Disentangling User Interest and Popularity Bias for Recommendation with Causal Embedding

Yu Zheng[1], Chen Gao[1], Xiang Li[2], Xiangnan He[3], Yong Li[1], Depeng Jin[1]
[1]Tsinghua University, [2]University of Hong Kong, [3]University of Science and Technology of China
y-zheng19@mails.tsinghua.edu.cn, gc16@mails.tsinghua.edu.cn, xli2@cs.hku.hk,
xiangnanhe@gmail.com, liyong07@tsinghua.edu.cn, jindp@tsinghua.edu.cn

Abstract

Recommendation models are usually trained on observational data. However, observational data exhibits various bias, such as popularity bias. Biased data results in a gap between online environment and offline evaluation, which makes it difficult to measure the real effect of recommendation. To bridge this gap, most existing methods exploit causal inference to eliminate the bias in historical observational data, e.g., by re-weighting training samples or leveraging a small fraction of unbiased data. However, different causes of an interaction are bundled together as a unified representation in these algorithms. In this paper, we present DICE, a general framework that learns representations where user interest and item popularity are structurally disentangled. We leverage separate embeddings for different causes, and make each embedding capture only one cause by training with cause-specific samples and imposing direct disentanglement supervision. Our algorithm outperforms state-of-the-art baselines with remarkable improvements on two real-world datasets with various backbone models. We further demonstrate that the learned embeddings successfully capture the desired causes.

1 Introduction

Measuring the real effect of recommender systems with observational data is difficult, because it is usually biased towards certain items [8, 14]. For example, popular items are more likely to be recommended which to some extent hides a user' real interest [11]. From the perspective of causal inference, the most effective way to evaluate a recommendation algorithm is using completely randomized experimental design [35], or known as online A/B testing. However, A/B testing suffers from long turnaround time and high risk of degrading user experience [23]. Therefore, unbiased learning with biased offline data is crucial for recommendation.

In order to reduce the influence of popularity bias, causal approaches are proposed to recover real interest from biased observational data. In recommendation with implicit feedback where only binary click data is available, Inverse Propensity Scoring (IPS) [2, 9, 14, 15, 24, 46] is widely used. It imposes lower weights for clicks on popular items and boosts long-tail items. Bonner *et al.* [8] proposed another direction of causal recommendation, by using embeddings for both biased and unbiased Matrix Factorization (MF) [27]. However, with unified representations for users and items in these approaches, different semantics are not disentangled with each other. As a result, these methods lack robustness for the data of different domains, especially when some of the causes are varying while other causes are stable. On the other hand, in recommendation with explicit feedback such as discrete ratings, one disentanglement trial in causal recommendation is Deconvolve [39], which identifies each observed rating as union of real rating and recommender-influenced rating. With strong modeling assumptions, real ratings are recovered using SVD. However, assumptions

Preprint. Under review.

in Deconvolve only hold true in explicit feedback, and those matrix computations are not valid for binary data in implicit feedback, which is more prevalent in modern recommendation.

In implicit feedback, there are usually multiple causes, such as real interest and popular trend, for one single observation of click. However, existing recommendation algorithms often use unified representations for each user or item, thus they fail to disentangle different causes. Since different causes are bundled together as one single representation, these approaches tend to be misguided by popularity bias in observational data [11]. In this paper, we focus on unbiased learning from biased data. Specifically, we investigate the two main causes, interest and popularity. We aim to learn disentangled representations for the two causes, and estimate clicks by combining them together.

Disentangling interest and popularity for recommendation is challenging and largely unexplored. Specifically, we face three challenges. First, there are infinite solutions for interest and popularity when only click data is available. Therefore, reasonable assumptions are needed to make this problem solvable. Meanwhile, the disentanglement design needs to be general across various backbone models. Second, learning disentangled representations is intrinsically hard. Embeddings are directly taken inner products to make recommendations, while requiring disentanglement among embeddings often conflicts with user preference modeling. Meanwhile, there is no ground-truth for real interest, which makes it more difficult to capture. Third, recommendations are made by combining different causes, which requires careful designs on how to learn and incorporate different components.

In this paper, we present a general framework for **D**isentangling **I**nterest and popularity bias with **C**ausal **E**mbeddings (DICE). We state concise and reasonable causal assumptions that decompose each click record into user's real interest and item's popularity influence. Those assumptions are not based on specific models, but sourced from how data is generated, which makes our approach effective upon various backbones. Unlike existing embedding based methods that learn unified representations for different causes, our approach adopts separate embeddings for interest and popularity. In order to force each embedding to capture only one cause, we train the model with cause-specific data. In addition, direct disentanglement supervision is added to attain stronger independence between embeddings of different causes. We estimate click behaviors by combining interest and popularity, with the help of multi-task and curriculum learning. Experimental results show that DICE outperforms state-of-the-art baselines with over 5% improvements in terms of Recall and NDCG. Furthermore, analysis on the quality of learned embeddings illustrates that interest and popularity modeling is highly independent, which makes DICE an interpretable framework.

