Knowledge and Information Systems
https://doi.org/10.1007/s10115-025-02404-7

RESEARCH

Deconfounding representation learning for mitigating latent confounding effects in recommendation

Guixian Zhang[1,2,3] · Guan Yuan[1,3] · Debo Cheng[4] · Lin Liu[4] · Jiuyong Li[4] · Ziqi Xu[5] · Shichao Zhang[2]

Received: 24 July 2024 / Revised: 5 March 2025 / Accepted: 7 March 2025

Abstract
Contrastive learning has gained significant attention in the field of recommender systems due to its ability to learn highly expressive representations with limited labels. However, historical user–item interaction data used for recommender systems often contain confounders, thereby establishing spurious correlations between user preferences and confounders during self-supervised training and misleading recommender systems to use these correlations as shortcuts for generating recommendations. Existing approaches for debiasing usually involve manually identifying observed confounders, but they are often tailored to specific situations and overlook latent confounders. To address this challenging problem, we propose a Deconfounding Graph Contrastive Learning (DeGCL) method to provide deconfounding recommendations by adjusting for a learned deconfounding representation from interaction data, using the back-door adjustment strategy. DeGCL learns the representation to capture latent confounding effects in observational data between users and items. It artificially adds interactions and noise to create contrastive views, which help deconfound the model. By adjusting for the learned representation, DeGCL mitigates latent confounding effects in training downstream recommendation models. Experiments on two real-world datasets demonstrate that our method outperforms state-of-the-art methods, suggesting its potential to provide more effective recommendations in practice.

Keywords Recommender systems · Contrastive learning · Causality-inspired machine learning · Graph neural network

1 Introduction

In the era of information overload, recommender systems have become indispensable tools [1] that align user interests with relevant content. The utility of these systems extends beyond just enhancing user experience; they confer significant commercial value to online platforms across domains, including e-commerce [2], social media [3], job matching [4], and others. However, prevailing recommendation models rely solely on historical user–system interactions to infer user preferences [4–9]. Problematically, these models presume that all interaction data objectively indicate a user's actual preferences for specific items. In reality, user interactions reflect a confluence of confounders beyond innate interests, including sit-

Extended author information available on the last page of the article

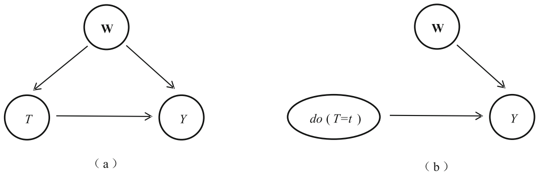


Fig. 1 Two causal graphs are utilised to illustrate the differences between conventional recommendation models (**a**) and the proposed DeGCL (**b**). In both graphs, the representation of users and items is considered as the treatment (denoted as T), while the feedback from users is regarded as the outcome (Y). **W** denotes the latent confounders. To address the latent confounding effects in a recommender system, we propose a learnable representation to absorb these confounding effects and satisfy the back-door adjustment criterion

uational contexts, external nudging, and unintentional biases. Consequently, models based solely on interaction data risk perpetuating these biases, leading to recommendations that may not accurately align with users' actual needs.

In order to address bias in recommendation scenarios, the most intuitive method would be to adopt the idea of randomised controlled trials (RCT) [10], which involves making multiple recommendations to users to find their real interests. However, RCT faces significant limitations in real-world applications due to ethical and practical constraints [5, 11, 12]. Hence, most methods use observational data to establish an effective recommendation system, but they require researcher expertise to identify influential confounders. Confounders, which affect both users and items, can distort the relationships between user/item data and the recommendations made by recommender systems [5, 10, 13]. These confounders are covariates that result in spurious correlations into model training, often poorly reflecting users' true preferences [7, 14]. Therefore, it is vital to identify and address sources of confounding in interaction data, a point emphasised in prior studies [2, 5, 15].

Causal inference is a statistical framework for identifying and quantifying causal relationships between variables [10]. Recently, causal inference techniques have been widely used in artificial intelligence systems with promising results [16–19]. It provides powerful tools to identify and mitigate confounding effects. For a clearer presentation of our method, we describe causal inference in more detail in Sect. 3. In recommender systems, these methods often involve using experts' knowledge to identify specific confounders, such as popularity [20–22], position [23–25] and exposure [8, 14, 26]. Typically, these methods represent the relationship between observable confounders (**W**), user and item representations (T), and user feedback (Y) as represented in a directed acyclic graph (DAG) in Fig. 1a. However, real-world systems often involve many confounders, some of which are not easily detected through manual selection. Beyond the 'observed confounders' that experts can manually identify, there are 'unobserved or latent confounders'. These are difficult to detect or measure. In real-world systems, the complexity and variability of these confounders make it impractical to design specific models for each confounder.

In the recommender system, real-world data tend to be limited-labelled and sparse [27]. Since contrastive learning does not rely on labelled data and can make good use of data, researchers have begun to build contrastive learning-based recommender systems [28–30] and conduct debiasing studies for certain types of bias [31–33]. However, since historical interaction data are observational rather than experimental, confounders in the data are often difficult to pinpoint. Past research has demonstrated the presence of multiple confounders in historical interaction data, such as exposure bias [34], position bias [35], popularity bias [31], and conformity bias [36]. However, the effects of these confounders are inconsistent [37], and the complexity associated with this inconsistency poses a challenge in designing

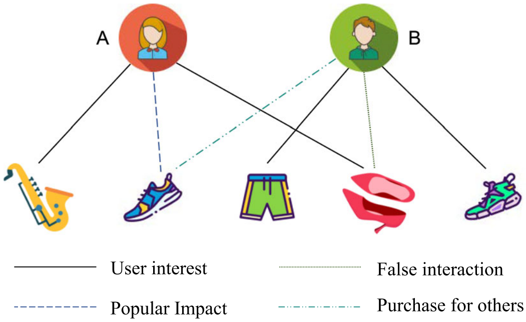


Fig. 2 A toy example about latent confounding effects in recommender systems

deconfounding models for recommender systems. In particular, under the contrastive learning paradigm, models are susceptible to distorted understanding due to the influence of latent confounders during the self-supervision process [38] and misleading recommender systems to use these correlations as shortcuts for recommendations. Existing contrastive learning algorithms do not pay attention to latent confounders, which poses additional challenges in accurately capturing user preferences. To illustrate the negative impact of latent confounders, we provide a toy example in Fig. 2. In this example, two users display similar historical interactions. User A clicked on high heels and musical instruments out of interest and on pop shoes due to their popularity. User B clicked on shorts and basketball shoes because of interest, clicked on high heels accidentally, and purchased pop shoes for someone else. Despite the apparent overlap in historical interactions, recommending similar items to both users might be misguided due to their divergent interests. The components of these interactions are complex and variable, influenced by numerous latent confounders.

To address the spurious correlations caused by the latent confounders, we propose a self-supervised learning-based method named Deconfounding Graph Contrastive Learning (DeGCL). DeGCL is designed to learn a deconfounding representation that absorbs the latent confounders represented by $\mathbf{W}$, thereby establishing unbiased effects between T (treatment variable) and Y (outcome variable). The relationships among $\mathbf{W}$, T and Y are represented in a direct acyclic graph (DAG), as shown in Fig. 1b. To ensure the learning of an effective and robust deconfounding representation, we introduce spurious interactions and small noises into the data to enhance the magnitude and diversity of the confounding effects, respectively. Subsequently, the learned deconfounding representation is employed to adjust for the confounding effects in the user's representation and the item's representation through back-door adjustment [10]. The detailed definition of back-door adjustment is explained in Sect. 3 "Preliminaries". Our main contributions are summarised as follows:

- We propose and design a learnable deconfounding representation (denoted by $\mathbf{D}$) to simultaneously absorb the latent confounding effects in the recommender system, in order to facilitate the model's deconfounding through back-door adjustment.
- We propose a deconfounding model based on contrastive learning, named DeGCL, and design two augmentation strategies to help the model perform better in deconfounding.
- We have conducted extensive experiments on two real-world datasets, and the results demonstrate the effectiveness of our DeGCL method.

