S} , i.e., $\sum_{l=0}^{|\mathcal{S}|} r_{t+l}$, where r_t is the reward signal returned by VirtualTaobao or KuaiEnv. Note that in this setting, user satisfaction is set as:

$$\text{satisfaction} = \begin{cases} \text{interest}, & \text{if no filter bubble ever occurs,} \\ 0, & \text{otherwise.} \end{cases}$$

I.e., if the recommendation does not trigger the exit mechanism, we can accumulate the rewards represent intrinsic interest. But whenever an overexposed item triggers the exit mechanism, the interaction is interrupted and no reward can be added anymore. Intuitively, to pursue long-term success, the recommender policy must find a trade-off between pursuing a higher single-round satisfaction and maintaining a long interaction sequence. We report the average cumulative satisfaction over 100 interaction sequences.

5.1.3 Baselines. We use the commonly used static recommendation models plus some straightforward policies as baselines. For KuaiEnv, the static recommendation modules in the user model are implemented as the powerful baseline **DeepFM** [16] and three debiasing methods: **IPS** [52], **PD** [69], and **DICE** [73]. We implement the debiasing modules of the three methods on the basic DeepFM model. Since these static methods do not have a strategy for varying the recommendation results, we add a Softmax layer on their predicting results and randomly sample items from the distribution. For comparing with the other policies, we add several effective policies include **Random**, **ϵ -greedy**, **UCB** [2] on the results of DeepFM. For VirtualTaobao, since the users and items are given by feature vectors, we can only implement a multilayer perceptron (**MLP**) with sampling on the Softmax results and the ϵ -greedy strategy. Besides, we implement the powerful RL baseline, **PPO** [49], on the same offline RL setting, i.e., it is learned by interacting with the user model but without the causal inference module. For comparison, we denoted this baseline as **CIRS w/o CI**.

5.2 Overall Performance Comparison

We evaluate the proposed CIRS and baselines in two environments. For general comparison, we do not limit the length of the interaction and set the max round to be large enough (but feasible to implement). We set the max round to be 50 and 100 for VirtualTaobao and KuaiEnv, respectively. For the parameters in the environment setting, we set the window size, i.e., the number of the most recent