2 Proposed method: DICE

In this section, we present our approach for disentangling interest and popularity bias with causal embeddings. We first introduce the causal modeling basis of DICE, and then elaborate on the algorithm design of learning disentangled representations.

2.1 Causal modeling

A click record of a user on an item mainly reflects two aspects: (1) the item's characteristics match the user's interest, (2) the item's popularity matches the user's conformity. A click could come from one or both of the two aspects. Similar to Deconvolve [39], we make concise assumptions on each click record as union of real interest and popularity bias. Formally, we have the following assumption.

Assumption 2.1 *A click record in biased recommendation scenario results from two independent causes, which are the user's interest and the item's popularity.*

$$P_{\text{click}} = P_{\text{interest}} + P_{\text{popularity}}, \tag{1}$$

where P represents matching probability for a given user and item. This assumption is justified because users tend to have both particularity and conformity when interacting with recommender systems [30]. Figure 1(a) illustrates the causal graph of our concise assumptions.

Based on this assumption, we could further obtain quantitative relations on each cause. We first introduce several notations. We use M^I to denote the matrix of interest matching probability for all users and items, and M^P for popularity matching probability. If a user u clicks a popular item

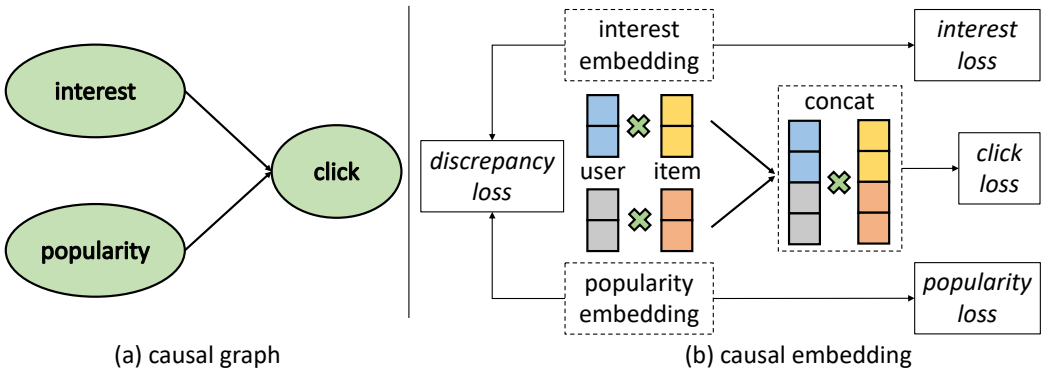

Figure 1: Causal graph and causal embeddings. (a) We make concise assumptions on each click that it results from two independent causes, user interest and item popularity. (b) We adopt separate embeddings for interest and popularity, thus each user or item has two embeddings. We force each embedding to capture only one cause by training different embeddings with cause-specific data and adding direct disentanglement supervision, under the framework of multi-task learning.

a, while does not click an unpopular item b, then we are not sure whether the user's interest on a is stronger than b, because popular items are more likely to be recommended to users. In other words, the click could come from the second cause (a is more popular than b). Meanwhile, we can also safely conclude that the total strength of the two causes of a is larger than b. Formally, we have two inequalities in this case:

$$\begin{aligned} &M_{ua}^P > M_{ub}^P, \\ &M_{ua}^I + M_{ua}^P > M_{ub}^I + M_{ub}^P. \end{aligned} \tag{2}$$

However, if a user clicks an unpopular item c, while does not click a popular item d, then we could attain more information. Since c is less likely to be recommended to the user, the click on c could be largely due to the user's interest. Formally, we have three inequalities in this case:

$$\begin{aligned} &M_{uc}^I > M_{ud}^I, \\ &M_{uc}^P < M_{ud}^P, \\ &M_{uc}^I + M_{uc}^P > M_{ud}^I + M_{ud}^P. \end{aligned} \tag{3}$$