The rest of the paper is organised as follows: We briefly review related work in Sect. 2. Section 3 addresses the preliminaries for causal inference in recommender systems. The proposed DeGCL method is introduced in Sect. 4. In Sect. 5, we experimentally verify and analyse the effectiveness of the DeGCL method. Finally, the paper is concluded in Sect. 6.

2 Related work

2.1 Self-supervised contrastive learning

Recently, researchers have been actively exploring methods to reduce model dependency on labels due to the high cost of obtaining labelled data in the real world [39–43]. Self-supervised learning has emerged as a promising approach [28, 44–46], leveraging self-supervised signals extracted directly from the data to train deep neural networks without manual annotation. This paradigm has demonstrated remarkable scalability and generalisability across diverse domains [47, 48].

Contrastive learning has gained significant popularity as one of the most popular self-supervised learning techniques due to its ability to learn rich semantic representations for downstream tasks [49, 50]. Contrastive learning aims to uncover meaningful information within the data. By employing a carefully designed loss function, typically the InfoNCE loss [51], it brings positive sample pairs closer together in the feature space while simultaneously pushing negative sample pairs further apart. The success of contrastive learning has been extensively studied from multiple perspectives [52–54], these studies have confirmed that contrastive learning serves as an effective means to extract latent information from the data, revealing valuable insights that might otherwise remain hidden. Wang and Isola [55] emphasised alignment and homogeneity as the key to contrastive loss, which was introduced into graph neural network-based recommender systems by Wang et al. [32]. However, despite its excellent results, existing studies [56, 57] have demonstrated that contrastive learning introduces an inductive bias, necessitating efforts to debias contrastive learning methods.

2.2 Graph neural networks in recommender systems

Due to the outstanding performance of graph neural networks (GNNs) on non-Euclidean data [17, 58–61], researchers have developed numerous GNN-based recommender systems [62] to capture the intricate interaction relationships within data. For instance, NGCF [63] and SHT [45] employ GNNs to model the user–item interaction graph, generating node representations through message propagation. Liu et al. [64] help the model capture user preferences by reasoning about historical interactions. Recently, GNN-based recommendations have incorporated techniques to improve representation quality and mitigate biases. The emergence of self-supervised contrastive learning in recommender systems [46] has notably addressed the issue of data sparsity. SGL [28] utilises self-supervision to mitigate overfitting to sparse signals, while MHCN [44] leverages hypergraph modelling to comprehend complex interactions.

Furthermore, MIXGCF [29] synthesises challenging negative samples to cover the latent space better. DirectAU [32] aligns user and item embeddings for more precise matching, while mitigating the exposure bias in the recommender system. NCL [30] learns about contrastive loss at the semantic level by constructing semantic neighbours. SimGCL [31] shows noise injection implicitly balances preference distribution. LightGCL [33] augments embeddings using singular value decomposition to reduce bias. GraphAug [38] looks at the noise impact of data augmentation and adaptively adjusts through information bottlenecks. However, deconfounding methods for the latent confounding effects in graph contrastive learning recommender systems are not yet available.

3 Preliminaries

The causal graph is widely used to represent the causal relationships between variables [10]. In this work, we use a directed acyclic graph (DAG), which is a graph containing only directed edges $\rightarrow$ with no directed cycles (i.e. no directed path where the start and end points are the same node), to represent the data generation process. The causal DAG is defined as follows:

Definition 1 (*Causal DAG* [10]) A causal DAG is a DAG $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents the set of variables and $\mathcal{E}$ represents the set of directed edges which capture the direct causal relationships among the variables.

In causal inference, the *treatment variable* refers to the intervention or manipulation applied in a study, with the objective of determining its causal effect on the outcome of interest [10]. For instance, in a study, the treatment variable could be a medication, a behavioural intervention, a policy change, or any other form of intervention.

The *outcome variable* represents the result that we aim to investigate regarding the effects of the treatment variable [10]. For example, in a study examining the efficacy of a new drug in reducing blood pressure, the outcome variable would be the degree of blood pressure reduction.

From the perspective of causal inference, as shown in Fig. 1a, we can conceptualise the embedding vector (i.e. the representation of users and items) as the treatment variable (T), the feedback from users as the outcome variable (Y), and the confounders as $\mathbf{W}$ in the causal graph. The primary objective of our work is to mitigate the influence of confounders on the embedding vector. A key component in achieving this is the do-operation [10], which is defined as follows:

Definition 2 ($do{-}operation$ [10]) The do-operation, denoted as $do(T = t)$, represents an intervention where the variable T is forcefully set to the value t. The expression $P(Y = y \mid do(T = t))$ denotes the conditional probability of observing $Y = y$ given that we intervene to set T to t. This probability measurement isolates the causal effect of the intervention by excluding the influence of confounders.

In an ideal scenario, we would have both interventional and observational data, allowing us to directly perform the do-operation. However, historical interaction data are only observational. As is well-known, Pearl et al. [10] proposed that if a set of variables $\mathbf{W}$ satisfies the back-door criterion, then the do-operation can be transformed into an expression without the do-operation. The back-door criterion is defined as follows:

Definition 3 (*Back-door Criterion* [10]) In a causal graph G, given an ordered pair of variables (T, Y), a set of variables $\mathbf{W}$ satisfies the back-door criterion concerning (T, Y) if it satisfies the following conditions: (1) No node in $\mathbf{W}$ is a descendant of T. (2) $\mathbf{W}$ blocks every path between T and Y that contains an arrow into T.

If a set of variables $\mathbf{W}$ satisfies the back-door criterion w.r.t., the pair (T, Y), then the causal effect of T on Y can be identified by adjusting for $\mathbf{W}$, i.e.

$$P(Y=y|do(T=t)) = \sum_{\mathbf{W}} P(Y=y|do(T=t), \mathbf{W}=w)P(\mathbf{W}=w|do(T=t)), \quad (1)$$

$$= \sum_{\mathbf{W}} P(Y=y|T=t, \mathbf{W}=w)P(\mathbf{W}=w|do(T=t)), \quad (2)$$

$$= \sum_{\mathbf{W}} P(Y=y|T=t, \mathbf{W}=w)P(\mathbf{W}=w). \quad (3)$$

The back-door adjustment is formulated as the following theorem:

Theorem 1 *(Back-door Adjustment) The back-door adjustment formula for estimating the causal effect of T on Y given the back-door set $\mathbf{W}$ is given by*

$$P(Y \mid do(T)) = \sum_{\mathbf{W}} P(Y \mid T, \mathbf{W})P(\mathbf{W}),$$

where $P(Y \mid T, \mathbf{W})$ represents the conditional probability of Y given T and $\mathbf{W}$, and $P(\mathbf{W})$ is the probability distribution of $\mathbf{W}$.

The back-door adjustment allows us to condition on a back-door set to mitigate the effects of confounders. However, in recommender systems, directly adjusting for confounders is impractical due to the presence of latent confounders. In this work, we propose a novel causal graph, as illustrated in Fig. 1b for addressing latent confounding bias. In this causal graph, we aim to design a deconfounding representation, $\mathbf{D}$, to capture information about both measured and latent confounders. Once learned, the representation $\mathbf{D}$ can be used to effectively achieve back-door adjustment within the recommender system.

4 Methodology

4.1 Overview

Our proposed Deconfounding Graph Contrastive Learning (DeGCL) method is rooted in the framework of self-supervised learning, which encompasses both the primary supervised task and a self-supervised learning component. We present the workflow of our DeGCL in Fig. 3. In the training phase, we concatenate the deconfounding representation with the node representation and train it by contrastive learning. In the inference phase, the deconfounding representation at this point has absorbed the latent confounding effect and achieved the deconfounding of the node representation. The summary of notations is shown in Table 1.