²<https://github.com/eyounx/VirtualTaobao>

³<https://chongminggao.github.io/KuaiRec/>

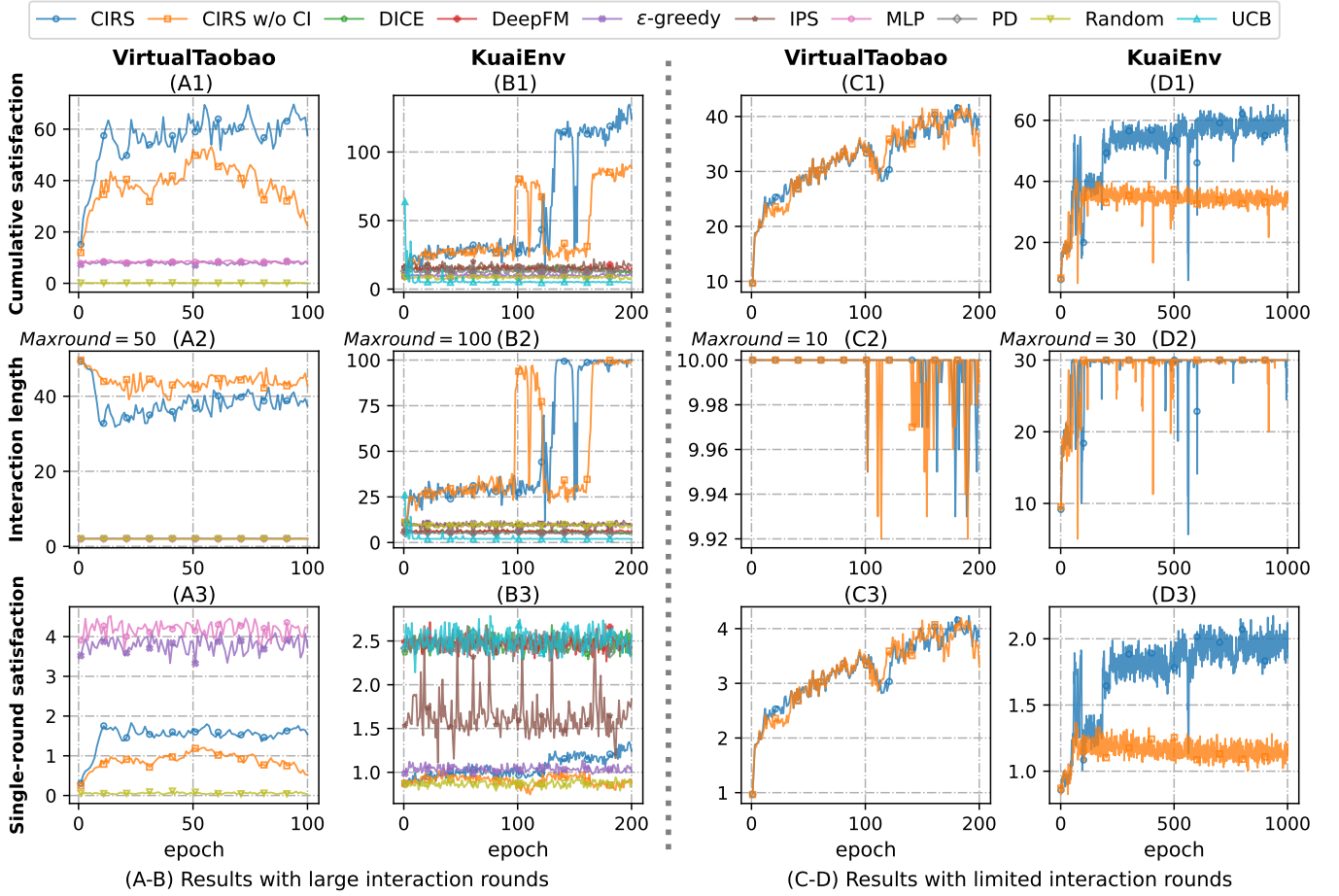


Figure 6: Results of all methods with large interaction rounds (Section 5.2) and limited interaction rounds (Section 5.3).

recommendations, to be $N = 5$ and exit threshold to be $d_Q = 3.0$ for VirtualTaobao and $N = 1$, $n_Q = 1$ for KuaiEnv.

The results are shown in Fig. 6 (A-B). From (A1) and (B1) we can see the proposed CIRS achieves the maximal average cumulative satisfaction after certain epochs' learning. In the first few epochs in VirtualTaobao, the performances of both CIRS and CIRS w/o CI improve because the RL policy gradually finds the right user preference so the satisfaction in each round increases (A3). Interestingly, the increasing of the single-round satisfaction compromises the length of trajectory in the beginning (A2). And the policy of CIRS eventually finds a balance point between length and single-round satisfaction, thus achieving the maximal cumulative satisfaction. However, without the causal inference module, the policy becomes unstable and the performance degenerates with epoch increasing.

In KuaiEnv, CIRS also achieves the largest cumulative satisfaction (B1) after enough epochs. For both VirtualTaobao and KuaiEnv, CIRS beats the counterpart method CIRS w/o CI, which demonstrates the effectiveness of the causal module in CIRS.