Our concise assumptions are based on how data is generated and they are model-independent, which could benefit both upstream base models and downstream applications. By turning one equality to multiple inequalities, we make the task of disentangling interest and popularity solvable. We then introduce our solution using causal embeddings. Specifically, we adopt two sets of embeddings to separately capture interest and popularity. Each user has an interest embedding $u^{(\text{int})}$ and a popularity embedding $u^{(\text{pop})}$, and each item also has $i^{(\text{int})}$ and $i^{(\text{pop})}$ for the two causes. We use inner product to compute matching probability in both latent spaces. To estimate whether a user will click an item, we combine the two causes together. Therefore, the recommendation score for user x and item y is formulated as:

$$s_{xy} = \langle x^{(\text{int})}, y^{(\text{int})} \rangle + \langle x^{(\text{pop})}, y^{(\text{pop})} \rangle, \tag{4}$$

where $\langle x, y \rangle$ means inner product of two embeddings. Figure 1(b) demonstrates the disentanglement design of interest embeddings and popularity embeddings. The causal embedding framework could be regarded as an embedding version of Assumption 2.1.

2.2 Learning disentangled representations

Disentanglement between interest embedding and popularity embedding means that each embedding captures only one factor. To achieve such target, we use different data to train different embeddings. In other words, based on these aforementioned inequalities, we could obtain user-item interactions that mainly result from one specific cause, and leverage these interactions to optimize corresponding embeddings. Overall, we apply a multi-task learning framework on top of causal embeddings. The

main task is to estimate click behaviors combining the two sets of embeddings. Moreover, we add two extra tasks on separately learning interest embeddings and popularity embeddings using cause-specific data. To enhance disentanglement, we further add a discrepancy task in order to make the two sets of embeddings independent with each other.

We utilize BPR [36] to model the pairwise quantitative relations in (2) and (3). Each positive sample is paired with certain number of negative samples, and each training instance is a triplet (u, i, j) containing user ID, positive item ID and negative item ID. We use $\mathcal{O}$ to denote the whole training instances, which could be divided into two parts, $\mathcal{O}_1$ and $\mathcal{O}_2$. Specifically, $\mathcal{O}_1$ denotes those instances where negative samples are more popular than positive samples, and $\mathcal{O}_2$ denotes the opposite cases.

Estimating clicks This is the main target for recommender systems, and we combine the two causes to estimate clicks as introduced in (4). For each instance in training set $\mathcal{O}$, we use BPR to maximize the margin between scores of positive items and negative items. The loss function for click estimation is thus formulated as follows:

$$L_{\text{click}} = \sum_{(u,i,j)\in\mathcal{O}} \text{BPR}(\langle u^{(\text{int})} \| u^{(\text{pop})}, i^{(\text{int})} || i^{(\text{pop})} \rangle, \langle u^{(\text{int})} \| u^{(\text{pop})}, j^{(\text{int})} || j^{(\text{pop})} \rangle), \tag{5}$$

where $\|$ means concatenation of two embeddings. We use the concatenation form here for simplicity, which is equivalent to the summation form in (4). The BPR loss pushes the recommendation score for positive item i to be higher than negative item j.

Interest modeling As introduced in previous sections, for those training instances (i.e. $\mathcal{O}_1$) that negative items are more popular than positive items, the interaction is largely due to user's interest. These data is interest-specific, because we have inequalities for interest modeling. We also use BPR to optimize interest embeddings to learn such pairwise preference. The loss function only takes effect for instances in $\mathcal{O}_1$:

$$L_{\text{interest}} = \sum_{(u,i,j)\in\mathcal{O}_1} \text{BPR}(\langle u^{(\text{int})}, i^{(\text{int})} \rangle, \langle u^{(\text{int})}, j^{(\text{int})} \rangle). \tag{6}$$

Popularity modeling For instances in both $\mathcal{O}_1$ and $\mathcal{O}_2$, we have inequalities for popularity modeling. However, the direction of inequality is different in the two cases. We use these popularity-specific data to optimize popularity embeddings. Therefore, the loss function for popularity modeling is formulated as:

$$\begin{aligned}
L^{(1)}_{\text{popularity}} &= \sum_{(u,i,j)\in\mathcal{O}_1} -\text{BPR}(\langle u^{(\text{pop})}, i^{(\text{pop})} \rangle, \langle u^{(\text{pop})}, j^{(\text{pop})} \rangle), \\
L^{(2)}_{\text{popularity}} &= \sum_{(u,i,j)\in\mathcal{O}_2} \text{BPR}(\langle u^{(\text{pop})}, i^{(\text{pop})} \rangle, \langle u^{(\text{pop})}, j^{(\text{pop})}, \rangle), \\
L_{\text{popularity}} &= L^{(1)}_{\text{popularity}} + L^{(2)}_{\text{popularity}}.
\end{aligned} \tag{7}$$

Interest modeling and popularity modeling disentangle the two causes by training different embeddings with different cause-specific data. Meanwhile, the main task on estimating clicks also strengthens this disentanglement as a constraint. For example, in terms of a training instance (u, i, j) where negative item j is more popular than positive item i, interest modeling task forces the two sets of embeddings to learn that user u's interest in i is larger than j, and popularity modeling task forces them to learn that item i's popularity is less than j. Meanwhile, estimating clicks forces them to learn that the overall strength of i is larger than j. Therefore, what the model really learns is that the advantage of i over j with respect to interest dominates the disadvantage in popularity, which could be best learned by capturing only one cause with one embedding.

