Information Sciences 677 (2024) 120872

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

# Neural Causal Graph collaborative filtering

Xiangmeng Wang [a,1], Qian Li [b,*], Dianer Yu [a], Wei Huang [c], Qing Li [d], Guandong Xu [a,e,**]

[a] *University of Technology Sydney, Sydney, NSW, Australia*
[b] *Curtin University, Perth, WA, Australia*
[c] *RIKEN Center for AIP, Tokyo, Japan*
[d] *Hong Kong Polytechnic University, Hong Kong Special Administrative Region of China*
[e] *Education University of Hong Kong, Hong Kong Special Administrative Region of China*

## ARTICLE INFO

Dataset link: https://github.com/Chrystalii/CNGCF

*Keywords:*
Graph representation learning
Causal inference
Neural causal model
Recommendation system

## ABSTRACT

Graph collaborative filtering (GCF) has emerged as a prominent method in recommendation systems, leveraging the power of graph learning to enhance traditional collaborative filtering (CF). One common approach in GCF involves employing Graph Convolutional Networks (GCN) to learn user and item embeddings and utilize these embeddings to optimize CF models. However, existing GCN-based methods often fall short of generating satisfactory embeddings, mainly due to their limitations in capturing node dependencies and variable dependencies within the graph. Consequently, the learned embeddings are fragile in uncovering the root causes of user preferences, leading to sub-optimal performance of GCF models. In this work, we propose integrating causal modeling with the learning process of GCN-based GCF models, leveraging causality-aware graph embeddings to capture complex dependencies in recommendations. Our methodology encompasses three key designs: 1) Causal Graph conceptualization, 2) Neural Causal Model parameterization, and 3) Variational inference for the Neural Causal Model. We define a Causal Graph to model genuine dependencies in GCF models and utilize this Causal Graph to parameterize a Neural Causal Model. The proposed framework, termed *Neural Causal Graph Collaborative Filtering (NCGCF)*, uses variational inference to approximate neural networks under the Neural Causal Model. As a result, NCGCF is able to leverage the expressive causal effects from the Causal Graph to enhance graph representation learning. Extensive experimentation on four datasets demonstrates NCGCF's ability to deliver precise recommendations consistent with user preferences.

## 1. Introduction

Collaborative Filtering (CF) [1] as an effective remedy has dominated recommendation research for years. Recently, Graph Collaborative Filtering (GCF) has been studied extensively and has become an emerging CF paradigm [2]. GCF enhances traditional

[*] Corresponding author.
[**] Corresponding author at: University of Technology Sydney, Sydney, NSW, Australia.
*E-mail addresses:* xiangmeng.wang@student.uts.edu.au (X. Wang), qli@curtin.edu.au (Q. Li), Dianer.Yu-1@student.uts.edu.au (D. Yu), Wei.Huang.vr@riken.jp (W. Huang), qing-prof.li@polyu.edu.hk (Q. Li), guandong.xu@uts.edu.au (G. Xu).
[1] Xiangmeng Wang and Qian Li equal contribution.

https://doi.org/10.1016/j.ins.2024.120872
Received 22 February 2024; Received in revised form 1 June 2024; Accepted 2 June 2024

Available online 5 June 2024

CF methods by modeling complex user-item interactions in a graph as well as auxiliary information, e.g., user and item attributes. Thus, GCF has shown great potential in deriving knowledge (e.g., user behavior patterns) embedded in graphs.

Generally, GCF models utilize graph representation learning techniques to derive useful information for downstream CFs. These models use graph neural networks to analyze graph connections and create embeddings, thereby improving CF model optimization. Graph Convolutional Networks (GCN) is one of the most widely adopted graph neural networks in GCF models, primarily due to its well-established performance in learning local and global information from large-scale graphs. As evidenced by several studies [3–6], GCN advances from other graph neural networks in capturing graph structural information. This is because GCN leverages convolutional operations to aggregate both local information from direct neighboring nodes and global information from higher-hop neighbors. Besides, the parameter sharing of GCN allows it to generalize well to unseen nodes during training and efficiently adapt to large-scale graphs. GCN-based GCF models aim to acquire vectorized user and item embeddings using a GCN and then use these embeddings to optimize a CF model. For instance, NGCF [3] exploits a GCN to propagate neighboring node messages in the interaction graph to obtain user and item embeddings. The learned embeddings capture user collaborative behavior and are used to predict preference scores for CF optimization. HGCF [4] combines GCN with hyperbolic learning to learn embeddings in hyperbolic space. Benefiting from the exponential neighborhood growth in hyperbolic space, HGCF facilitates learning higher-order user and item relations from the interaction graph.

Though promising results are witnessed, leveraging graph learning techniques in GCF models presents severe challenges. Compared to traditional methods, such as matrix factorization on pure user-item interactions, graph learning seeks to model complex, multi-hop relationships propagated across graph entities. The complexity of relation modeling increases as the graph becomes more sophisticated by including various types of nodes, e.g., user gender [7], item brands [8]. For example, consider a social network where users are connected based on their friendships, following lists and followers [9]. Graph learning techniques must not only capture these diverse relationships but also consider other potential factors, such as user conformity impacts [10] on social content recommendations.

However, existing GCN-based GCF methods often fall short of capturing complex graph relationships due to two fundamental drawbacks. Firstly, *they ignore distinguishable node dependencies between neighboring nodes and the target node.* Most GCN-based GCF methods treat all messages from the neighborhood equally, which inevitably overlooks the varying dependencies of neighboring nodes to the target node. However, a user node might have different relations with other neighboring nodes, e.g., item brands. These different relations reflect distinct user preferences, which is the essence of personalized recommendations [11]. By ignoring the difference in relations, the learned user and item embeddings eventually lose expressive power in the recommendation task, i.e., we cannot know which node is the root cause of user interests. Secondly, *they lack an explicit encoding of complex relations between variables in the recommendation.* Most GCN-based GCF methods assume the co-occurrence of users and items is independent. However, user preferences are influenced by various variables in real-world recommendations, such as user conformity [12] caused by user social networks. Discarding these relations leads to the learned embeddings being unable to capture such structural complexity.

Causal modeling sheds light on solving the above drawbacks. On the one hand, causal modeling identifies intrinsic cause-effect relations between nodes and true user preferences [13]. For example, we might treat each neighboring node as the cause (e.g., an item brand) and the user preference as the effect in a Causal Graph [14]. By estimating the causal effect, we could encode the crucial node dependencies into user and item embeddings to uncover the root causes of user interests. On the other hand, the Causal Graph is able to model genuine causal relations among the variables in GCFs, capturing variable dependencies inherent in the GCF-based methods. Those causal relations represent the underlying mechanisms driving the recommendation and can be utilized to guide graph learning toward complex user behaviors.