For other baselines, we can see that all other methods except Random can achieve better single-round performance (A3 and B3). However, their recommendation results are too limited and narrow even with the randomness introduced by the basic policies (i.e., Random sampling, Softmax-based sampling, and ϵ -greedy). Note that

in VirtualTaobao, even the random sampling cannot bring longer interaction sequence because of the curse of dimensionality: The action space has 88 dimensions so the Euclidean distance of any two random points becomes indiscriminate. IPS has the maximal high variance in terms of the single-round performance (B3), which has been widely discussed [52]. UCB is a policy that can automatically balance exploration and exploitation, it has the best performance in the beginning. However, after several epochs of exploration, the policy enhances its belief on certain items and thus leads to the filter bubble. Therefore, UCB ends up with the lowest interaction length.

To conclude, except the deep RL policy-based methods (i.e., CIRS and CIRS w/o CI), static recommendation models with heuristic policies (i.e., Softmax-based sampling, and ϵ -greedy, and UCB) cannot overcome the overexposure effect, thus results in filter bubbles and low user satisfaction. Furthermore, by comparing CIRS with CIRS w/o CI, we show the effectiveness of causal inference in the offline RL framework.

5.3 Results with Limited Interaction Rounds

In real-world recommendation scenarios, users have limited energy and will not spend too many rounds for interacting with the recommender. Therefore, we limit the max round to be 10 and 30

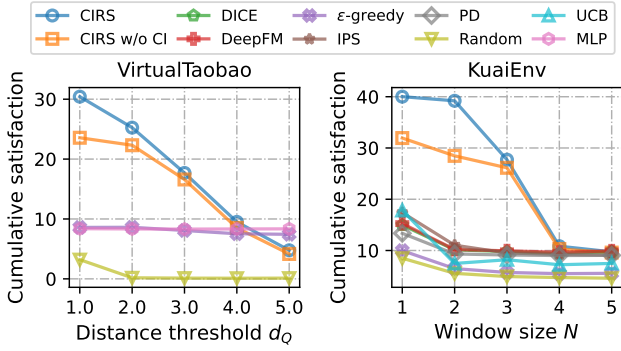


Figure 7: Results under different user sensitivity

in VirtualTaobao and KuaiEnv, respectively. We aim to investigate whether the policy can exploit to improve the single-round satisfaction under such a situation. We alter the exit threshold $d_Q = 1.0$ in VirtualTaobao for better demonstration.

From the results in Fig. 6 (C-D), we can see that CIRS outperforms CIRS w/o CI in KuaiEnv, and tie it in VirtualTaobao. In VirtualTaobao, both of the two policies achieve the same level of single-round performance (greater than 4.0) that is similar as the static methods (C3 and A3). In KuaiEnv, both the two policies attain higher single-round performances (D3) compared with that in the former setting (B3). Specially, CIRS has a great improvement on single-round performance (D3 and B3), which means that it is suitable for real-world interactive recommendation scenarios with limited interaction rounds. Actually, in (B1 and B3), we can also see that the performance of CIRS continues increasing at $epoch = 200$. This means CIRS has potential even under huge rounds, given enough training time. Again, in this setting, we conclude that by integrating with causal inference, we let CIRS outperform its counterpart greatly.

5.4 Results with Different User Sensitivity

To validate the generality of CRS, we vary the parameters d_Q and N in VirtualTaobao and KuaiEnv, respectively. A large d_Q or N means that users get more sensitive to filter bubbles and become easier to quit the interaction. The results in Fig. 7 show that CIRS outperforms all baselines when users are less sensitive. However, the performance inevitably decreases when users become more sensitive. At $d_Q \geq 4$ or $N \geq 4$, CIRS and CIRS w/o CI can only achieve the same performance with other baselines. This means, facing extremely picky users, even the model enhanced by causal inference cannot alleviate the dissatisfaction caused by filter bubbles.

Meanwhile, other baselines have similar performance under different user sensitivity — the recommendations make users feel bored and quit even through the user is more tolerant to filter bubble (i.e., less sensitive to item overexposure). This also demonstrates they are not suitable for addressing the filter bubble issue.