Discrepancy task Besides the three tasks above that disentangle interest and popularity by optimizing different embeddings with cause-specific data, we impose direct supervision to reinforce this disentanglement. Specifically, we add an extra discrepancy task on the distribution of the embeddings. Suppose $\mathbf{E}^{(\text{int})}$ and $\mathbf{E}^{(\text{pop})}$ represent two sets of embeddings of all users and items. We examine three candidate discrepancy loss functions, which are L1-inv, L2-inv and distance correlation (*dCor*).

L1-inv and L2-inv *maximize* L1 and L2 distances between $\mathbf{E}^{(\text{int})}$ and $\mathbf{E}^{(\text{pop})}$ respectively. *dCor* of two sets of embeddings could be calculated as follows:

$$dCor(\mathbf{E}^{(\text{int})}, \mathbf{E}^{(\text{pop})}) = \frac{dCov(\mathbf{E}^{(\text{int})}, \mathbf{E}^{(\text{pop})})}{\sqrt{dVar(\mathbf{E}^{(\text{int})}) \cdot dVar(\mathbf{E}^{(\text{pop})})}}, \tag{8}$$

where $dCov(\cdot)$ and $dVar(\cdot)$ represent distance covariance and distance variance, respectively. We refer to [42, 43] for more details on *dCor*. From high level, *dCor* is a more reasonable choice, since it focuses on the correlations of pairwise distances between interest embeddings and popularity embeddings. In other words, *dCor* makes the two sets of embeddings as independent with each other as possible. Thus the three options for discrepancy loss function are $-L1(\mathbf{E}^{(\text{int})}, \mathbf{E}^{(\text{pop})})$, $-L2(\mathbf{E}^{(\text{int})}, \mathbf{E}^{(\text{pop})})$ and $dCor(\mathbf{E}^{(\text{int})}, \mathbf{E}^{(\text{pop})})$. We will compare the three options in experiments.

2.3 Combining causes with curriculum learning

In the proposed framework, recommendation is made by combining the two causes. Therefore, we train causal embeddings with the four tasks simultaneously. We combine these loss functions together:

$$L = L_{\text{click}} + \alpha(L_{\text{interest}} + L_{\text{popularity}}) + \beta L_{\text{discrepancy}}. \tag{9}$$

Since estimating clicks is the main task for recommendation, α and β should be less than 1 from intuition. Meanwhile, discrepancy task directly influences the distribution of embeddings, thus too large β would negatively impact user preference modeling. Figure 1(b) illustrates the holistic design of the proposed framework.

As introduced previously, we could obtain two or three inequalities when the negative sample is less or more popular than the positive sample, respectively. Notice that those inequalities will hold true with high probability when the popularity gap is sufficiently large. Therefore, we adopt popularity based negative sampling with margin to guarantee those quantitative relations. Specifically, if the popularity of the positive sample is p, then we will sample negative instances from items with popularity larger than $p + m_{up}$, or lower than $p - m_{down}$, where m_{up} and m_{down} are positive margin values. By sampling negative items with popularity margin, we gain high confidence on our causal assumptions.

Inspired by curriculum learning [6], we adopt an easy-to-hard strategy on training DICE by adding decay on margin values and loss weights. Specifically, when margin values m_{up} and m_{down} are very large, we have high confidence on those inequalities for interest and popularity modeling, which means the tasks are *easier* and we set relatively high loss weights for L_{interest} and $L_{\text{popularity}}$. As we train the model, we increase the difficulty by decaying margin values, as well as loss weights, by a factor of 0.9 after each epoch. With curriculum learning, the proposed approach learns stronger disentanglement for high-confidence samples.