4.2 Construction of deconfounding representations

Existing causal learning methods in recommender systems typically target only a single, pre-specified confounder [13, 65]. However, real-world environments often contain large-scale, multivariate confounders, including numerous latent factors that can distort observed patterns. Recommender systems that do not take into account the influence of latent confounders will unconditionally favour recommending items that are similar to those they have interacted with before, thereby reducing the likelihood that users will encounter content that broadens their horizons or challenges their existing beliefs. However, due to the existence of latent confounders, back-door adjustment is not suitable directly for latent confounders and observed confounders.

To address this issue, we propose constructing a deconfounding representation based on the causal graph for back-door adjustment. The key objective is to encapsulate the multidimensional latent confounding effects within this representation. We integrate this representation into our model as an additional input alongside standard user–item data. If the confounder representation is omitted during testing, predictions are based solely on the deconfounded user and item embeddings. In this way, DeDCL reduces reliance on spurious correlations and eliminates the need to identify specific confounding variables.

We initialise the deconfounding representation by constructing random variables for the user and item representations, respectively. These are then integrated into the original representations. Let $\tilde{\mathbf{E}}_u^{(0)}$ and $\tilde{\mathbf{E}}_v^{(0)}$ denote the initial embedding representation of a node, which is obtained from the original user and item features. We define $\mathbf{D}_u^0 \sim \mathcal{U}(0,1)$ and $\mathbf{D}_v^0 \sim \mathcal{U}(0,1)$ as random variables for the user and item deconfounding representations, respectively. The integrated representations are then formulated as

$$\mathbf{E}_u^{(0)} = [\tilde{\mathbf{E}}_u^{(0)}; \mathbf{D}_u^0], \tag{4}$$

$$\mathbf{E}_v^{(0)} = [\tilde{\mathbf{E}}_v^{(0)}; \mathbf{D}_v^0], \tag{5}$$

where $\mathbf{E}_u^{(0)}$ and $\mathbf{E}_v^{(0)}$ are the new representations concatenating the original and deconfounding representations, respectively. Previous work [66] has demonstrated that uniform distributions allow learned representations to unfold fully in the embedding space, enhancing expressiveness. Hence, our DeDCL method, which employs the generation of representations from a uniform distribution, contributes to the uniformity of these representations

4.3 Enforcing the deconfounding representation to absorb latent confounding effects

Simply appending a trainable vector to the original features is inadequate for capturing complex confounding effects, particularly involving latent confounders. It is necessary to optimise an auxiliary loss function that explicitly updates the deconfounding representation. Let $P(\hat{Y})$ denote the predicted probability of a user–item match from the recommendation system. Suppose there exists an debiased embedding, $T_C \sim P(T \mid Y)$, which reflects only the intrinsic user–item preferences unaffected by confounders. Conversely, let $T_N \sim P(T \mid Y, \mathbf{W})$ represent the biased real-world embedding influenced by confounders. A conventional recommendation system model is typically represented as $P(\hat{Y} \mid T_N)$. In order to obtain the debiased embedding T_C, we utilise the deconfounding representation $\mathbf{D}$ to achieve the back-door adjustment of T, i.e. to mitigate the effect of the confounders W. In this case, DeGCL redefines the outputs as $P(\hat{Y} \mid T_C)$. However, $\mathbf{D}$ is not guaranteed to completely absorb the latent confounding effects caused by $\mathbf{W}$, and T_C is still marginally biased. To ensure that $\mathbf{D}$ is effectively updated during the training process, we utilise $\mathcal{L}_{\text{diff}}$ to ensure that the difference between the trained user / item deconfounding representation $(\mathbf{D_u^1}, \mathbf{D_v^1})$ and the initial user/item deconfounding representation $(\mathbf{D_u^0}, \mathbf{D_v^0})$.

$$\mathcal{L}\text{diff} = \sum_{(u,v)\in\mathcal{B}} \frac{1}{N} \sum_{n=1}^{N} (\mathbf{d}_{u,n}^1 - \mathbf{d}_{u,n}^0)^2 + \frac{1}{M} \sum_{m=1}^{M} (\mathbf{d}_{v,m}^1 - \mathbf{d}_{v,m}^0)^2, \tag{6}$$

where N is the number of the items, M is the number of the users, and $\mathcal{B}$ is a mini-batch.

Since confounders are embedded in historical interaction data, we can optimise $\mathbf{D}$ as a confounder absorber while training the recommender system on biased data. We introduce

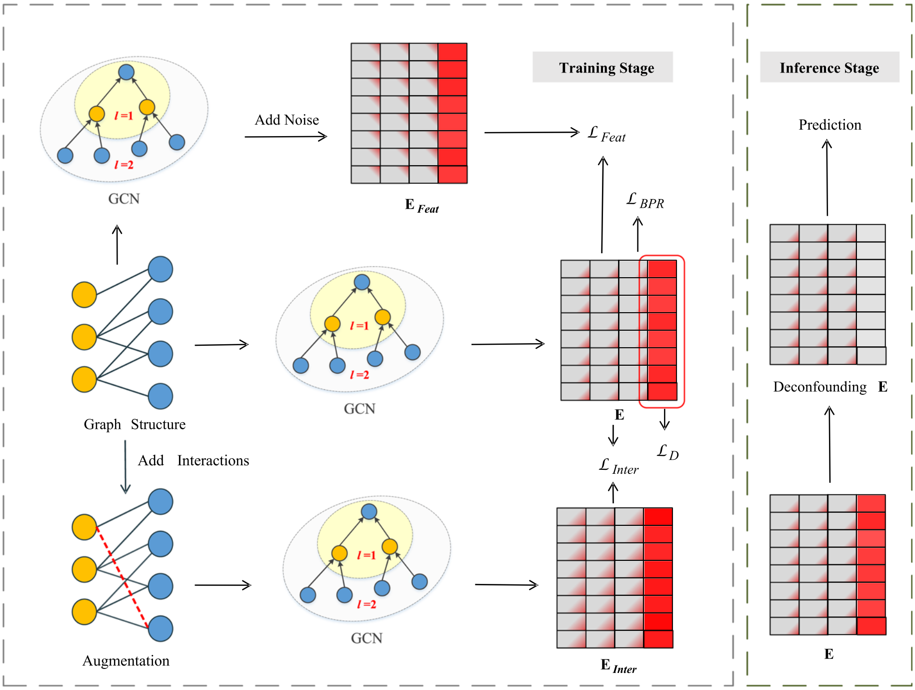


Fig. 3 Our proposed DeGCL method is structured as follows: we concatenate the deconfounding representation with the representation from the first epoch, and then proceed to incorporate it into subsequent training phases. The red part of the figure denotes the learned deconfounding representation $\mathbf{D}$. In the inference stage, we remove $\mathbf{D}$ to realise the deconfounding of the node representations

Table 1 The summary of notations and corresponding descriptions

Notation	Description
$\tilde{\mathbf{U}}$	Original user representation
$\tilde{\mathbf{V}}$	Original item representation
$\mathbf{D}_u^0$	Initialised user deconfounding representation
$\mathbf{D}_v^0$	Initialised item deconfounding representation
$\mathbf{E}$	Node embedding
L	Number of layers
$\tilde{\mathbf{A}}$	Normalised undirected adjacency matrix
$\mathcal{B}$	Mini-batch
$\mathbf{D}_u^1$	Trained user deconfounding representation
$\mathbf{D}_v^1$	Trained item deconfounding representation

a bias loss function that maximises the correlation between $\mathbf{D}$ and the biased prediction $P(\hat{Y} \mid T_C, T_N, \mathbf{D})$. Intuitively, this encourages $\mathbf{D}$ to absorb as much of the latent confounding effects from T_N, thereby purging confounders from the embedding. Consequently, $\mathbf{D}$ acts as a ‘vial’ to encapsulate multivariate confounding signals. We adopt a negative approach to the Bayesian personalised ranking (BPR) loss [67] for learning the deconfounding representation in the following form.

$$\mathcal{L}_{\text{bias}} = \sum_{(s,i,j)\in\mathcal{B}} \log\left(\sigma\left({\mathbf{d}_{u,s}^{1}}^{\top}\mathbf{d}_{v,i}^{1} - {\mathbf{d}_{u,s}^{1}}^{\top}\mathbf{d}_{v,j}^{1}\right)\right), \tag{7}$$

where $\mathcal{B}$ is a mini-batch, i and j represent a pair of positive and negative items of user u, and σ is a activation function. When $\mathcal{L}_{\text{bias}}$ is larger, it means that $\mathbf{D}$ contains more information affecting the recommendation. Therefore, we derive the loss function for learning the deconfounding representation.

$$\mathcal{L}_D = \mathcal{L}\text{diff} + \alpha * \mathcal{L}_{\text{bias}}, \tag{8}$$

where α is a hyperparameter.