5.5 Effect of Key Parameters

We investigate the effect of the key parameters of CIRS in KuaiEnv. We compare the learned α_u and β_i in the user model with two statistics of data, i.e., the activity of the users and the popularity of

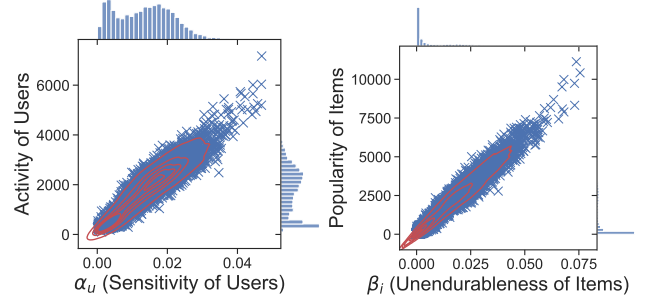


Figure 8: Relationship between the learned α_u, β_i and the data statistics (user activity and item popularity).

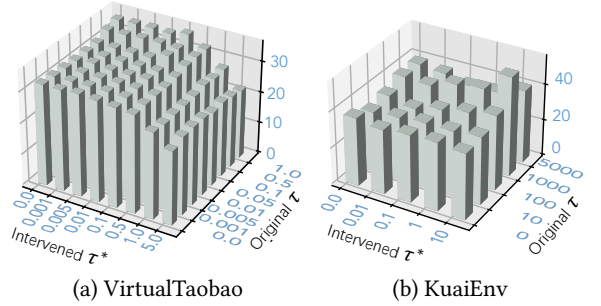


Figure 9: Cumulative satisfaction with different τ - τ^* pairs.

items, which are derived by summing the rows and columns of the *big matrix*. Fig. 8 shows the reasonable results: user sensitivity, i.e., α_u , is proportional to the user activity, i.e., an active user is easier to get bored when viewing overexposed videos. In addition, item unendurableness, i.e., β_i , is proportional to the item popularity, i.e., popular videos are less endurable when they are overexposed.

Furthermore, we investigate the effect of different combinations of τ (in Eq. (3)) and τ^* (in Eq. (6)) on the cumulative satisfactions. CIRS with $\tau = 0, \tau^* = 0$ degenerates to CIRS w/o CI since both $e_t(u, i)$ and $e_t^*(u, i)$ become 0, i.e., the modeling of user satisfaction will not take into account the overexposure effect. The results in Fig. 9 demonstrate that suitable τ - τ^* pairs indeed improve the performance compared to CIRS w/o. Note that the orders of magnitude of τ and τ^* differ greatly in KuaiEnv, because the unit of time in Eq. (3) and Eq. (6) are different. The former uses second(s) in log data and the latter uses step(s) in the RL planning and evaluation stages.

6 CONCLUSION AND FUTURE WORK

This work studies filter bubbles in interactive recommendation, where overexposure effect can be easily tracked and estimated. Accordingly, we propose a counterfactual interactive recommender system (CIRS), with leveraging causal inference in offline RL to deduce users' varying satisfaction. To conduct evaluations, we innovatively create a faithful RL environment, KuaiEnv, based on a real-world fully observed user rating dataset. Extensive experiments demonstrate that the proposed method can burst filter bubbles and increase users' cumulative satisfaction.

In the future, it is interesting to explore other types of biases in interactive recommendation. This work illustrates filter bubble would evolve during the process of user-system interaction. We believe other biases may also have this nature. It will be valuable to transfer the experience of this work to tackle other biases in a dynamic environment. The proposed causality-enhanced IRS framework could be adapted for other biases.