3 Experiments

Datasets We conduct experiments on two million-scale datasets collected from real-world applications, Movielens-10M dataset [17] and Netflix Prize dataset [7]. We also use Yahoo! Front Page dataset [28] to explore how different algorithms perform under fully unbiased situations. In order to measure the performance of causal learning, unbiased test sets are needed, and thus all datasets are transformed following the standard protocol introduced in related literatures [8, 29]. We binarize the datasets by keeping ratings of five stars as one, and others as zero. To simulate unbiased recommendation, we randomly sample half of the records with equal probability in terms of items, and leave the other half as biased training data. In other words, items are sampled with probability as **inverse** popularity, which means popular items are less selected. Finally, we obtain a 70/10/20 split for training set (50% biased and 20% unbiased), validation set (10% unbiased) and test set (20% unbiased). We refer to [8, 29] for details on extracting an unbiased test set from biased data.

Baselines We compare our approach with IPS [15] and CausE [8], the two state-of-the-art approaches for causal recommendation. In terms of IPS, we include plain IPS [37, 24], and its two variants, IPS-Cap [9] and IPS-Cap-Norm [15]. Specifically, plain IPS suffers from high variance, which makes it sensitive to extreme data. IPS-Cap adds max-capping to limit the largest IPS value

Table 1: Recommendation results on Movielens-10M and Netflix datasets.

	Movielens-10M			Netflix		
Name	Recall	Hit Ratio	NDCG	Recall	Hit Ratio	NDCG
MF	0.1605	0.5329	0.0973	0.1420	0.6433	0.1107
MF + IPS	0.1540	0.4966	0.0767	0.1200	0.5361	0.0732
MF + IPS-Cap	0.1588	0.5069	0.0787	0.1249	0.5482	0.0750
MF + IPS-Cap-Norm	**0.1846**	**0.5693**	0.0995	0.1426	0.6236	0.1003
MF + CausE	0.1584	0.5351	**0.0997**	**0.1428**	**0.6478**	**0.1139**
MF + DICE (ours)	**0.1936**	**0.6078**	**0.1158**	**0.1483**	**0.6600**	**0.1147**
GCN	0.1906	0.6024	0.1218	**0.1527**	**0.6689**	**0.1238**
GCN + IPS	0.1673	0.5464	0.0967	0.1269	0.5851	0.0874
GCN + IPS-Cap	0.1758	0.5648	0.1025	0.1316	0.5975	0.0917
GCN + IPS-Cap-Norm	**0.1942**	**0.6044**	**0.1223**	0.1342	0.6220	0.1086
GCN + CausE	0.1304	0.4730	0.0840	0.0905	0.5134	0.0776
GCN + DICE (ours)	**0.2121**	**0.6409**	**0.1365**	**0.1652**	**0.6973**	**0.1310**

which significantly reduces the variance. And IPS-Cap-Norm further adds normalization on IPS values to balance between bias and variance.

Backbones Causal approaches usually serve as additional methods upon backbone models. We use the most adopted backbone, Matrix Factorization (MF) [27] to compare different approaches. Meanwhile, we also incorporate the state-of-the-art collaborate filtering model, Graph Convolutional Networks (GCN) [16, 47, 18], to investigate whether algorithms generalize across different backbones.

Hyper-parameters For MF, GCN and IPS, we fix the embedding size as 128. While for CausE and DICE, the embedding size is fixed as 64, since they contain two sets of embeddings. Therefore, the number of parameters are the same for all methods to guarantee fair comparison. We set α as 0.1 and β as 0.01, which shows great performance and agnostic to both datasets and backbone models in experiments. We use BPR [36] as the loss function for all baselines. We use Adam [25] for optimization. Other hyper-parameters for our method and baselines are tuned by grid search.

We evaluate top-k recommendation performance for implicit feedback [36], which is the most common setting for recommendation. We use three frequently used metrics, which are Recall, Hit Ratio and NDCG. We first compare the overall performance of DICE and other baselines. After that we investigate how different algorithms perform on fully biased or unbiased set-ups which is closer to applications. Furthermore, we study whether each set of embeddings in DICE successfully learn the only desired cause. At last, we compare the three candidate loss functions for discrepancy task.

3.1 Recommendation performance

We use BPR-MF [36] and LightGCN [18], which are both state-of-the-art backbone models. Results on two datasets are listed in Table 1. We observe that our proposed DICE framework outperforms baselines with significant improvements on not only both datasets, but also both backbones. For example, DICE makes over 15% improvements with respect to NDCG using MF as backbone on Moveilens-10M dataset, and over 8% improvements with respect to Recall using GCN as backbone on Netflix dataset. Firstly, the disentanglement design of interest embeddings and popularity embeddings successfully distinguish the two causes of user interactions. It allows the framework to capture invariant interest from biased training data, and adapt to varying popularity bias in test cases. Secondly, the concise causal assumptions are sourced from how the data is generated, thus the framework is independent with backbone models. Results based on MF and GCN illustrate that DICE is a general framework, which could be smoothly integrated into various embedding based recommendation algorithms. Thirdly, the disentanglement of different causes makes our framework highly interpretable, which is significant for recommender systems. The improvements verify the effectiveness of disentangling interest and popularity bias with causal embeddings.