4.4 Construction of treatments

To better utilise the limited and sparse historical data, we employ a contrastive learning paradigm for treatment construction, which is divided into two phases: the training stage and the inference stage.

Training DeGCL uses *LightGCN* [62] as the backbone to learn node representations

$$\mathbf{E} = \frac{1}{1+L}\left(\mathbf{E}^{(0)} + \tilde{\mathbf{A}}\mathbf{E}^{(0)} + \ldots + \tilde{\mathbf{A}}^{L}\mathbf{E}^{(0)}\right), \tag{9}$$

where $\mathbf{E}^{(0)}$ represents the initial node embedding defined in Sect. 4.2 incorporating the deconfounding representation, L is the number of layers, and $\tilde{\mathbf{A}}$ is the normalised undirected adjacency matrix of historical interaction data.

While we assume that there must be potential confounders in real-world data, we do not discern which interactions originate from causality and which from non-causality. To address this issue, we devise an interaction enrichment method that introduces unknown latent confounders into the data. Given that a significant disparity in node degrees within graph neural networks can result in bias against low-degree nodes, our approach prioritises these nodes. Specifically, for nodes in the posterior $\varphi\%$ by degree (with the size of n), we randomly create connections with nodes of degree less than ϕ, repeating this process n times to obtain a new adjacency matrix $\mathbf{A}'$. The specific steps of the algorithm are presented in Algorithm 1.

We use the new adjacency matrix $\mathbf{A}'$ generated by Algorithm 1 to train the embeddings

$$\mathbf{E}_{\text{Inter}} = \frac{1}{1+L}\left(\mathbf{E}^{(0)} + \tilde{\mathbf{A}}'\mathbf{E}^{(0)} + \ldots + {\tilde{\mathbf{A}}'}^{L}\mathbf{E}^{(0)}\right). \tag{10}$$

On the other hand, noise perturbations are often encountered in real-world recommender systems due to the inherent diversity and randomness of user behaviour. We add a small amount of noise to the node representation to serve as the second data augmentation method. This provides enough diversity for model training and ensures model robustness. It is performed from features to form a contrastive loss $\mathcal{L}_{\text{Feat}}$, and throughout the training process, we introduce a very small amount of random uniform noise

$$\Delta = \omega \odot \text{sign}\left(\mathbf{e}_i\right), \omega \in\sim U(0,1), \tag{11}$$

Algorithm 1 Adding spurious interactions

Require: Graph adjacency matrix $\mathbf{A}$, Low-degree threshold ϕ, sampling ratio $\varphi\%$
Ensure: Updated graph adjacency matrix $\mathbf{A}'$
$N \leftarrow$ number of nodes in $\mathbf{A}$
$\mathcal{D} \leftarrow$ degree of each node in $\mathbf{A}$
$n \leftarrow \lfloor \varphi\% \cdot N \rfloor$
$V' \leftarrow$ top n nodes by degree in $\mathbf{A}$
$\mathbf{A}' \leftarrow$ copy of $\mathbf{A}$
for $i \leftarrow 1$ to n **do**
  for $j \leftarrow 1$ to n **do**
    $v \leftarrow V'[j]$
    $\mathcal{D}_{\text{less}} \leftarrow \{u \in V' : \mathcal{D}[u] < \phi\}$
    $u \leftarrow$ randomly select a node from D_{less}
    $\mathbf{A}'[u][v] \leftarrow 1$
    $\mathbf{A}'[v][u] \leftarrow 1$
  end for
end for
return $\mathbf{A}'$

which makes Δ and $\mathbf{e}_i$ still have the same tendency, $||\Delta||_2 = \xi$ and ξ is a very small constant. From this, we can obtain a feature-level representation for data augmentation as follows:

$$\begin{aligned}\mathbf{E}_{\text{Feat}} = \frac{1}{1+L}\Big(&\mathbf{E}^{(0)} + \left(\tilde{\mathrm{A}}\mathbf{E}^{(0)} + \Delta^{(1)}\right) + \left(\tilde{\mathrm{A}}\left(\tilde{\mathrm{A}}\mathbf{E}^{(0)} + \Delta^{(1)}\right) + \Delta^{(2)}\right) + \ldots \\ &+ \left(\tilde{\mathrm{A}}^{L}\mathbf{E}^{(0)} + \tilde{\mathrm{A}}^{L-1}\Delta^{(1)} + \ldots + \tilde{\mathrm{A}}\Delta^{(L-1)} + \Delta^{(L)}\right)\Big)\end{aligned} \tag{12}$$

This can be understood in the real world as follows: when the user or item is affected by a certain factor (e.g. weather or brand ambassadors), the user's interaction with the item seems to change accordingly. Contrastive loss (*InfoNCE* [51]) is used to align these differently enhanced representations in the following form.

$$\mathcal{L}_{cl} = \sum_{i \in \mathcal{B}} -\log \frac{\exp\left(\mathbf{z}_i'^{\top}\mathbf{z}_i''/\tau\right)}{\sum_{j \in \mathcal{B}} \exp\left(\mathbf{z}_i'^{\top}\mathbf{z}_j''/\tau\right)}, \tag{13}$$

where $\mathbf{z}'$ and $\mathbf{z}''$ are representations of nodes i, j in batch $\mathcal{B}$ from augmented graphs, respectively, and τ is temperature. When the loss is computed for the original graph with different augmentations, $\mathcal{L}_{cl}$ is made $\mathcal{L}_{\text{Inter}}$ and $\mathcal{L}_{\text{Feat}}$, respectively. For the recommendation loss, we use BPR loss [67] to measure the quality of recommendations in the following form.

$$\mathcal{L}_{\text{BPR}} = -\sum_{(u,i,j) \in \mathcal{B}} \log\left(\sigma\left(\mathbf{e}_u{}^{\top}\mathbf{e}_i - \mathbf{e}_u^{\top}\mathbf{e}_j\right)\right), \tag{14}$$

where $\mathcal{B}$ is a mini-batch, $\mathbf{e}_j$ is a randomly sampled item, and σ is a activation function.

This framework synergistically combines graph convolution, noise injection, and edge augmentation to learn the representation $\mathbf{D}$, and the final loss of DeDCL is defined as

$$\mathcal{L}_{\text{main}} = \mathcal{L}_{\text{BPR}} + \mathcal{L}_{\text{Feat}} + \mathcal{L}_{\text{Inter}}. \tag{15}$$

Inference With Eq. 6 and Eq. 7 in the training stage, $\mathbf{D}^1$ absorbs the latent confounding effect well. Therefore, we use the initialised $\mathbf{D}^0$ instead of $\mathbf{D}^1$ in the inference stage, i.e.

we use a very small uniform noise to simulate the unbiased environment, so as to mitigate the latent confounding effect in the representations and achieve the intended purpose of the model.

5 Experiments

5.1 Experimental setup

5.1.1 Datasets

We validate the performance of DeGCL on two real-world benchmark datasets: MovieLens-1M [68] [1] and Yelp2018 [63]. The MovieLens-1M dataset contains 575,281 interactions with 3,492 movies by 6,038 MovieLens users, with a density of 0.0273. The Yelp2018 dataset includes 1,561,406 interactions involving 3,668 Yelp users across 38,048 products, with a density of 0.0013. For rigorous evaluation, we split both datasets into training, validation, and test sets in a 70/10/20 ratio. To ensure objective comparison, we report the average of five runs for all our experiments.