REFERENCES

- [1] Guy Aridor, Duarte Goncalves, and Shan Sikdar. 2020. Deconstructing the Filter Bubble: User Decision-Making and Recommender Systems. In *RecSys '20*. 82–91.
- [2] Peter Auer. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3 (2002), 397–422.
- [3] Xueying Bai, Jian Guan, and Hongning Wang. 2019. A Model-Based Reinforcement Learning with Adversarial Training for Online Recommendation. In *NeurIPS '19*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.).
- [4] Stephen Bonner and Flavian Vasile. 2018. Causal Embeddings for Recommendation. In *RecSys '18*. 104–112.
- [5] 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).
- [6] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H. Chi. 2019. Top-K Off-Policy Correction for a REINFORCE Recommender System. In *WSDM '19*. 456–464.
- [7] Minmin Chen, Bo Chang, Can Xu, and Ed H. Chi. 2021. User Response Models to Improve a REINFORCE Recommender System. In *WSDM '21*. 121–129.
- [8] Xinshi Chen, Shuang Li, Hui Li, Shaohua Jiang, Yuan Qi, and Le Song. 2019. Generative adversarial user model for reinforcement learning based recommendation system. In *ICML '19*. 1052–1061.
- [9] Tim Donkers and Jürgen Ziegler. 2021. The Dual Echo Chamber: Modeling Social Media Polarization for Interventional Recommending. In *RecSys '21*. 12–22.
- [10] Amir Feder, Katherine A Keith, Emaad Manzoor, Reid Pryzant, Dhanya Sridhar, Zach Wood-Doughty, Jacob Eisenstein, Justin Grimm, Roi Reichart, Margaret E Roberts, et al. 2021. Causal Inference in Natural Language Processing: Estimation, Prediction, Interpretation and Beyond. *arXiv preprint arXiv:2109.00725* (2021).
- [11] Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and Challenges in Conversational Recommender Systems: A Survey. *AI Open* 2 (2021), 100–126.
- [12] Chongming Gao, Shijun Li, Wenqiang Lei, Biao Li, Peng Jiang, Jiawei Chen, Xiangnan He, Jiaxin Mao, and Tat-Seng Chua. 2022. KuaiRec: A Fully-observed Dataset for Recommender Systems. *arXiv preprint arXiv:2202.10842* (2022).
- [13] Chongming Gao, Shuai Yuan, Zhong Zhang, Hongzhi Yin, and Junming Shao. 2019. BLOMA: Explain Collaborative Filtering via Boosted Local Rank-One Matrix Approximation. In *DASFAA '19*. Springer, 487–490.
- [14] Mingkun Gao, Hyo Jin Do, and Wai-Tat Fu. 2018. Burst Your Bubble! An Intelligent System for Improving Awareness of Diverse Social Opinions. In *IUI '18*. 371–383.
- [15] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline A/B Testing for Recommender Systems. In *WSDM '18*. 198–206.
- [16] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine Based Neural Network for CTR Prediction. In *IJCAI'17*. 1725–1731.
- [17] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR '20*. 639–648.
- [18] MA Hernán and JM Robins. 2020. *Causal Inference: What If*. Boca Raton: Chapman & Hall/CRC.
- [19] Jin Huang, Harrie Oosterhuis, Maarten de Rijke, and Herke van Hoof. 2020. Keeping Dataset Biases out of the Simulation: A Debaised Simulator for Reinforcement Learning Based Recommender Systems. In *RecSys '20*. 190–199.
- [20] Eslam Hussein, Prerna Juneja, and Tanushree Mitra. 2020. Measuring Misinformation in Video Search Platforms: An Audit Study on YouTube. *Proc. ACM Hum.-Comput. Interact.* 4, CSCW1, Article 048 (May 2020).
- [21] Eugene Ie, Vihan Jain, Jing Wang, Sanmit Narvekar, Ritesh Agarwal, Rui Wu, Heng-Tze Cheng, Morgane Lustman, Vince Gatto, Paul Covington, et al. 2019. Reinforcement learning for slate-based recommender systems: A tractable decomposition and practical methodology. *arXiv preprint arXiv:1905.12767* (2019).
- [22] Rolf Jagerman, Ilya Markov, and Maarten de Rijke. 2019. When People Change Their Mind: Off-Policy Evaluation in Non-Stationary Recommendation Environments. In *WSDM '19*. 447–455.
- [23] Olivier Jeunen and Bart Goethals. 2021. Pessimistic Reward Models for Off-Policy Learning in Recommendation. In *RecSys '21*. 63–74.
- [24] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980* (2014).
- [25] Haruka Kiyohara, Kosuke Kawakami, and Yuta Saito. 2021. Accelerating Offline Reinforcement Learning Application in Real-Time Bidding and Recommendation: Potential Use of Simulation. *arXiv preprint arXiv:2109.08331* (2021).