Given the compelling nature of casual modeling in GCN-based GCFs, this paper aims to integrate GCNs and causal models to facilitate causality-aware GCF learning. Motivated by Neural-Causal Connection [15], this paper proposes to connect GCN learning with the Structural Causal Model (SCM) [16]. Since the SCM is induced from a Causal Graph and the GCN works on graph-structured data, the integration of the two models becomes practical. In particular, we first conceptualize the Causal Graph for the SCM, which is built by revisiting existing CFs and padding their limitations in user preference modeling. Then, we formulate the SCM into a Neural Causal Model, called *Neural Causal Graph Collaborative Filtering (NCGCF)*. Our NCGCF uses variational inference to approximate structural equations as trainable neural networks, making the learned graph embeddings equally expressive as the causal effects modeled by the SCM. The integration of causal modeling and graph representation learning offers a novel perspective to facilitate accurate recommendations. The contributions of this work are:

- We complete the Neural-Causal Connection for causal modeling of graph convolutional network in recommendations.
- Our proposed NCGCF is the first Neural Causal Model for graph collaborative filtering, which generates causality-aware graph embeddings for enhanced recommendations.
- We validate the effectiveness of our proposed framework through extensive experiments. Our experimental results demonstrate that our approach outperforms existing methods in achieving satisfactory recommendation performance.

## 2. Notations and formulation

We first provide the notations used throughout the paper and our motivations for defining our Causal Graph. We then give our task formulation, which covers detailed steps toward connecting GCN with the Structural Causal Model.

**Notations.** We use uppercase letters such as $U$ to denote a set of variables. In particular, we use $U, V, E, Y$ to represent user, item, preference representation and recommendation variables. We use lowercase letters such as $u$ to represent a random variable. In

**Table 1**
Key notations and descriptions.

| Notation | Description |
|---|---|
| $G$ | A Causal Graph |
| $\mathcal{M}$ | A Structural Causal Model |
| $\mathcal{M}(\theta)$ | A Neural Causal Model |
| $\mathcal{V} = \{U, V, E, Y\}$ | Endogenous variables in $G$ |
| $\mathcal{F} = \{f_U, f_V, f_E, f_Y\}$ | Structural equations for $G$ |
| $U$, $f_U$ | User variable and its structural equation |
| $V$, $f_V$ | Item variable and its structural equation |
| $E$, $f_E$ | Preference representation variable and its structural equation |
| $Y$, $f_Y$ | Recommendation result variable and its structural equation |
| $Z$ | Exogenous variables in $G$ |
| $\mathbf{Z}_u$, $\mathbf{Z}_v$ | Latent vectors of exogenous variables for a user variable $u$ and an item variable $v$ |
| $\mathbf{A}_u$, $\mathbf{A}_v$ | Causal adjacency vector for a user variable $u$ and an item variable $v$ |
| $\mathbf{d}_u$, $\mathbf{d}_v$ | Feature vectors for a user variable $u$ and an item variable $v$ |
| $\mathbf{u}$, $\mathbf{v}$ | Latent factors for a user $u$ and an item $v$ |
| $\mathbf{e}$, $\mathbf{y}$ | A user preference vector and a user interaction vector |
| $\mathbf{h}_u$, $\mathbf{h}_v$ | Hidden factors for a user $u$ and an item $v$ from the semi-implicit generative model |
| $\mathbf{m}_{uv}$ | Neighbor message from a node $v$ for a user $u$ |
| $\theta_1$, $\theta_2$, $\theta_3$ | Network parameters for the user encoder, the item encoder and the collaborative filtering decoder |
| $\phi_1$, $\phi_2$ | Network parameters for the aggregation operators |
| $\varphi_1$, $\varphi_2$ | Network parameters for the causality-aware message passing operators |
| $l$ | A graph learning layer |
| $do(i = x)$ | The do-operator that forces a variable $i$ to take the value $x$ |

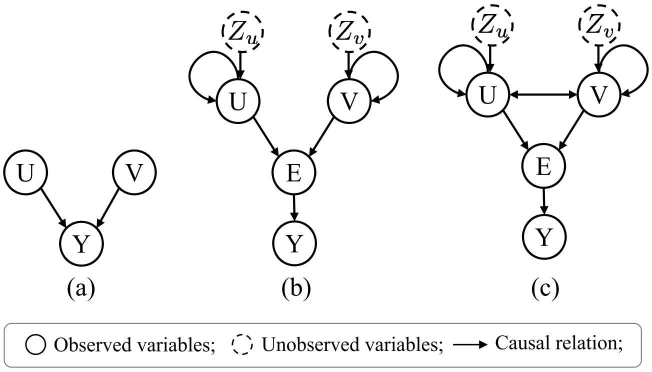

Z_u
Z_v
Z_u
Z_v
U
V
U
V
U
V
E
E
Y
Y
Y
(a)
(b)
(c)
○ Observed variables; Unobserved variables; Causal relation;

**Fig. 1.** Paradigms of user preference modeling in a class of CFs: (a) Early CF, (b) GCF, and (c) Our causality-aware GCF. $Z_u$ represents hidden exogenous variables for users, e.g., user conformity; $Z_v$ are hidden exogenous variables for items, e.g., item exposure. $U$ and $V$ denote user and item, respectively. $E$ denotes preference representations from graph representation learning. $Y$ represents users' predicted recommendations.

particular, we use $u, v, e, y$ to represent a specific user, item, preference and recommendation variable. Moreover, we use bold font lowercase to represent the latent vector embeddings, such as $\mathbf{u}, \mathbf{v}, \mathbf{e}, \mathbf{y} \in \mathbb{R}^d$, where $d$ is the dimension of the embedding vectors. The weight matrix and bias vector are denoted as $\mathbf{W}$ and $\mathbf{b}$, respectively. Primary notations are also complemented in Table 1.

### 2.1. Motivation

**Definition 2.1** *(Causal Graph).* A *Causal Graph* [14] is a directed acyclic graph (DAG) $G = (\{\mathcal{V}, Z\}, \mathcal{E})$ represents causal relations among endogenous and exogenous variables. $\mathcal{V}$ is a set of endogenous variables of interest, e.g., user and item nodes in graph learning. $Z$ is a set of exogenous variables outside the model, e.g., item exposure. $\mathcal{E}$ is edge set denoting causal relations among $G$.

Following Definition 2.1, we start by providing the causal graphs of a class of CF methods, including early CF methods in Fig. 1 (a) and existing GCF methods in Fig. 1 (b). Specifically, we aim to show the fundamental drawback shared by these two types of methods: they are fragile in capturing complex user-item relations by assuming the co-occurrence of users and items is independent. We put forward our defined causal graph in Fig. 1 (c), which considers user-item dependencies for better user preference modeling.