Table 2: Results on biased dataset.

Name	Recall	Hit Ratio	NDCG
MF	0.1483	0.5151	0.0985
IPS-Cap-Norm	0.1669	0.5524	0.1105
DICE	**0.1764**	**0.5731**	**0.1172**

Table 3: Results on unbiased dataset.

Name	Recall	Hit Ratio	NDCG
MF	0.0662	0.3631	0.0545
CausE	0.0671	0.3614	0.0543
DICE	0.0679	0.3618	0.0529

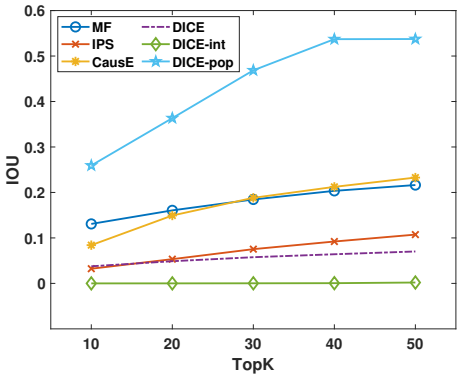

Figure 2: Overalp with ItemPop.

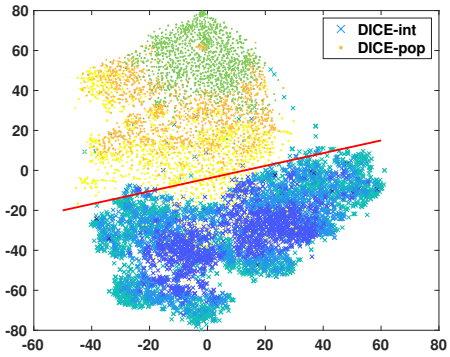

Figure 3: t-SNE of item embeddings.

3.2 Causal learning on biased and unbiased data

In previous experiments, all the algorithms are trained with a large fraction of biased data (50%) and a small fraction of unbiased data (20%). Adding extra unbiased data is not only a hard requirement of certain baseline method (CausE), but also reduces the difficulty of causal learning. However, unbiased data is often too expensive to obtain in real-world recommender systems. Therefore, in this section, we investigate how different algorithms perform when training with only biased data, which is more difficult but closer to applications. We remove the 20% unbiased training data in Movielens-10M dataset, and compare our proposed method with IPS-Cap-Norm, which is the most competitive baseline. CausE is omitted since it requires unbiased training data. Table 2 shows the results training with only biased data. Although performance of all methods drops drastically compared with easier situation in Table 1, the proposed DICE framework could still disentangle interest and popularity with only biased data, and outperforms other baselines significantly.

As introduced in [8], unbiased interest could be directly learned under fully random recommendation policy. We also conduct experiments on a random recommendation dataset, Yahoo! Front Page dataset [28]. This dataset contains 15 days of user click log for news articles on Yahoo!'s front page. The articles were chosen **uniformly at random**, which serves as a totally unbiased evaluation environment. Due to its random nature, a simple MF algorithm could successfully learn users' unbiased interest, and there is no need for introducing more advanced causal techniques, such as CausE or DICE. Table 3 demonstrates the results on this random recommendation dataset. In correspondence to our analysis, the performance of different methods are almost the same with each other. Close performance of different methods confirms that completely random experimental design is the easiest approach for unbiased learning. However, random recommendation is at the risk of bad user experience, thus causal learning from biased data is always important for recommendation.

3.3 Embedding disentanglement of real interest and popularity bias

In this section, we investigate the quality of embedding disentanglement in DICE. As there is ground-truth for popularity, we first study whether popularity embeddings capture the desired cause. Here we introduce another two versions of the framework, DICE-int and DICE-pop. They only use interest or popularity embeddings for recommendation, respectively. Note that in DICE we concatenate the two embeddings. We compare the overlapped recommended items of all methods with ItemPop, which recommends the top popular items. Intersection Over Union (IOU) is used as the metric. Figure 2 illustrates the results on Movielens-10M dataset. We observe that using popularity embeddings

greatly simulates the ItemPop algorithm, and the overlapped items even surpass 50% when TopK is above 40. Compared with other baselines like IPS and CausE with IOU less than 20%, DICE-pop is much more similar to ItemPop, which confirms that popularity embeddings indeed capture popularity bias. On the other hand, there is almost no overlapped items between DICE-int and ItemPop, proving that popularity bias is almost fully distilled from interest embeddings.