- [26] Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic Algorithms. In *NeurIPS '00*. 1008–1014.
- [27] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *KDD '08*. 426–434.
- [28] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. *arXiv preprint arXiv:2005.01643* (2020).
- [29] Qian Li, Xiangmeng Wang, and Guandong Xu. 2021. Be Causal: De-biasing Social Network Confounding in Recommendation. *arXiv preprint arXiv:2105.07775* (2021).
- [30] Dawen Liang, Laurent Charlin, James McInerney, and David M. Blei. 2016. Modeling User Exposure in Recommendation. In *WWW '16*. 951–961.
- [31] 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 *SIGIR '20*. 831–840.
- [32] Feng Liu, Huifeng Guo, Xutao Li, Ruiming Tang, Yunming Ye, and Xiuqiang He. 2020. End-to-End Deep Reinforcement Learning Based Recommendation with Supervised Embedding. In *WSDM '20*. 384–392.
- [33] Feng Liu, Ruiming Tang, Xutao Li, Weinan Zhang, Yunming Ye, Haokun Chen, Huifeng Guo, and Yuzhou Zhang. 2018. Deep Reinforcement Learning based Recommendation with Explicit User-item Interactions Modeling. *arXiv preprint arXiv:1810.12027* (2018).
- [34] Ping Liu, Karthik Shivaram, Aron Culotta, Matthew A. Shapiro, and Mustafa Bilgic. 2021. The Interaction between Political Typology and Filter Bubbles in News Recommendation Algorithms. In *WWW '21*. 3791–3801.
- [35] Yong Liu, Yingtai Xiao, Qiong Wu, Chunyan Miao, Juyong Zhang, Binqiang Zhao, and Haihong Tang. 2020. Diversified Interactive Recommendation with Implicit Feedback. In *AAAI '20*. 4932–4939.
- [36] David Lopez-Paz, Robert Nishihara, Soumith Chintala, Bernhard Scholkopf, and Léon Bottou. 2017. Discovering causal signals in images. In *CVPR '17*. 6979–6987.
- [37] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Ji Yang, Minmin Chen, Jiaxi Tang, Lichan Hong, and Ed H. Chi. 2020. Off-Policy Learning in Two-Stage Recommender Systems. In *WWW '20*. 463–473.
- [38] Farzan Masrour, Tyler Wilson, Heng Yan, Pang-Ning Tan, and Abdol Esfahanian. 2020. Bursting the Filter Bubble: Fairness-aware Network Link Prediction. In *AAAI '20*. 841–848.
- [39] James McInerney, Brian Brost, Praveen Chandar, Rishabh Mehrotra, and Benjamin Carterette. 2020. Counterfactual Evaluation of Slate Recommendations with Sequential Reward Interactions. In *KDD '20*. 1779–1788.
- [40] Tien T. Nguyen, Pik-Mai Hui, F. Maxwell Harper, Loren Terveen, and Joseph A. Konstan. 2014. Exploring the Filter Bubble: The Effect of Using Recommender Systems on Content Diversity. In *WWW '14*. 677–686.
- [41] Zachary A. Pardos and Weijie Jiang. 2020. Designing for Serendipity in a University Course Recommendation System. In *LAK '20*. 350–359.
- [42] Eli Pariser. 2011. *The filter bubble: How the new personalized web is changing what we read and how we think*. Penguin.
- [43] Judea Pearl. 2009. *Causality*. Cambridge University Press.
- [44] Manoel Horta Ribeiro, Raphael Ottoni, Robert West, Virgilio A. F. Almeida, and Wagner Meira. 2020. Auditing Radicalization Pathways on YouTube. In *FAT* '20*. 131–141.
- [45] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased Recommender Learning from Missing-Not-At-Random Implicit Feedback. In *WSDM '20*. 501–509.
- [46] Tobias Schnabel, Paul N. Bennett, Susan T. Dumais, and Thorsten Joachims. 2018. Short-Term Satisfaction and Long-Term Coverage: Understanding How Users Tolerate Algorithmic Exploration. In *WSDM '18*. 513–521.
- [47] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as Treatments: Debiasing Learning and Evaluation. In *ICML '16*. 1670–1679.
- [48] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional Continuous Control Using Generalized Advantage Estimation. *arXiv preprint arXiv:1506.02438* (2015).
- [49] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347* (2017).
- [50] Jing-Cheng Shi, Yang Yu, Qing Da, Shi-Yong Chen, and An-Xiang Zeng. 2019. Virtual-Taobao: Virtualizing Real-world Online Retail Environment for Reinforcement Learning. In *AAAI '19*. 4902–4909.
- [51] Larissa Spinelli and Mark Crovella. 2020. How YouTube Leads Privacy-Seeking Users Away from Reliable Information. In *UMAP '20 Adjunct*. 244–251.
- [52] Adith Swaminathan and Thorsten Joachims. 2015. Counterfactual Risk Minimization: Learning from Logged Bandit Feedback. In *ICML '15*. 814–823.