Early CFs largely resort to user-item associative matching [17] and follow the causal graph shown in Fig. 1 (a), where user node $U$ and item node $V$ constitute a collider to affect the recommendation result $Y$. For example, matrix factorization typically assumes $P(Y = 1 \mid u, v) \propto \mathbf{u}^\top \mathbf{v}$, where $\mathbf{u}$ and $\mathbf{v}$ are user and item IDs and the probability of recommendations $Y$ is estimated from matching the inner product between $\mathbf{u}$ and $\mathbf{v}$. Latent factor-based methods assume $P(Y = 1 \mid u, v) \propto \mathrm{LFM}(u)^\top \mathrm{LFM}(v)$, where LFM is a latent factor model that learns the user and item latent vectors, and a simple inner product is used for similarity matching to determine recommendations.

As shown in Fig. 1 (b), GCF works on graph-structure data to consider auxiliary information, e.g., user/item attributes, which potentially captures exogenous variables $Z_u$ and $Z_v$, e.g., user conformity, item exposure. Besides, as user-user and item-item relations are propagated through multi-hop neighbors within the graph, GCF can capture the inner connections of users and items to model more complex user behavior patterns, e.g., user collaborative behavior [3]. However, existing GCF methods still assume the independence between users and items. This is because user and item embeddings are learned separately from the graph representation learning and then subsequently applied to a CF model for user-item associative matching. For example, NGCF [3] assumes $P(Y=1 \mid u, v) \propto E = \mathrm{CF}\left(\mathrm{agg}\left(u, z_u, \mathrm{msg}\left(N_u\right)\right), \mathrm{agg}\left(v, z_v, \mathrm{msg}\left(N_v\right)\right)\right)$, where CF is a CF model for user-item associative matching. $N_u$ and $N_v$ are neighbor sets for users and items; agg and msg are the aggregation and message passing operations, respectively.

In summary, both Fig. 1 (a) and (b) assume the co-occurrence of users and items is independent in the observational data, i.e., there is no edge $U \to V$ or $V \to U$. However, this assumption is unrealistic in the real world because user behaviors are influenced by the recommended items for various reasons. For instance, users may be more likely to click the items if they are recommended [18], which is also known as the item exposure bias problem. Besides, the exposure of items is determined by user preferences estimated from the recommendation model, which is the essence of the personalized recommendation. Therefore, we conceptualize the causal relations under GCN-based GCF as the Causal Graph in Fig. 1 (c). Our Causal Graph includes the modeling of $U \longleftrightarrow V$, such that user-item relations can be captured for better user preference modeling. By given the Causal Graph in Fig. 1 (c), the directed edge $(u \to v) \in \mathcal{E}$ captures the causal relation from a user $u$ to an item $v$, where $u \in U$ and $v \in V$ and $u$ is a parent node of $v$, i.e., $u \in pa(v)$. $G$ induces a set of causal adjacency vectors $\mathbf{A}_u$ and $\mathbf{A}_v$, which specify the neighbors of a user node $u$ and an item node $v$, respectively. Each element $\mathbf{A}_u^v = 1$ if $v \in pa(u)$, otherwise, $\mathbf{A}_u^v = 0$. Similarly, $\mathbf{A}_v^u = 1$ if $u \in pa(v)$.

### 2.2. Task formulation

The key innovation of this work is to integrate causal modeling into the learning process of a GCN-based GCF model. The problem can be formulated as follows:

**Definition 2.2** *(Task formulation).* Establish the connection between the GCN-based GCF model and the Causal Graph depicted in Fig. 1 (c). Motivated by Neural-Causal Connection [15], the goal is to approximate a Neural Causal Model (NCM) based on the provided Causal Graph.

To achieve this goal, we first convert the Causal Graph into a Structural Causal Model (SCM) (**Section 3.1**). Subsequently, the NCM is defined based on the SCM, with each structural equation in the SCM corresponding to a neural network in the NCM (**Section 3.2**). To approximate the trainable neural networks in the NCM, we employ a unified framework described in **Section 4**. This framework enables causal modeling, making the learned graph embeddings as expressive as the causal effects modeled by the SCM. Overall, through the integration with causal modeling, our approach offers a novel perspective on graph representation learning, leveraging the expressive power of the causality-aware graph embeddings to capture complex causal relations in the recommendation.

## 3. Neural Causal Model

This section evokes the concept of the Structural Causal Model (SCM) and the Neural Causal Model (NCM). The SCM converts causal relations among the Causal Graph in Fig. 1 (c) as structural equations; The NCM defines each of the structural equations as a parameterized neural network.

### 3.1. Structural Causal Model

The Causal Graph in Fig. 1 (c) has four variables of interest (i.e., endogenous variables): $U$ (user), $V$ (item), $E$ (preference representation) and $Y$ (recommendation). Besides, two exogenous variables $Z_u$ and $Z_v$ are also involved, representing hidden impacts such as user conformity and item exposure. The causal mechanism of modeling the four endogenous variables $\{U, V, E, Y\}$ is done by a SCM [16].

**Definition 3.1** *(Structural Causal Model).* A *Structural Causal Model* [16] $\mathcal{M} = \langle \mathcal{V}, Z, \mathcal{F}, P(Z) \rangle$ is the mathematical form of the Causal Graph $G$ that includes a collection of structural equations $\mathcal{F}$ on endogenous variables $\mathcal{V}$ and a distribution $P(Z)$ over exogenous variables $Z$. A structural equation $f_U \in \mathcal{F}$ for a variable $u \in U \subseteq \mathcal{V}$ is a mapping from $u$'s parents and exogenous variables of $u$:

$$u \leftarrow f_U\left(pa(u), Z_u\right), Z_u \sim P(Z) \tag{1}$$

where $pa(u) \subseteq \mathcal{V} \backslash u$ is $u$'s parents from the Causal Graph $G$. $Z_u \in Z$ is a set of exogenous variables connected with $u$.