5.1.2 Baselines

Given that our proposed DeGCL is developed using the self-supervised graph contrastive learning approach, we compare it against the state-of-the-art methods that also utilise self-supervised graph contrastive learning for recommendations. The baselines for our comparison include:

- LightGCN [62] implements an efficient graph convolutional network using simple aggregators.
- SGL [28] introduces self-supervised learning for graph neural network methods, which in turn changes the data sparsity problem.
- MIXGCF [29] proposes a method to synthesise difficult negative samples in continuous space to obtain richer information.
- NCL [30] learns about contrastive loss at the semantic level by constructing semantic neighbours.
- SimGCL [31] adds uniformity to the model by using uniform noise as data augmentation, while mitigating the popularity bias in the recommender system.
- DirectAU [32] optimises the learning objectives in terms of both alignment and uniformity thus improving the representation quality, while mitigating the exposure bias in the recommender system.
- LightGCL [33] improves the data sparsity and popularity bias problem by using singular value decomposition for augmentation, while mitigating the popularity bias in the recommender system.
- GraphAug [38] denoises the data augmentation information and then adaptively adjusts the contrastive view based on the information bottleneck.

[1] https://grouplens.org/datasets/movielens/1m/.

Table 2 Performance of DeDCL in comparison with state-of-the-art methods on the MovieLens-1M dataset

Method	Recall@10	NDCG@10	Recall@30	NDCG@30	Recall@50	NDCG@50
LightGCN	0.14752	0.26594	0.22636	0.27332	0.36743	0.28914
SGL	<u>0.17369</u>	0.29595	0.31609	0.30312	0.41035	0.33544
MIXGCF	0.17159	<u>0.29768</u>	0.32545	0.30673	0.42049	0.33044
NCL	0.16210	0.28612	0.31369	0.29668	0.40741	0.32326
SimGCL	0.17169	0.29586	0.32842	<u>0.30839</u>	0.42522	<u>0.33632</u>
DirectAU	0.16516	0.26177	0.30408	0.28252	0.39242	0.30772
LightGCL	0.15989	0.28237	0.30685	0.29210	0.40082	0.31861
GraphAug	0.17034	0.29095	<u>0.32883</u>	0.30633	<u>0.42761</u>	0.33502
DeGCL	**0.17747**	**0.30360**	**0.33727**	**0.31607**	**0.43263**	**0.34362**

The best results in the table are bolded, and the runner-up results are underlined

5.1.3 Evaluation metrics

We evaluate recommendation performance using two standard metrics: Recall@K and NDCG@K. These metrics examine the quality of top-K item suggestions for each user. Specifically, Recall@K measures the fraction of test set user–item interactions that are retrieved in the top-K recommendations, thereby evaluating the model's ability to recover ground truth interactions. On the other hand, NDCG@K balances the relevance of recommendations with their rank position, assigning higher scores to hits at top ranks, thus assessing how effectively a model prioritises items that users are likely to prefer. We calculate both metrics at varying K values of 10, 30, and 50, providing a comprehensive view of the model's performance across different sizes of recommendation lists. A smaller K value assesses precision in the most highly ranked items, while a larger K value evaluates the model's ability to generate a broader set of relevant suggestions.

5.1.4 Hyperparameters

For a fair comparison, we optimise all models using the same experimental protocol. We initialise hyperparameters based on values reported in original papers when available or strong defaults from prior work otherwise. Specifically, we use a hidden dimension of 64 with Xavier normal initialisation, the Adam optimiser with learning rate 10^{-3}, graph convolutional layers of 4, L_2 regularisation of 10^{-4} and the batch size of 2,048. We tune the deconfounding loss ratio α via grid search, selecting 0.2. All implementations are trained for 500 epochs with early stopping based on validation Recall@50 and NDCG@50.

5.2 Performance comparison

In this paper, all self-supervised graph contrastive learning methods are evaluated using LightGCN as a baseline. Based on the results in Tables 2 and 3, we have the following observations:

- All self-supervised graph contrastive learning methods effectively improve LightGCN under various settings. LightGCN, which relies solely on the graph convolution of observed user–item interaction graphs for representation, may overfit sparse signals from

Table 3 Performance of DeDCL in comparison with state-of-the-art methods on the Yelp2018 dataset

Method	Recall@10	NDCG@10	Recall@30	NDCG@30	Recall@50	NDCG@50
LightGCN	0.02857	0.03253	0.06865	0.04771	0.09918	0.05879
SGL	0.03897	0.04438	0.08983	0.06338	0.12820	0.07729
MIXGCF	0.04110	0.04682	<u>0.09467</u>	0.06677	0.13294	0.08054
NCL	0.03917	0.04459	0.09001	0.06366	0.12895	0.07776
SimGCL	0.04178	<u>0.04835</u>	0.09437	0.06723	0.13297	<u>0.08156</u>
DirectAU	<u>0.04180</u>	0.04827	0.09445	<u>0.06770</u>	<u>0.13315</u>	0.08175
LightGCL	0.03786	0.04332	0.08604	0.06125	0.12346	0.07481
GraphAug	0.03972	0.04501	0.09077	0.06410	0.12927	0.07806
DeGCL	**0.04286**	**0.04891**	**0.09537**	**0.06785**	**0.13655**	**0.08274**

The best results in the table are bolded, and the runner-up results are underlined

historical data. Self-supervised contrastive loss encourages representations that capture robust user and item semantics, improving generalisation compared to LightGCN's basic feature propagation.

- SGL's structural data augmentation allows the model to identify user interests more finely based on sparse signals. MIXGCF's enhanced negative sampling also improves the ranking of popular items. However, these techniques may overfit historical interactions, which help understand a user's neighbourhood, which makes them more adaptable to situations with smaller K. However, they may fail to capture global confounders affecting long-tailed items, which affects generalisation ability. Semantic enhancement of NCL shows some advantages, but simple clustering needs to learn causal relationships between representations well.
- DirectAU demonstrates that better uniformity helps to improve recommendation performance, leading to competitive results. Adding uniform noise and uniformly initialised demixing representations brings uniformity to the model, which is why DeGCL and SimGCL outperform other models. SimGCL's strategy of augmenting the data with uniform noise regularisation is particularly effective for wider recommendations, as evidenced by its excellent performance at $K = 50$. Random noise injection produces interventions that diversify the causes of user–item interactions, thereby reducing the impact of spurious correlations.

In a word, our proposed DeGCL achieves the best performance by explicitly mitigating observed and latent confounding effects when building recommender systems. This validates the impact of confounders on recommender systems and the need to mitigate these confounding effects during prediction.

5.3 Ablation studies

To elucidate the contribution of each component in our proposed DeDCL model, we conducted an ablation study by sequentially removing key modules. We first removed the Interaction Enrichment module (IE) from the contrastive learning process. To verify the significance of Spurious interactions on the Deconfounding effect (SD), we stopped training the deconfounding representation through the interaction enrichment module while retaining the contrastive learning. Next, to verify the importance of removing latent confounding

Table 4 Ablation experiments on the MovieLens-1M dataset

Variants	IE	SD	DE	FA	Recall@50	NDCG@50
DeGCL	✔	✔	✔	✔	0.43263	0.34362
DeGCL-1	×	✔	✔	✔	0.41396	0.32686
DeGCL-2	✔	×	✔	✔	0.41537	0.33761
DeGCL-3	✔	✔	×	✔	0.40562	0.31658
DeGCL-4	×	×	×	✔	0.39707	0.32482
DeGCL-5	✔	✔	✔	×	0.37550	0.29986

Table 5 Ablation experiments on the Yelp2018 dataset

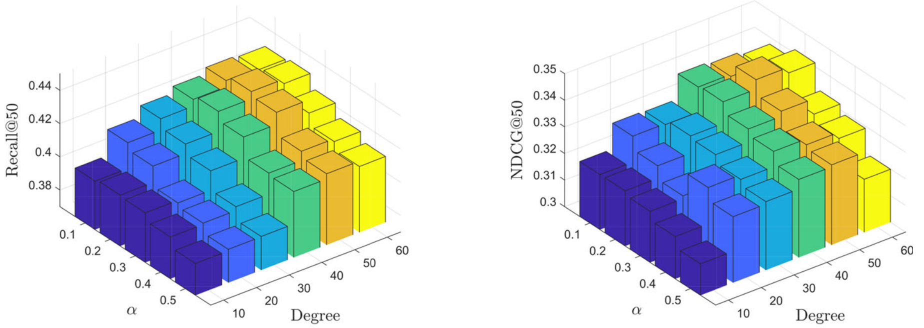
Variants	IE	SD	DE	FA	Recall@50	NDCG@50
DeGCL	✔	✔	✔	✔	0.13655	0.08274
DeGCL-1	×	✔	✔	✔	0.13445	0.08164
DeGCL-2	✔	×	✔	✔	0.13433	0.08149
DeGCL-3	✔	✔	×	✔	0.13415	0.08150
DeGCL-4	×	×	×	✔	0.12542	0.07992
DeGCL-5	✔	✔	✔	×	0.10561	0.06333

effects, we removed the DEconfounding representation module (DE). Finally, we eliminated Feature Augmentation (FA) from contrastive learning. With the four ablation measures, we obtained five variants of the DeGCL model.