- [53] Matus Tomlein, Branislav Pecher, Jakub Simko, Ivan Srba, Robert Moro, Elena Stefancova, Michal Kompan, Andrea Hrkova, Juraj Podrouzek, and Maria Bielikova. 2021. An Audit of Misinformation Filter Bubbles on YouTube: Bubble Bursting and Recent Behavior Changes. In *RecSys '21*. 1–11.
- [54] Antonela Tommasel, Juan Manuel Rodriguez, and Daniela Godoy. 2021. I Want to Break Free! Recommending Friends from Outside the Echo Chamber. In *RecSys '21*. 23–33.
- [55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NeurIPS '17*.
- [56] Shiqi Wang, Chongming Gao, Min Gao, Junliang Yu, Zongwei Wang, and Hongzhi Yin. 2022. Who Are the Best Adopters? User Selection Model for Free Trial Item Promotion. *arXiv preprint arXiv:2202.09508* (2022).
- [57] Tan Wang, Jianqiang Huang, Hanwang Zhang, and Qianru Sun. 2020. Visual commonsense r-cnn. In *CVPR '20*. 10760–10770.
- [58] Wenjie Wang, Fuli Feng, Xiangnan He, Xiang Wang, and Tat-Seng Chua. 2021. Deconfounded Recommendation for Alleviating Bias Amplification. In *KDD '21*. 1717–1725.
- [59] Wenlin Wang, Hongteng Xu, Ruiyi Zhang, Wenqi Wang, Piyush Rai, and Lawrence Carin. 2021. Learning to recommend from sparse data via generative user feedback. In *AAAI '21*.
- [60] Zhenlei Wang, Jingsen Zhang, Hongteng Xu, Xu Chen, Yongfeng Zhang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Counterfactual Data-Augmented Sequential Recommendation. In *SIGIR '21*. 347–356.
- [61] Teng Xiao and Donglin Wang. 2021. A General Offline Reinforcement Learning Framework for Interactive Recommendation. In *AAAI '21*.
- [62] Xin Xin, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M. Jose. 2020. Self-Supervised Reinforcement Learning for Recommender Systems. In *SIGIR '20*. 931–940.
- [63] Shuyuan Xu, Yingqiang Ge, Yunqi Li, Zuohui Fu, Xu Chen, and Yongfeng Zhang. 2021. Causal Collaborative Filtering. *arXiv preprint arXiv:2102.01868* (2021).
- [64] Ya Xu, Nanyu Chen, Addrian Fernandez, Omar Sinno, and Anmol Bhasin. 2015. From Infrastructure to Culture: A/B Testing Challenges in Large Scale Social Networks. In *KDD '15*. 2227–2236.
- [65] Yuanbo Xu, Yongjian Yang, En Wang, Jiayu Han, Fuzhen Zhuang, Zhiwen Yu, and Hui Xiong. 2020. Neural Serendipity Recommendation: Exploring the Balance between Accuracy and Novelty with Sparse Explicit Feedback. *ACM Trans. Knowl. Discov. Data* 14, 4, Article 50 (June 2020).