In DICE, two sets of embeddings are disentangled for the two causes, interest and popularity. We visualize the learned item embeddings in DICE using t-SNE [33]. Figure 3 shows the learned item embeddings, where *crosses* represent interest embeddings and *dots* represent popularity embeddings. With special design and direct supervision on disentanglement, the two sets of embeddings are far from each other, and they could even be approximately separated by a linear classifier (red line in the figure). Moreover, we divide all the items to three groups based on their popularity, which are popular, normal and unpopular. In Figure 3, items of different groups are painted in different colors. We observe that popularity embeddings are layered according to item popularity, where items of similar popularity are near in the embedding space. On the other hand, with respect to interest embeddings, items of different popularity are mixed with each other. Visualizations of the learned item embeddings illustrate the high quality of disentanglement in the proposed framework. Based on disentangled embeddings, reasonable interpretations could be made, which is crucial for recommendation.

3.4 Summary comparison of different discrepancy loss

We provide three options for discrepancy loss, L1-inv, L2-inv and *dCor*. We examine the three candidates on two datasets with two backbones. Overall, *dCor* attains better performance than L1-inv and L2-inv with over 2% improvements. However, *dCor* relies on heavy matrix computations which is much more time-consuming than L1-inv and L2-inv. Specifically, training with *dCor* (about 100s per epoch) as discrepancy loss is much slower than L1-inv and L2-inv (about 44s per epoch), which means L1-inv and L2-inv might be more appropriate for large scale applications.

4 Related work

Popularity bias Several works [1, 3, 4, 10, 11, 21, 34, 40] have investigated the role of popularity bias in recommender systems. Steck [40] examined the traded-off between popularity and accuracy, and proposed a relatively unbiased metric. Jannach *et al.* [21] analyzed the source of popularity bias, and propose two schemes to counter these biases. Unlike these previous approaches, we model popularity bias as one of the causes for observational data, and use causal embeddings to capture it.

Causal recommendation Recently causal inference is leveraged for unbiased recommendation [2, 9, 14, 15, 22, 24, 29, 37, 45, 46]. IPS is widely adopted, which impose lower weights for popular items. Specifically, the weight is set as the inverse of item popularity [2, 37, 46]. Bottou *et al.* [9] add max-capping on IPS value to reduce the variance of IPS. Gruson *et al.* [15] further add normalization which also achieved better results than plain IPS. Besides IPS, Bonner *et al.* [8] proposed CausE that requires a large biased dataset and a small unbiased dataset. Each user or item has two embeddings to perform MF on two datasets respectively. Unlike these approaches that bundle different causes into unified representations, our approach achieves causal recommendation with disentangled embeddings. In recommendation with explicit feedback, Sinha *et al.* [39] decomposed observed ratings to union of real ratings and recommender influence. With several strong assumptions, they attained a closed-form solution to recover real ratings from observational ratings based on SVD. However, these assumptions could not be adapted to the more prevalent implicit feedback setting. To our knowledge, our proposed approach is the first attempt to disentangle different causes in recommendation on implicit feedback.

Disentangled representation learning Learning representations in which different semantics are disentangled is crucial for robust use of neural models [5, 31, 38, 41]. Existing approaches mainly focus on computer vision [12, 13, 19, 20, 26]. For example, β-VAE [19] learns interpretable representations from raw images in an unsupervised manner. Disentangled representation learning in recommender systems was not explored until recently [32, 44]. These methods decompose user intent into finer granularity, such as the size or color of an item, while ignoring the bias in observational data. Unlike existing algorithms, our approach learns disentangled representations to model the causal relations in recommendation.

5 Conclusion

In this paper, we propose a general framework for disentangling user interest and popularity bias for recommendation with causal embeddings. Based on concise and reasonable assumptions, DICE consistently outperforms state-of-the-art algorithms with remarkable improvements. Analysis on disentanglement demonstrates that user interest and item popularity is largely independent in the two sets of embeddings. The learned embeddings are of high quality and interpretability, which is promising to explore novel applications using the learned disentangled representations. A particular meaningful direction for future work is extending DICE to more embedding based recommendation backbones. Overall, we believe disentangling interest and popularity opens new doors for unbiased learning in recommender systems.

References

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. Controlling popularity bias in learning-to-rank recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 42–46, 2017.

[2] Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. A general framework for counterfactual learning-to-rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 5–14, 2019.