As shown in Tables 4 and 5, the removal of interaction enrichment in contrastive learning leads to a slight decrease in model effectiveness, which indicates that data augmentation from structure plays a role in facilitating contrastive learning. Moreover, when spurious interactions are no longer employed for training the deconfounding representation, there is a slight decrease in the model's effectiveness. This suggests that incorporating random interactions as a simulation of real-world latent confounders allows the deconfounding representation to more effectively learn these effects, which is crucial for its functionality.

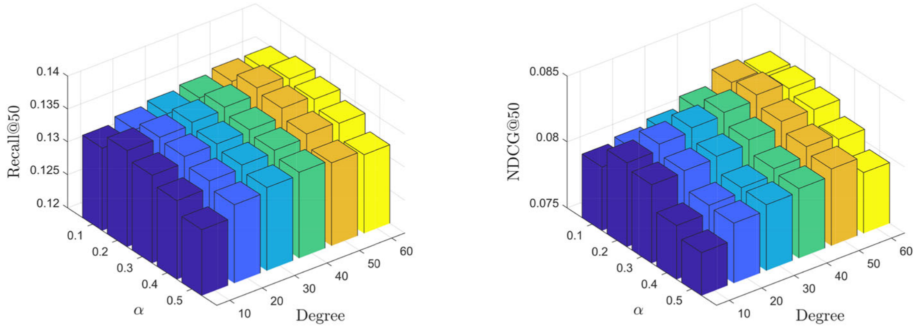
Further analysis revealed that removing the deconfounding representation significantly reduces model effectiveness, highlighting the importance of addressing unobservable latent confounders in complex real-world scenarios. Additionally, when both the interaction enrichment module and the deconfounding representation are removed, the model's effectiveness decreases significantly. Interestingly, for the denser MovieLens-1M dataset, the effect of adding spurious interactions on the training of the deconfounding representation is not as pronounced as in the Yelp2018 dataset, since the denser dataset itself represents a more complex confounding scenario. When the feature augmentation is removed, the model's effectiveness reaches its lowest level despite the deconfounding of the model. When the deconfounding representation is trained together with the node representation, the representations may suffer from the pattern collapse problem, i.e. the generated representations are too similar and lack diversity. This may lead to a decrease in the quality of the representation and the inability to make distinctions during predicting.

Overall, the interaction enrichment module is more effective in modelling latent confounders, thus achieving the goal of deconfounding and helping to reduce spurious correlations associated with the condition of treatment progeny. Additionally, the injection of noise helps to reduce overfitting due to chance. The representations generated by these



(a) Recall@50 (b) NDCG@50

Fig. 4 Parameter sensitivity analysis on the MovieLens-1M dataset



(a) Recall@50 (b) NDCG@50

Fig. 5 Parameter sensitivity analysis on the Yelp2018 dataset

modules successfully avoid the effects of latent confounders. Our ablation studies empirically validate this synergistic effect.

5.4 Parameter sensitivity

We analyse two key hyperparameters controlling DeGCL's deconfounding and graph augmentation. The mixing ratio α modulates the deconfounding loss, governing how representations absorb latent confounders. The degree limit constrains random edge additions to avoid distorting the graph. By randomly adding spurious interactions, it amounts to counterfactually adding the effect of confounders to the interaction data. In this paper, we present hyperparameter sensitivity analyses on the MovieLens-1M dataset and Yelp2018 dataset. The results are visualised in Figs. 4 and 5.

Our experiments indicate that an α value of 0.2 optimises performance by adequate debiasing without overfitting to spurious artefacts. As highlighted in Pearl's work on causal graphs [10], confounders can bias learned representations. DeGCL is designed to eliminate confounding while retaining useful latent factors through its deconfounding objective. Higher values of α tend to make models overly sensitive to augmented interactions, leading to erroneous fitting to artificial confounders. The optimal degree limit of 50 suggests that moderate graph completion achieves a balance between graph densification and preservation of the valid structure.

In summary, α and the degree limit act as adjustable parameters to fine-tune the efficacy of our proposed techniques. When appropriately balanced, DeGCL enhances generalisation by exposing models to a range of counterfactual relationships not present in the original sparse graph. However, improper tuning risks contaminating representations if augmentations stray too far from reality. Therefore, conducting a thorough hyperparameter analysis is essential for applying data augmentation responsibly. Our carefully tuned configurations demonstrate that representations can be effectively enriched with fictional connections and confounders while still maintaining accuracy, provided the augmentation process is controlled meticulously.

6 Conclusions

In this paper, we integrate causal insights into self-supervised graph contrastive learning for mitigating latent confounding effects in recommender systems. Previous methods only focus on observed confounders, while ignoring latent confounders. To tackle this issue, we propose a Deconfounding Graph Contrastive Learning (DeGCL) method. Specifically, we begin by designing a causal graph to depict the recommendation system under the influence of confounders. We then develop a deconfounding representation that captures latent confounding effects and ensures their quality by randomly adding interactions that produce spurious confounding effects. Finally, we mitigate the impact of confounders on model predictions by adjusting for the learned deconfounding representation. Comparative and ablation experiments on two real-world datasets demonstrate the effectiveness of our model and the essential role of each module.

Acknowledgements This work has been supported in part by the National Natural Science Foundation of China under Grant 71774159, the Project of Guangxi Science and Technology GuiKeAB23026040, the China Postdoctoral Science Foundation under Grant 2021T140707, the Jiangsu Postdoctoral Science Foundation under grant number 2021K565C, the Science and Technology Foundation of Xuzhou under Grant KC22047, Australian Research Council under Grant DP230101122, the Graduate Innovation Program of China University of Mining and Technology 2024WLKXJ183, the Fundamental Research Funds for the Central Universities 2024-10949, the Postgraduate Research & Practice Innovation Program of Jiangsu Province KYCX24_2781, the Research Fund of Guangxi Key Lab of Multi-source Information Mining & Security (MIMS24-13), and Guangxi Key Laboratory of Big Data in Finance and Economics (Grant No. FEDOP2022A03).

Author Contributions **Guixian Zhang** contributed to the conceptualisation, methodology, validation, investigation, writing—original draft and editing, and visualisation. **Guan Yuan** was involved in writing—review and editing, supervision, and funding acquisition. **Debo Cheng** assisted in the conceptualisation, writing—review and editing, and formal analysis. **Lin Liu**, **Jiuyong Li** and **Ziqi Xu**contributed to the conceptualisation and writing—review and editing. **Shichao Zhang** was involved in the conceptualisation, writing—review and editing, supervision, and funding acquisition.

Data availability Data will be made available on request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest that could have appeared to influence the work reported in this paper.