- [66] Mengyue Yang, Quanyu Dai, Zhenhua Dong, Xu Chen, Xiuqiang He, and Jun Wang. 2021. Top-N Recommendation with Counterfactual User Preference Simulation. In *CIKM '21*.
- [67] Junliang Yu, Min Gao, Hongzhi Yin, Jundong Li, Chongming Gao, and Qinyong Wang. 2019. Generating Reliable Friends via Adversarial Training to Improve Social Recommendation. In *ICDM '19*. IEEE, 768–777.
- [68] Ruiyi Zhang, Tong Yu, Yilin Shen, Hongxia Jin, Changyou Chen, and Lawrence Carin. 2019. Reward Constrained Interactive Recommendation with Natural Language Feedback. In *NeurIPS '19*.
- [69] 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 *SIGIR '21*. 11–20.
- [70] Zhong Zhang, Nian Shao, Chongming Gao, Rui Miao, Qinli Yang, and Junming Shao. 2022. Mixhead: Breaking the Low-rank Bottleneck in Multi-head Attention Language Models. *Knowledge-Based Systems* (2022), 108075.
- [71] Xiangyu Zhao, Long Xia, Lixin Zou, Hui Liu, Dawei Yin, and Jiliang Tang. 2020. Whole-Chain Recommendations. In *CIKM '20*. 1883–1891.
- [72] Xiangyu Zhao, Long Xia, Lixin Zou, Hui Liu, Dawei Yin, and Jiliang Tang. 2021. UserSim: User Simulation via Supervised Generative Adversarial Network. In *WWW '21*. 3582–3589.
- [73] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling User Interest and Conformity for Recommendation with Causal Embedding. In *WWW '21*. 2980–2991.
- [74] Sijin Zhou, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. 2020. Interactive Recommender System via Knowledge Graph-Enhanced Reinforcement Learning. In *SIGIR '20*. 179–188.
- [75] Ziwei Zhu, Yun He, Xing Zhao, and James Caverlee. 2021. Popularity Bias in Dynamic Recommendation. In *KDD '21*. 2439–2449.
- [76] Lixin Zou, Long Xia, Zhuoye Ding, Jiaying Song, Weidong Liu, and Dawei Yin. 2019. Reinforcement Learning to Optimize Long-Term User Engagement in Recommender Systems. In *KDD '19*. 2810–2818.
- [77] Lixin Zou, Long Xia, Pan Du, Zhuo Zhang, Ting Bai, Weidong Liu, Jian-Yun Nie, and Dawei Yin. 2020. Pseudo Dyna-Q: A Reinforcement Learning Framework for Interactive Recommendation. In *WSDM '20*. 816–824.
- [78] Lixin Zou, Long Xia, Yulong Gu, Xiangyu Zhao, Weidong Liu, Jimmy Xiangji Huang, and Dawei Yin. 2020. Neural Interactive Collaborative Filtering. In *SIGIR '20*. 749–758.