[3] Punam Bedi, Anjali Gautam, Chhavi Sharma, et al. Using novelty score of unseen items to handle popularity bias in recommender systems. In *2014 International Conference on Contemporary Computing and Informatics (IC3I)*, pages 934–939. IEEE, 2014.

[4] Alejandro Bellogín, Pablo Castells, and Iván Cantador. Statistical biases in information retrieval metrics for recommender systems. *Information Retrieval Journal*, 20(6):606–634, 2017.

[5] Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Rosemary Ke, Sébastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher Pal. A meta-transfer objective for learning to disentangle causal mechanisms. *arXiv preprint arXiv:1901.10912*, 2019.

[6] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[7] James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. Citeseer, 2007.

[8] Stephen Bonner and Flavian Vasile. Causal embeddings for recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 104–112, 2018.

[9] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research*, 14(1):3207–3260, 2013.

[10] Rocío Cañamares and Pablo Castells. A probabilistic reformulation of memory-based collaborative filtering: Implications on popularity biases. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 215–224, 2017.

[11] Rocío Cañamares and Pablo Castells. Should i follow the crowd? a probabilistic analysis of the effectiveness of popularity in recommender systems. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 415–424, 2018.

[12] Emilien Dupont. Learning disentangled joint continuous and discrete representations. In *Advances in Neural Information Processing Systems*, pages 710–720, 2018.

[13] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[14] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. Offline a/b testing for recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 198–206, 2018.

[15] Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney, Samantha Hansen, Damien Tardieu, and Ben Carterette. Offline evaluation to make decisions about playlistrecommendation algorithms. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 420–428, 2019.

[16] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017.

[17] Joseph A. Harper, F. Maxwell anUntitled.texd Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4), December 2015.

[18] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.

[19] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *Iclr*, 2(5):6, 2017.

[20] Jun-Ting Hsieh, Bingbin Liu, De-An Huang, Li F Fei-Fei, and Juan Carlos Niebles. Learning to decompose and disentangle representations for video prediction. In *Advances in Neural Information Processing Systems*, pages 517–526, 2018.

[21] Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. What recommenders recommend: an analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction*, 25(5):427–491, 2015.

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

[23] Thorsten Joachims and Adith Swaminathan. Counterfactual evaluation and learning for search, recommendation and ad placement. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 1199–1201, 2016.

[24] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 781–789, 2017.

[25] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[26] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

[28] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306, 2011.

[29] Dawen Liang, Laurent Charlin, and David M Blei. Causal inference for recommendation. In *Causation: Foundation to Application, Workshop at UAI*. AUAI, 2016.

[30] Yiming Liu, Xuezhi Cao, and Yong Yu. Are you influenced by others when rating? improve rating prediction by conformity modeling. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 269–272, 2016.

[31] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, pages 4114–4124, 2019.

[32] Jianxin Ma, Chang Zhou, Peng Cui, Hongxia Yang, and Wenwu Zhu. Learning disentangled representations for recommendation. In *Advances in Neural Information Processing Systems*, pages 5712–5723, 2019.

[33] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

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

[35] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic Books, 2018.

[36] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

[37] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. Recommendations as treatments: debiasing learning and evaluation. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*, pages 1670–1679, 2016.

[38] Bernhard Schölkopf. Causality for machine learning. *arXiv preprint arXiv:1911.10500*, 2019.

[39] Ayan Sinha, David F Gleich, and Karthik Ramani. Deconvolving feedback loops in recommender systems. In *Advances in neural information processing systems*, pages 3243–3251, 2016.

[40] Harald Steck. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 125–132, 2011.

[41] Raphael Suter, Đorđe Miladinović, Bernhard Schölkopf, and Stefan Bauer. Robustly disentangled causal mechanisms: Validating deep representations for interventional robustness. *arXiv preprint arXiv:1811.00007*, 2018.

[42] Gábor J Székely and Maria L Rizzo. Brownian distance covariance. *The annals of applied statistics*, pages 1236–1265, 2009.

[43] Gábor J Székely, Maria L Rizzo, Nail K Bakirov, et al. Measuring and testing dependence by correlation of distances. *The annals of statistics*, 35(6):2769–2794, 2007.

[44] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tonog Xu, and Tat-Seng Chua. Disentagnled graph collaborative filtering. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.

[45] Yixin Wang, Dawen Liang, Laurent Charlin, and David M Blei. The deconfounded recommender: A causal inference approach to recommendation. *arXiv preprint arXiv:1808.06581*, 2018.

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

[47] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 974–983, 2018.