Following Definition 3.1 and the causal relations in Fig. 1 (c), endogenous variables $\{U, V, E, Y\} = \mathcal{V}$ are modeled by structural equations $\{f_U, f_V, f_E, f_Y\} = \mathcal{F}$. Formally,

$$\mathcal{F}(\mathcal{V}, Z) := \begin{cases} U \leftarrow f_U\left(U, V, Z_u\right) \\ V \leftarrow f_V\left(U, V, Z_v\right) \\ E \leftarrow f_E(U, V) \\ Y \leftarrow f_Y(E) \end{cases} \tag{2}$$

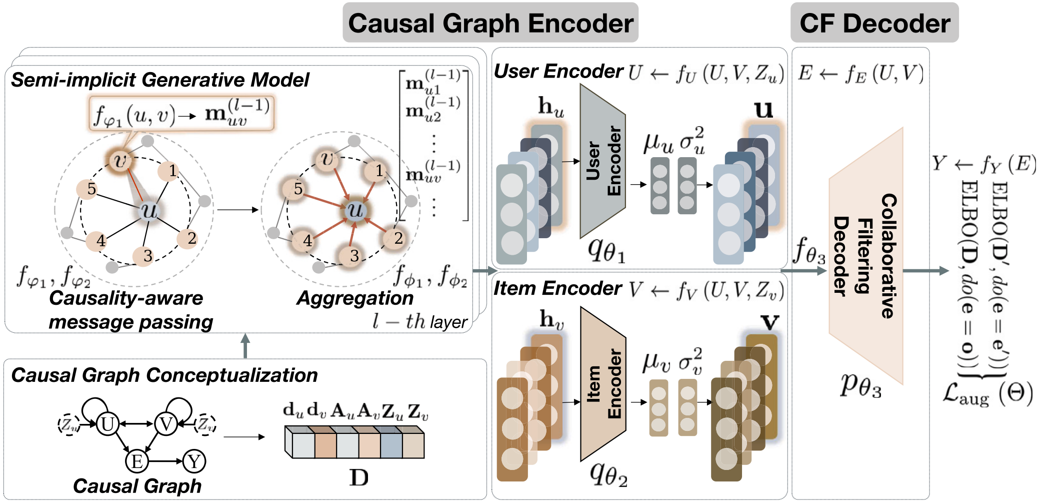

Causal Graph Encoder
CF Decoder
Semi-implicit Generative Model
$f_{\varphi_1}(u,v) \rightarrow \mathbf{m}_{uv}^{(l-1)}$
$\mathbf{m}_{u1}^{(l-1)}$
$\mathbf{m}_{u2}^{(l-1)}$
$\mathbf{m}_{uv}^{(l-1)}$
$f_{\varphi_1}, f_{\varphi_2}$
Causality-aware message passing
$f_{\phi_1}, f_{\phi_2}$
Aggregation
$l-th$ layer
User Encoder $U \leftarrow f_U(U,V,Z_u)$
$\mathbf{h}_u$
User Encoder
$\mu_u\ \sigma_u^2$
$\mathbf{u}$
$q_{\theta_1}$
Item Encoder $V \leftarrow f_V(U,V,Z_v)$
$\mathbf{h}_v$
Item Encoder
$\mu_v\ \sigma_v^2$
$\mathbf{v}$
$q_{\theta_2}$
$E \leftarrow f_E(U,V)$
$f_{\theta_3}$
Collaborative Filtering Decoder
$p_{\theta_3}$
$Y \leftarrow f_Y(E)$
$\mathrm{ELBO}(\mathbf{D}', do(\mathbf{e}=\mathbf{e}'))$
$\mathrm{ELBO}(\mathbf{D}, do(\mathbf{e}=\mathbf{o}))$
$\mathcal{L}_{\text{aug}}(\Theta)$
Causal Graph Conceptualization
$Z_u$ U V $Z_v$ E Y
Causal Graph
$\mathbf{d}_u\,\mathbf{d}_v\,\mathbf{A}_u\,\mathbf{A}_v\,\mathbf{Z}_u\,\mathbf{Z}_v$
$\mathbf{D}$

**Fig. 2.** NCGCF framework: causal graph conceptualization prepossesses a user-item interaction graph by using the causal relations under our defined Causal Graph; causal graph encoder models the causal relations under the graph-structured data using a semi-implicit generative model, and outputs user and item representations with a user encoder and an item encoder; collaborative filtering (CF) decoder uses CF to construct preference vectors based on user and item representations. Finally, NCGCF is optimized through a counterfactual instance-aware ELBO to capture user preference shifts.

These structural equations model the direct causal relation from a set of causes (e.g., $pa(u)$) to a variable (e.g., $u \in U$) accounting for the impact of exogenous variables (e.g., $Z_u$).

### 3.2. Neural network for Causal Model

We now formally introduce Neural-Causal Connection [15], i.e., the connection between deep neural networks (e.g., GCNs) and causal models is done by establishing an NCM.

**Definition 3.2** *(Neural-Causal Connection)*. A *Neural Causal Model* [15] is defined as $\mathcal{M}(\theta)$ and is parameterized for the SCM $\mathcal{M}$ in Definition 3.1. Each structural equation in $\mathcal{M}$ is defined as a feedforward neural network in $\mathcal{M}(\theta)$, e.g., Multi-layer perceptron (MLP). Exogenous variables $Z$ are mapped into hidden vectors $\mathbf{Z}$ that follow the Gaussian distribution $\mathcal{N}\left(0, \mathbf{I}_K\right)$.

The NCM $\mathcal{M}(\theta)$ is expressive [15], such that it generates distributions associated with the Pearl Causal Hierarchy (PCH) [19], i.e., modeling "observational" (layer 1), "interventional" (layer 2) and "counterfactual" (layer 3) distributions.

In accordance with Definition 3.2, we aim to build an NCM $\mathcal{M}(\theta)$ that models structural equations defined in Eq. (2) as parameterized feedforward neural networks. Formally,

$$\mathcal{M}(\theta) \triangleq \begin{cases} \mathbf{Z}_u \sim \mathcal{N}\left(0, \mathbf{I}_K\right), \mathbf{Z}_v \sim \mathcal{N}\left(0, \mathbf{I}_K\right), \\ \mathbf{u} \propto f_U = q_{\theta_1}(f_{\phi_1}\left(\mathbf{Z}_u, f_{\varphi_1}(U \mid U, V)\right)), \\ \mathbf{v} \propto f_V = q_{\theta_2}(f_{\phi_2}\left(\mathbf{Z}_v, f_{\varphi_2}(V \mid U, V)\right)), \\ \mathbf{e} \propto f_E = p_{\theta_3}(\mathbf{u}, \mathbf{v}), \\ \mathbf{y} \sim f_Y = \text{Multinomial}(N, \mathbf{e}) \end{cases} \tag{3}$$

- $Z_u$, $Z_v$ are mapped into low-dimensional hidden vectors $\mathbf{Z}_u$ and $\mathbf{Z}_v$ using Gaussian distribution $\mathcal{N}\left(0, \mathbf{I}_K\right)$.
- $\mathbf{u} \propto f_U$: user representation $\mathbf{u}$ is calculated by a user encoder $q_{\theta_1}$. The user encoder takes as input the aggregated (i.e., $f_{\phi_1}$) information of user exogenous variables $\mathbf{Z}_u$ and user's causality-aware neighbor messages $f_{\varphi_1}$.
- $\mathbf{v} \propto f_V$: item representation $\mathbf{v}$ is given by an item encoder $q_{\theta_2}$. The item encoder uses aggregated (i.e., $f_{\phi_2}$) information of item exogenous variables $\mathbf{Z}_v$ and item's causality-aware neighbor messages $f_{\varphi_2}$.
- $\mathbf{e} \propto f_E$: user preference probability $\mathbf{e}$ is produced by a collaborative filtering decoder $p_{\theta_3}$ by using latent representations $\mathbf{u}$ and $\mathbf{v}$.
- $\mathbf{y} \sim f_Y$: user interaction $\mathbf{y}$ is sampled from a multinomial distribution with the probability $\mathbf{e}$. $N$ is the user's total interaction number.