References

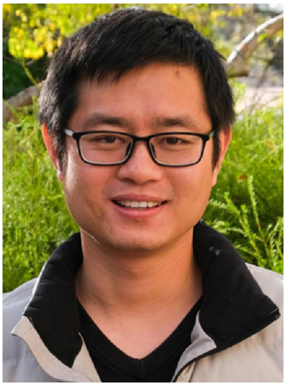
1. Wang L, Zhang X, Wang R, Yan C, Kou H, Qi L (2020) Diversified service recommendation with high accuracy and efficiency. Knowl-Based Syst 204:106196
2. Elahi M, Jannach D, Skjærven L, Knudsen E, Sjøvaag H, Tolonen K, Holmstad Ø, Pipkin I, Throndsen E, Stenbom A et al (2022) Towards responsible media recommendation. AI Ethics 1–12
3. Fu X, Li J, Wu J, Qin J, Sun Q, Ji C, Wang S, Peng H, Yu PS (2023) Adaptive curvature exploration geometric graph neural network. Knowl Inf Syst 65(5):2281–2304
4. Chen X, Fan W, Chen J, Liu H, Liu Z, Zhang Z, Li Q (2023) Fairly adaptive negative sampling for recommendations. In: Proceedings of the ACM web conference 2023, pp 3723–3733
5. Wang Y, Liang D, Charlin L, Blei DM (2020) Causal inference for recommender systems. In: Proceedings of the 14th ACM conference on recommender systems, pp 426–431
6. Zhang S, Li X, Zong M, Zhu X, Cheng D (2017) Learning k for KNN classification. ACM Trans Intell Syst Technol (TIST) 8(3):1–19
7. Yu D, Li Q, Yin H, Xu G (2023) Causality-guided graph learning for session-based recommendation. In: Proceedings of the 32nd ACM international conference on information and knowledge management, pp 3083–3093
8. Cao T, Xu Q, Yang Z, Huang Q (2023) Mitigating confounding bias in practical recommender systems with partially inaccessible exposure status. IEEE Trans Pattern Anal Mach Intell 01:1–18
9. Chen J, Li H, Zhang X, Zhang F, Wang S, Wei K, Ji J (2024) Sr-HetGNN: session-based recommendation with heterogeneous graph neural network. Knowl Inf Syst 66(2):1111–1134
10. Pearl J (2009) Causality. Cambridge University Press, Cambridge
11. Ling Z, Xu E, Zhou P, Du L, Yu K, Wu X (2024) Fair feature selection: a causal perspective. ACM Trans Knowl Discov Data 18(7):1–23
12. Zhang G, Cheng D, Yuan G, Zhang S (2024) Learning fair representations via rebalancing graph structure. Inf Process Manag 61(1):103570
13. Xu S, Tan J, Heinecke S, Li VJ, Zhang Y (2023) Deconfounded causal collaborative filtering. ACM Trans Recomm Syst 1(4):1–25
14. Wang W, Feng F, He X, Wang X, Chua T-S (2021) Deconfounded recommendation for alleviating bias amplification. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, pp 1717–1725
15. Chaney AJ, Stewart BM, Engelhardt BE (2018) How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. In: Proceedings of the 12th ACM conference on recommender systems, pp 224–232
16. Bonner S, Vasile F (2018) Causal embeddings for recommendation. In: Proceedings of the 12th ACM conference on recommender systems, pp 104–112
17. Zhang G, Yuan G, Cheng D, Liu L, Li J, Zhang S (2025) Disentangled contrastive learning for fair graph representations. Neural Netw 181:106781
18. Wang T-Z, Qin T, Zhou Z-H (2023) Estimating possible causal effects with latent variables via adjustment. In: International conference on machine learning. PMLR, pp 36308–36335
19. Cheng D, Li J, Liu L, Xu Z, Zhang W, Liu J, Le TD (2024) Disentangled representation learning for causal inference with instruments. IEEE Transactions on Neural Networks and Learning Systems
20. Krishnan A, Sharma A, Sankar A, Sundaram H (2018) An adversarial approach to improve long-tail performance in neural collaborative filtering. In: Proceedings of the 27th ACM international conference on information and knowledge management, pp 1491–1494
21. Zhang Y, Feng F, He X, Wei T, Song C, Ling G, Zhang Y (2021) Causal intervention for leveraging popularity bias in recommendation. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pp 11–20
22. Zheng Y, Qin J, Wei P, Chen Z, Lin L (2023) CIPL: counterfactual interactive policy learning to eliminate popularity bias for online recommendation. IEEE Trans Neural Netw Learn Syst 35(12):17123–17136
23. Agarwal A, Zaitsev I, Wang X, Li C, Najork M, Joachims T (2019) Estimating position bias without intrusive interventions. In: Proceedings of the twelfth ACM international conference on web search and data mining, pp 474–482
24. Jeunen O (2023) A probabilistic position bias model for short-video recommendation feeds. In: Proceedings of the 17th ACM conference on recommender systems, pp 675–681
25. Wang Y, Xue Y, Liu B, Wen M, Zhao W, Guo S, Yu PS (2023) Click-conversion multi-task model with position bias mitigation for sponsored search in ecommerce. In: Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval, pp 1884–1888
26. Cai R, Wu F, Li Z, Qiao J, Chen W, Hao Y, Gu H (2023) Rest: debiased social recommendation via reconstructing exposure strategies. ACM Trans Knowl Discov Data 18(2):1–24

27. Wang R, Ji W, Liu M, Wang X, Weng J, Deng S, Gao S, Yuan C-A (2018) Review on mining data from multiple data sources. Pattern Recognit Lett 109:120–128
28. Wu J, Wang X, Feng F, He X, Chen L, Lian J, Xie X (2021) Self-supervised graph learning for recommendation. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pp 726–735
29. Huang T, Dong Y, Ding M, Yang Z, Feng W, Wang X, Tang J (2021) MixGCF: an improved training method for graph neural network-based recommender systems. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, pp 665–674
30. Lin Z, Tian C, Hou Y, Zhao WX (2022) Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In: Proceedings of the ACM web conference 2022, pp 2320–2329
31. Yu J, Yin H, Xia X, Chen T, Cui L, Nguyen QVH (2022) Are graph augmentations necessary? Simple graph contrastive learning for recommendation. In: Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval, pp 1294–1303
32. Wang C, Yu Y, Ma W, Zhang M, Chen C, Liu Y, Ma S (2022) Towards representation alignment and uniformity in collaborative filtering. In: Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining, pp 1816–1825
33. Cai X, Huang C, Xia L, Ren X (2023) LightGCL: simple yet effective graph contrastive learning for recommendation. In: Proceedings of the 11th international conference on learning representations
34. Gupta S, Wang H, Lipton Z, Wang Y (2021) Correcting exposure bias for link recommendation. In: International conference on machine learning. PMLR, pp 3953–3963
35. Xie R, Zhang S, Wang R, Xia F, Lin L (2021) Hierarchical reinforcement learning for integrated recommendation. In: Proceedings of the AAAI conference on artificial intelligence, vol 35, pp 4521–4528
36. Liang Y, Yang E, Guo G, Cai W, Jiang L, Wang X (2024) Deconfounding user preference in recommendation systems through implicit and explicit feedback. ACM Trans Knowl Discov Data
37. Chen J, Dong H, Wang X, Feng F, Wang M, He X (2023) Bias and debias in recommender system: a survey and future directions. ACM Trans Inf Syst 41(3):1–39
38. Zhang Q, Xia L, Cai X, Yiu S, Huang C, Jensen CS (2024) Graph augmentation for recommendation. In: 2024 IEEE 40th international conference on data engineering (ICDE). IEEE
39. Zhang S, Li X, Zong M, Zhu X, Wang R (2017) Efficient KNN classification with different numbers of nearest neighbors. IEEE Trans Neural Netw Learn Syst 29(5):1774–1785
40. Deng J, Chen X, Jiang R, Yin D, Yang Y, Song X, Tsang IW (2024) Disentangling structured components: Towards adaptive, interpretable and scalable time series forecasting. IEEE Trans Knowl Data Eng (01):1–18
41. Deng J, Deng J, Yin D, Jiang R, Song X (2023) TTS-Norm: forecasting tensor time series via multi-way normalization. ACM Trans Knowl Discov Data 18(1):1–25
42. Zhang S, Li J, Zhang W, Qin Y (2022) Hyper-class representation of data. Neurocomputing 503:200–218
43. Zhang S (2022) Challenges in KNN classification. IEEE Trans Knowl Data Eng 34(10):4663–4675
44. Yu J, Yin H, Li J, Wang Q, Hung NQV, Zhang X (2021) Self-supervised multi-channel hypergraph convolutional network for social recommendation. In: Proceedings of the web conference 2021, pp 413–424
45. Xia L, Huang C, Zhang C (2022) Self-supervised hypergraph transformer for recommender systems. In: Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining, pp 2100–2109
46. Zhou X, Sun A, Liu Y, Zhang J, Miao C (2023) SelfCF: a simple framework for self-supervised collaborative filtering. ACM Trans Recomm Syst 1(2):1–25
47. Peng L, Wang N, Xu J, Zhu X, Li X (2022) Gate: Graph CCA for temporal self-supervised learning for label-efficient FMRI analysis. IEEE Trans Med Imaging 42(2):391–402
48. Deng J, Chen X, Fan Z, Jiang R, Song X, Tsang IW (2021) The pulse of urban transport: exploring the co-evolving pattern for spatio-temporal forecasting. ACM Trans Knowl Discov Data (TKDD) 15(6):1–25
49. Veličković P, Fedus W, Hamilton WL, Liò P, Bengio Y, Hjelm RD (2019) Deep graph infomax. In: Proceedings of the 7th international conference on learning representations
50. Chen T, Kornblith S, Norouzi M, Hinton G (2020) A simple framework for contrastive learning of visual representations. In: Proceedings of the 37th international conference on machine learning, pp 1597–1607
51. Oord Avd, Li Y, Vinyals O (2018) Representation learning with contrastive predictive coding. Preprint at arXiv:1807.03748
52. Hjelm RD, Fedorov A, Lavoie-Marchildon S, Grewal K, Bachman P, Trischler A, Bengio Y (2019) Learning deep representations by mutual information estimation and maximization. In: Proceedings of the 7th international conference on learning representations