## 4. Variational inference for NCGCF

We now introduce our framework, namely, *Neural Causal Graph Collaborative Filtering (NCGCF)*. We show the NCGCF framework in Fig. 2, which includes three major components based on the variational autoencoder structure:

- **Causal Graph Encoder**: approximates $f_U$ and $f_V$. The causal graph encoder includes a user encoder, an item encoder and a semi-implicit generative model. The semi-implicit generative model calculates causal relations between nodes as causality-aware messages. The user encoder and item encoder then use these causality-aware messages to output user representation and item representation, respectively.
- **Collaborative Filtering (CF) Decoder**: approximates $f_E$ using a CF method to estimate user preference.
- **Counterfactual Instances-based Optimization**: optimizes model parameters by implementing $f_Y$ with counterfactual instances to capture user preference shifts.

### *4.1. Causal graph encoder*

The causal graph encoder aims to model $f_U$ and $f_V$ in Eq. (3). However, this is not a trivial task as the true posteriors of $f_U$ and $f_V$ do not follow standard Gaussian distributions due to the existence of causal relations between node pairs. Besides, these causal relations should be modeled into causality-aware messages using neural networks. Thus, traditional variational inference [20] that approximates posteriors to simple, tractable Gaussian vectors is not applicable.

Semi-implicit variational inference (SIVI) [21] that models complex distributions through implicit posteriors proves to be an effective alternative. Inspired by SIVI, we devise a semi-implicit generative model on top of the user and item encoder to model implicit posteriors. In particular, the semi-implicit generative model calculates causal relations between nodes as causality-aware messages. Those causality-aware messages are encoded into user and item hidden factors $\mathbf{h}_u$ and $\mathbf{h}_v$. Then, the user encoder takes $\mathbf{h}_u$ as the input to output the user representation $\mathbf{u}$. Analogously, the item encoder uses $\mathbf{h}_v$ to calculate item representation.

#### *4.1.1. Semi-implicit generative model*

Our semi-implicit generative model contains two operators: *causality-aware message passing* and *aggregation*. The *causality-aware message passing* uses learnable neural networks to model each of the dependency terms for a node and its neighbors within a structural equation. For example, $f_{\varphi_1}(u,v)$ models the dependency between a user node $u$ and his/her neighbor item node $v$, such that the learned message becomes a descriptor of the causal relation $u \to v$. The *aggregation* uses weighted-sum aggregators to aggregate user/item exogenous variables and the calculated causality-aware neighbor messages. Finally, user and item hidden factors $\mathbf{h}_u$ and $\mathbf{h}_v$ are output for latter user and item encoder learning.

- *Causality-aware message passing*: For the user encoder, given user $u$'s features $\mathbf{d}_u$ and its causal adjacency vector $\mathbf{A}_u$, the messages from $u$'s neighbor $v$ is given by:

$$\begin{aligned}\mathbf{m}_{uv}^{(l-1)} &= f_{\varphi_1}(u,v) = \mathrm{MLP}^{(l)}\left(\mathbf{h}_u^{(l-1)} \| \mathbf{h}_v^{(l-1)}\right)\\ &= \mathrm{ReLU}\left(\mathbf{W}_{\varphi_1}^{(l)}\left(\mathbf{h}_u^{(l-1)} \| \mathbf{h}_v^{(l-1)}\right)\right), \text{ for } l \in \{1, \cdots, L\}\end{aligned} \tag{4}$$

  where $\mathbf{m}_{uv}^{(l-1)}$ is the neighbor message at the $l-1$-th graph learning layer.[2] $v$ is a neighbor for $u$ and $v \in N_u \propto \mathbf{A}_u$. $\mathbf{h}_v^{(l-1)}$ and $\mathbf{h}_u^{(l-1)}$ are hidden factors for the neighbor $v$ and the user $u$ at the $l-1$-th layer.[3] $\mathbf{W}_{\varphi_1}^{(l)}$ is the weight matrix for $f_{\varphi_1}$ at the $l$-th layer and $\|$ denotes column-wise concatenation. Analogously, for the item encoder, we can calculate the neighbor message $\mathbf{m}_{vu}^{(l-1)}$ for an item $v$ by replacing $f_{\varphi_1}$ with $f_{\varphi_2}$ in Eq. (4).

- *Aggregation*: For the user encoder, at each graph learning layer $l$, we perform aggregation operation on the messages $\mathbf{m}_{uv}^{(l-1)}$ from $u$'s neighbors and the user exogenous variables $\mathbf{Z}_u$ to obtain the hidden factor $\mathbf{h}_u^{(l)}$:

$$\mathbf{h}_u^{(l)} = \left(\mathbf{h}_u^{(l-1)} \| f_{\phi_1}\left(\left\{\mathbf{W}_{\phi_1}^{(l)} \mathbf{m}_{uv}^{(l-1)} : v \in N_u\right\}\right) \| \mathbf{Z}_u\right) \tag{5}$$

  where $\mathbf{h}_u^{(l)}$ is the learned hidden factor for $u$ at the $l$-th graph learning layer. $f_{\phi_1}$ is the aggregation operator chosen as weighted-sum, following [22]. $\mathbf{W}_{\phi_1}^{(l)}$ is the weight for $f_{\phi_1}$ that specifies the different contributions of neighbor messages to the target node at the $l$-th layer. $\|$ is the column-wise concatenation. $\mathbf{Z}_u$ is low-dimensional latent factors for user exogenous variables given by Gaussian distribution $\mathcal{N}\left(0, \mathbf{I}_K\right)$. Similarly, for the item encoder, we calculate item $v$'s hidden factors $\mathbf{h}_v^{(l)}$ by using $f_{\phi_2}$ with $\mathbf{W}_{\phi_2}$ in Eq. (5).

[2] The neighbor message at the 0-th layer, i.e., $\mathbf{m}_{uv}^{(0)}$, is initialized from a normal distribution.

[3] $\mathbf{h}_v^{(0)}$ and $\mathbf{h}_u^{(0)}$ are initialized as node features $\mathbf{d}_v$ and $\mathbf{d}_u$.

Having obtained the hidden factors $\mathbf{h}_u^{(l)}$ for user $u$ and $\mathbf{h}_v^{(l)}$ for item $v$ at each graph learning layer $l \in \{1, \cdots, L\}$, we adopt layer-aggregation to concatenate vectors at all layers into a single vector:

$$\mathbf{h}_u = \mathbf{h}_u^{(1)} + \cdots + \mathbf{h}_u^{(L)}, \quad \mathbf{h}_v = \mathbf{h}_v^{(1)} + \cdots + \mathbf{h}_v^{(L)} \tag{6}$$

By performing layer aggregation, we capture higher-order connectivities of node pairs across different graph learning layers. Finally, our semi-implicit generative model outputs $\mathbf{h}_u$ and $\mathbf{h}_v$ as hidden factors of users and items.

#### 4.1.2. User and item encoder

Given hidden factors $\mathbf{h}_u$ for a user $u$, the user encoder outputs mean and variance in $\mathcal{N}\left(\mu_u, \mathrm{diag}\left(\sigma_u^2\right)\right)$, from which user embedding $\mathbf{u}$ is sampled:

$$q_{\theta_1}\left(\mathbf{u} \mid \mathbf{h}_u\right) = \mathcal{N}\left(\mathbf{u} \mid \mu_u, \mathrm{diag}\left(\sigma_u^2\right)\right) \tag{7}$$

where $\mu_u$ and $\mathrm{diag}\left(\sigma_u^2\right)$ are the mean and variance for user $u$, which are obtained by sending $u$'s hidden factors $\mathbf{h}_u$ to a one-layer neural network with the activation function $\mathrm{ReLU}(x) = \max(0, x)$:

$$\mu_u = \mathrm{ReLU}\left(\mathbf{W}_{\theta_1}^{\mu_u}\mathbf{h}_u + \mathbf{b}\right), \sigma_u^2 = \exp\left(\mathrm{ReLU}\left(\mathbf{W}_{\theta_1}^{\sigma_u}\mathbf{h}_u + \mathbf{b}\right)\right) \tag{8}$$

where $\mathbf{W}_{\theta_1} = \{\mathbf{W}_{\theta_1}^{\mu_u}, \mathbf{W}_{\theta_1}^{\sigma_u}\}$ is a hidden-to-output weight matrix for the user encoder $q_{\theta_1}$; $\mathbf{b}$ is the bias vector. Analogously, the item encoder follows the same paradigm as the user encoder to generate the mean and variance for item $v$ based on $v$'s hidden factors $\mathbf{h}_v$:

$$\begin{gathered} q_{\theta_2}\left(\mathbf{v} \mid \mathbf{h}_v\right) = \mathcal{N}\left(\mathbf{v} \mid \mu_v, \mathrm{diag}\left(\sigma_v^2\right)\right), \\ \mu_v = \mathrm{ReLU}\left(\mathbf{W}_{\theta_2}^{\mu_v}\mathbf{h}_v + \mathbf{b}\right), \sigma_v^2 = \exp\left(\mathrm{ReLU}\left(\mathbf{W}_{\theta_2}^{\mu_v}\mathbf{h}_v + \mathbf{b}\right)\right) \end{gathered} \tag{9}$$

where $\mathbf{W}_{\theta_2} = \{\mathbf{W}_{\theta_2}^{\mu_v}, \mathbf{W}_{\theta_2}^{\sigma_v}\}$ is the weight matrix for the item encoder $q_{\theta_2}$.

### 4.2. Collaborative filtering decoder

Collaborative filtering is largely developed based on latent factors. These models involve mapping users and items into latent factors in order to estimate the preference scores of users towards items. We use latent factor-based collaborative filtering in our decoder for modeling the user preference $\mathbf{e}$, which is a probability vector over the entire item set for recommendations. The predicted user interaction vector $\mathbf{y}$ is assumed to be sampled from a multinomial distribution with probability $\mathbf{e}$.

Formally, we define a generative function $f_{\theta_3}(\mathbf{u}, \mathbf{v})$ recovering classical latent factor-based CF to approximate user preference vector $\mathbf{e}$:

$$\mathbf{e} = softmax(f_{\theta_3}(\mathbf{u}, \mathbf{v})) = softmax(\mathbf{u}^\top \mathbf{v}) \tag{10}$$

where $\mathbf{u}$ and $\mathbf{v}$ are latent factors for a user $v$ and an item $v$ drawn from Eq. (7) and Eq. (9), respectively. The $softmax$ function transforms the calculated preference scores to probability vector $\mathbf{e}$ over the item corpus.

Then, the decoder $p_{\theta_3}\left(\mathbf{e} \mid \mathbf{u}, \mathbf{v}\right)$ produces interaction probability $\mathbf{y}$ by approximating a logistic log-likelihood:

$$\log p_{\theta_3}\left(\mathbf{y} \mid \mathbf{e}\right) = \sum_v y_{uv} \log \sigma\left(\mathbf{e}\right) + \left(1 - y_{uv}\right) \log\left(1 - \sigma\left(\mathbf{e}\right)\right) \tag{11}$$

where $y_{uv}$ is the historical interaction between $u$ and $v$, e.g., click. $\sigma(\mathbf{e}) = 1/(1 + \exp(-\mathbf{e}))$ is the logistic function.

### 4.3. Counterfactual instances-based optimization

We wish our NCGCF to be robust to unseen (unknown) user preference shifts to further enhance the recommendation robustness. Catching user preferences is at the core of any recommendation model; however, user preferences may change over time [11,23]. For example, a user may once love items with the brand *Nike* but change his taste for liking *Adidas*. Such a user preference shift can be captured by actively manipulating user preferences, i.e., manipulating $\mathbf{e}$.

Since our NCGCF is a Neural Causal Model and is capable of generating "interventional" distributions (cf. Section 3.2) within the Pearl Causal Hierarchy, the manipulations can be done by performing interventions [14] on the user preference vector $\mathbf{e}$ using a do-operator $do(\cdot)$, i.e., $do(\mathbf{e} = \mathbf{e}')$. The data after interventions are called *counterfactual instances* that, if augmented to original training instances, increase the model robustness to unseen interventions (i.e., user preference shifts). Inspired by [24], we optimize NCGCF by considering two data scenarios, i.e., the clean data scenario in which our NCGCF accesses the data without interventions, and the counterfactual data scenario in which the data is generated by known interventions on user preference vectors.