53. Tian Y, Sun C, Poole B, Krishnan D, Schmid C, Isola P (2020) What makes for good views for contrastive learning? In: Proceedings of the 34th international conference on neural information processing systems, pp 6827–6839
54. Xu J, Chen S, Ren Y, Shi X, Shen H, Niu G, Zhu X (2023) Self-weighted contrastive learning among multiple views for mitigating representation degeneration. In: Proceedings of the 37th international conference on neural information processing systems, pp 1119–1131
55. Wang T, Isola P (2020) Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In: Proceedings of the 37th international conference on machine learning, pp 9929–9939
56. Saunshi N, Ash J, Goel S, Misra D, Zhang C, Arora S, Kakade S, Krishnamurthy A (2022) Understanding contrastive learning requires incorporating inductive biases. In: Proceedings of the 39th international conference on machine learning, pp 19250–19286
57. Guo X, Wang Y, Wei Z, Wang Y (2023) Architecture matters: uncovering implicit mechanisms in graph contrastive learning. In: Proceedings of the 37th international conference on neural information processing systems, pp.28585–28610
58. Guan R, Li Z, Tu W, Wang J, Liu Y, Li X, Tang C, Feng R (2024) Contrastive multiview subspace clustering of hyperspectral images based on graph convolutional networks. IEEE Trans Geosci Remote Sens 62:1–14
59. Bao Y, Shen Q, Cao Y, Shi Q (2025) Spatial-temporal clustering enhanced multi-graph convolutional network for traffic flow prediction. Appl Intell 55(6):1–19
60. Zhang G, Zhang S, Yuan G (2024) Bayesian graph local extrema convolution with long-tail strategy for misinformation detection. ACM Trans Knowl Discov Data 18(4):1–21
61. Chen J, Liu H, Hopcroft JE, He K (2024) Leveraging contrastive learning for enhanced node representations in tokenized graph transformers. In: Proceedings of the 38th annual conference on neural information processing systems, vol 37, pp 85824–85845
62. He X, Deng K, Wang X, Li Y, Zhang Y, Wang M (2020) LightGCN: simplifying and powering graph convolution network for recommendation. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, pp 639–648
63. Wang X, He X, Wang M, Feng F, Chua T-S (2019) Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, pp 165–174
64. Liu H, Zhang Y, Li P, Qian C, Zhao P, Wu X (2023) DeepCPR: deep path reasoning using sequence of user-preferred attributes for conversational recommendation. ACM Trans Knowl Discov Data 18(1):1–22
65. Liu H, Zhang H, Li P, Zhao P, Wu X (2024) Denoising implicit feedback for graph collaborative filtering via causal intervention. IEEE Trans Big Data 1–14
66. Zhang X, Yu FX, Kumar S, Chang S-F (2017) Learning spread-out local feature descriptors. In: Proceedings of the IEEE international conference on computer vision, pp 4595–4603
67. Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009) BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, pp 452–461
68. Harper FM, Konstan JA (2015) The MovieLens datasets: history and context. ACM Trans Interact Intell Syst 5(4):1–19

Guixian Zhang is pursuing the Ph.D. degree at the China University of Mining and Technology. His main research interests include data mining, causal inference, and Trustworthy AI.

Guan Yuan received the Ph.D. degree from China University of Mining and Technology, Xuzhou, China, in 2012. He is currently a professor with China University of Mining and Technology. His main research interests include spatio-temporal data mining, social network computing, and machine learning.



Debo Cheng received the Ph.D. degree in computer and information science from the University of South Australia (UniSA), Adelaide, SA, Australia, in 2021. He currently serves as a Research Fellow with UniSA STEM, UniSA. His main interests include data mining, machine learning, causal inference, and causal machine learning.

Jiuyong Li received the Ph.D. degree in computer science from Griffith University, Brisbane, QLD, Australia, in 2002. He is currently a Professor with the University of South Australia (UniSA), Adelaide, SA, Australia. His research work has been supported by eight Australian Research Council Discovery projects and he has led several industrial and applied projects. His main research interests include data mining, causal discovery, privacy and fairness, and bioinformatics.

Lin Liu received the Ph.D. degree in computer systems engineering from the University of South Australia (UniSA), Adelaide, SA, Australia, in 2006. She is currently a Professor with UniSA. Her research interests include data mining, causal inference, and bioinformatics.

Ziqi Xu received the Ph.D. degree in Computer Science from the University of South Australia, Adelaide, Australia. He was previously a CERC Fellow with Data61, Commonwealth Scientific and Industrial Research Organization (CSIRO). He is currently a Lecturer in Data Science and Artificial Intelligence with the School of Computing Technologies, RMIT University, Melbourne, Australia. His research interests include responsible AI, causal inference, fairness, and explainable machine learning

Shichao Zhang received the Ph.D. degree in computer science at the Deakin University, Australia. He is currently a Professor with Guangxi Normal University. His research interests include artificial intelligence, information quality and pattern discovery.

Authors and Affiliations

Guixian Zhang[1,2,3] **· Guan Yuan**[1,3] **· Debo Cheng**[4] **· Lin Liu**[4] **· Jiuyong Li**[4] **·**
Ziqi Xu[5] **· Shichao Zhang**[2]

✉ Guan Yuan
yuanguan@cumt.edu.cn

✉ Debo Cheng
chedy055@mymail.unisa.edu.au

Guixian Zhang
guixian@cumt.edu.cn

Lin Liu
Lin.Liu@unisa.edu.au

Jiuyong Li
Jiuyong.Li@unisa.edu.au

Ziqi Xu
ziqi.xu@rmit.edu.au

Shichao Zhang
zhangsc@mailbox.gxnu.edu.cn

1 School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, Jiangsu, China

2 Guangxi Key Lab of Multi-source Information Mining & Security, Guangxi Normal University, Guilin 541004, Guangxi, China

3 Mine Digitization Engineering Research Center of the Ministry of Education, China University of Mining and Technology, Xuzhou 221116, Jiangsu, China

4 UniSA STEM, University of South Australia, Adelaide, SA 5095, Australia

5 School of Computing Technologies, RMIT University, Melbourne, VIC 3000, Australia