Formally, for the clean data scenario, assuming that NCGCF observes only clean data $\mathbf{D}$ during training. In this case, we retain the original value $\mathbf{o}$ of user preference $\mathbf{e}$ by using $do(\mathbf{e} = \mathbf{o})$. Then, NCGCF is trained by maximizing the likelihood function $\log p_{\theta_3}\left(\mathbf{y} \mid do(\mathbf{e} = \mathbf{o})\right)$. Since this marginal distribution is intractable [20,25], we instead maximize the intervention evidence lower-bound (ELBO) with $do(\mathbf{e} = \mathbf{o})$, i.e. $\max_{\theta_1,\theta_2,\theta_3} \mathrm{ELBO}(\mathbf{D}, do(\mathbf{e} = \mathbf{o}))$. In particular,

$$\begin{aligned}&\mathrm{ELBO}(\mathbf{D}, do(\mathbf{e}=\mathbf{o})) = \\ &\mathbb{E}_{\theta}\left[\log \frac{p_{\theta_3}(\mathbf{y} \mid do(\mathbf{e}=\mathbf{o}))p(\mathbf{u})p(\mathbf{v})}{q_{\theta_1}(\mathbf{u} \mid \Xi, do(\mathbf{e}=\mathbf{o}))\, q_{\theta_2}(\mathbf{v} \mid \Xi, do(\mathbf{e}=\mathbf{o}))}\right] \\ =&\mathbb{E}_{\theta}\left[\log p_{\theta_3}(\mathbf{y} \mid do(\mathbf{e}=\mathbf{o}))\right] \\ &- \mathrm{KL}\left(q_{\theta_1}(\mathbf{u} \mid \Xi) \| p(\mathbf{u}), q_{\theta_2}(\mathbf{v} \mid \Xi) \| p(\mathbf{v})\right)\end{aligned} \tag{12}$$

where $\Xi$ represents required parameters for the conditional probability distributions of $q_{\theta_1}$, $q_{\theta_2}$ and $p_{\theta_3}$, i.e., $\Xi = \{\mathbf{Z}_u, \mathbf{d}_u, \mathbf{A}_u\}$ for $q_{\theta_1}$, $\Xi = \{\mathbf{Z}_v, \mathbf{d}_v, \mathbf{A}_v\}$ for $q_{\theta_2}$ and $\Xi = \{\mathbf{u}, \mathbf{v}\}$ for $p_{\theta_3}$. $\theta = \{\theta_1, \theta_2, \theta_3\}$ is a set of model parameters and $\mathrm{KL}(\cdot)$ is KL-divergence between two distributions.

For the counterfactual data scenario, we assume NCGCF accesses counterfactual data $\mathbf{D}'$ generated by known interventions $do(\mathbf{e}=\mathbf{e}')$ on user preference vectors. The counterfactual vectors $\mathbf{e}'$ hold the same dimension with $\mathbf{e}$ and are drawn from a random distribution. Then, the ELBO of NCGCF with the counterfactual data is,

$$\begin{aligned}\mathrm{ELBO}(\mathbf{D}', do(\mathbf{e}=\mathbf{e}')) = \mathbb{E}_{\theta}\left[\log p_{\theta_3}\left(\mathbf{y} \mid do(\mathbf{e}=\mathbf{e}')\right)\right] \\ - \mathrm{KL}\left(q_{\theta_1}(\mathbf{u} \mid \Xi) \| p(\mathbf{u}), q_{\theta_2}(\mathbf{v} \mid \Xi) \| p(\mathbf{v})\right)\end{aligned} \tag{13}$$

Inspired by data augmentation and adversarial training, we augment the clean data with counterfactual instances to enhance the robustness of our NCGCF meanwhile capturing user preference shifts. In particular, the total loss function after augmentation is as below,

$$\begin{aligned}\mathcal{L}_{\mathrm{aug}}(\Theta) = \lambda(\mathrm{ELBO}(\mathbf{D}, do(\mathbf{e}=\mathbf{o})) \\ + (1-\lambda)(\mathrm{ELBO}(\mathbf{D}', do(\mathbf{e}=\mathbf{e}'))\end{aligned} \tag{14}$$

where $\mathcal{L}_{\mathrm{aug}}(\Theta)$ is the loss function for training our NCGCF and $\Theta$ are model parameters. $\lambda$ is the trade-off parameter between the clean and the counterfactual data scenario. During the training stage, the loss function is calculated by averaging the ELBO over all users.

## 5. Experiments

We thoroughly evaluate the proposed NCGCF for the recommendation task to answer the following research questions:

- **RQ1:** How does NCGCF perform as compared with state-of-the-art recommendation methods?
- **RQ2:** How do different components impact NCGCF's performance?
- **RQ3:** How do parameters in the causal graph encoder affect NCGCF?

### 5.1. Experimental settings

#### 5.1.1. Datasets

We conduct experiments on one synthetic dataset and three real-world datasets to evaluate NCGCF. The synthetic dataset is constructed in accordance with the Causal Graph depicted in Fig. 1(c). The construction process follows a series of assumptions that reflect causal relations between users and items. For instance, we assume the causal relation between user features and user preferences based on prior knowledge, such as the positive effect of high income on preference over high price. Similar assumptions also apply to item features to user preferences, e.g., the positive effect of the brand "Apple" on the preference for high-priced items. In particular, the **Synthetic** dataset construction is under the following four steps:

1. Feature generation: We simulate $|U| = 1{,}000$ users and $|I| = 1{,}000$ items, where each user has one discrete feature (*gender*) and one continuous feature (*income*), while each item has three discrete features, i.e., *type*, *brand* and *price*. For discrete features, their values in $\{0, 1\}$ are sampled from Bernoulli distributions. We sample continuous features from random sampling, in which random feature values are chosen from the minimum (i.e., 0) and the maximum (i.e., 1000) feature values. For both users and items, we assume two exogenous variables (i.e., $Z_u$ and $Z_v$) drawn from the Gaussian distribution.
2. Causal neighbor sampling: We synthesize the causal relations $U \to U$ and $V \to V$ by creating user/item causal neighbors. In particular, we set the causal neighbor number $N_c = 10$. We assume a user $u$'s causal neighbors (i.e., $U \to U$) are those who have interacted with the same item with the user $u$. In other words, users who have shown interest in similar items are considered causal neighbors for each other. For item causal neighbor sampling (i.e., $V \to V$), we first convert items with their features into dense vectors through item2vec [26], then calculate the Euclidean distances between two items. We assume those items that have the $N_c$ smallest distances from the target item are causal neighbors for the target item.
3. User preference estimation: For each user $u$ and item $v$, the user preference $\mathbf{u} \in \mathbb{R}^d$ towards item property $\mathbf{v} \in \mathbb{R}^d$ is generated from a multi-variable Gaussian distribution $\mathcal{N}(0, \mathbf{I})$. Then, the preference score $y_{uv}$ between user $u$ and item $v$ is calculated by the inner product of $\mathbf{u}$ and $\mathbf{v}$. Besides, we assume the fine-grained causal relations from user/item features to the preference

**Table 2**
Statistics of the datasets.

| Dataset | Synthetic | Amazon-Beauty | Amazon-Appliances | Epinions |
|---|---|---|---|---|
| # Users | 1,000 | 271,036 | 446,774 | 116,260 |
| # Items | 1,000 | 29,735 | 27,888 | 41,269 |
| # Interactions | 12,813 | 311,791 | 522,416 | 181,394 |
| # Density | 0.0128 | 0.0039 | 0.0041 | 0.0038 |

score based on prior knowledge. For example, we assume a positive effect of the "high" *income* on the preference over "high" *price*, thus tuning the preference score to prefer items with high prices. Besides, a user should have similar preference scores toward an item and the item's causal neighbors.

4. User interaction sampling: Once we obtain a user $u$'s preference scores for all items (i.e., $I$), we normalize preference scores by $\frac{\exp(r_i)}{\sum_{i' \in I} \exp(r_{i'})}$. We select items with $k$-top scores as the interactions for the user $u \in U$, where $k$ is a constant chosen randomly from range $[20, 100]$.

Apart from the synthetic dataset, we also use three benchmark datasets to test our performance in real-world scenarios. We also assume fine-grained causal relations in these real-world datasets to ensure users interact with items causally.

- **Amazon-Beauty** and **Amazon-Appliances**: two sub-datasets from Amazon Product Reviews[4] [27], which record large crawls of user reviews and product metadata (e.g., *brand*). Following [28], we use *brand* and *price* to build item features since other features (e.g., *category*) are too sparse and contain noisy information. We use co-purchased information from the product metadata to build item-item causal relations, i.e., $V \rightarrow V$. The co-purchased information records item-to-item relationships, i.e., a user who bought item $v$ also bought item $i$. We assume an item's causal neighbors are those items that are co-purchased together. For user-user causal relation (i.e., $U \rightarrow U$), we assume a user's causal neighbors are those who have similar interactions, i.e., users who reviewed the same item are neighbors for each other.
- **Epinions**[5] [29]: a social dataset recording social relations between users. We convert user/item features from the dataset into one-hot embeddings. We use social relations to build user causal neighbors, i.e., a user's social friends are the neighbors of the user. Besides, items bought by the same user are causal neighbors to each other.

For the three real-world datasets, we regard user interactions with overall ratings above 3.0 as positive interactions. For the synthetic dataset, we regard all user-item interactions as positive as they are top items selected based on users' preferences. The statistics of the four datasets are shown in Table 2. For model training, we randomly split samples in both datasets into training, validation, and test sets by the ratio of 70%, 10%, and 20%.

### *5.1.2. Baselines*

We compare NCGCF with eight competitive recommendation methods.

- **BPR** [30]: a well-known matrix factorization-based model with a pairwise ranking loss to enable recommendation learning from implicit feedback.
- **NCF** [17]: extends CF to neural network architecture. It maps users and items into dense vectors and feeds user and item vectors into an MLP to predict user preferences.
- **MultiVAE** [25]: extends CF to variational autoencoder (VAE) structure for implicit feedback modeling. It formulates CF learning as a generative model and uses variational inference to model the posterior distributions.
- **NGCF** [3]: a GCF that incorporates two GCNs to learn user and item embeddings. The learned embeddings are passed to a matrix factorization to capture the collaborative signal for recommendations.
- **VGAE** [20]: a graph learning method that extends VAE to handle graph-structured data. We use VGAE to obtain user and item embeddings and inner product those embeddings to predict user preference scores.
- **GC-MC** [5]: a graph-based auto-encoder framework for matrix completion. The encoder is a GCN that produces user and item embeddings. The learned embeddings reconstruct the rating links through a bilinear decoder.
- **LightGCN** [6]: a SOTA graph-based recommendation model that simplifies the GCN component. It includes the essential part in GCNs, i.e., neighbor aggregation, to learn user and item embeddings for collaborative filtering.
- **CACF** [31]: a method that learns attention scores from individual treatment effect estimation. The attention scores are used as user and item weights to enhance the CF.

[4] https://nijianmo.github.io/amazon/index.html.

[5] http://www.cse.msu.edu/~tangjili/trust.html.

#### 5.1.3. Evaluation metrics

We use three Top-$K$ recommendation evaluation metrics, i.e., Precision@$K$, Recall@$K$ and Normalized Discounted Cumulative Gain(NDCG)@$K$. The three evaluation metrics measure whether the recommended Top-$K$ items are consistent with users' preferences in their historical interactions. We report the average results with respect to the metrics over all users. The Wilcoxon signed-rank test is used to evaluate whether the improvements against baselines are significant.

#### 5.1.4. Parameter settings

We implement our NCGCF using Pytorch. The code and datasets are released in https://github.com/Chrystalii/CNGCF. The latent embedding sizes of neural networks for all neural-based methods are fixed as $d = 64$. The in-dimension and out-dimension of the graph convolutional layer in NCGCF, NGCF, VGAE, GC-MC and LightGCN are set as 32 and 64, respectively. We apply a dropout layer on top of the graph convolutional layer to prevent model overfitting for all GCN-based methods. The hyper-parameters of all methods are chosen by the grid search, including the learning rate $l_r$ in $\{0.0001, 0.0005, 0.001, 0.005\}$, $L_2$ norm in $\left\{10^{-5}, 10^{-4}, \cdots, 10^{1}, 10^{2}\right\}$, and the dropout ratio $p$ in $\{0.0, 0.1, \cdots, 0.8\}$. The Adam optimizer is applied to all methods for model optimization, where the batch size is fixed as 1024. We set the maximum epoch for all methods as 400 and use the early stopping strategy, i.e., terminate model training when the validation Precision@10 value does not increase for 20 epochs. To ensure a fair comparison, all baseline methods are trained using the same data used in our NCGCF. This includes using causality-enhanced node features and causal relations, such as item-item and user-user relationships, in the training process for all models.

### 5.2. Recommendation performance (RQ1)

We show the recommendation performance of our NCGCF and all baselines on the four datasets in Table 3. By analyzing Table 3, we have the following findings.

- NCGCF consistently outperforms the strongest baselines on both synthetic and real-world datasets, achieving the best recommendation performance across all three evaluation metrics. In particular, NCGCF outperforms the strongest baselines by 23.4%, 7.0%, 34.3% and 5.7% w.r.t Precision@10 on Synthetic, Amazon-Beauty, Amazon-Appliances and Epinions, respectively. Additionally, NCGCF improves 2.5%/3.8%, 8.4%/22.1%, 13.3%/35.9% and 10.6%/2.8% for Recall@10/NDCG@10 on the four datasets, respectively. The superiority of NCGCF can be attributed to two factors: the power of neural graph learning and the modeling of causality. Firstly, graph learning explicitly models the interactions between users and items as a graph, and uses graph convolutional networks to capture the non-linear relations from neighboring nodes. This allows graph learning to capture more complex user behavior patterns. Secondly, modeling causal relations allows us to identify the causal effects of different items on users, thus capturing true user preferences on items. By injecting causal modeling into graph representation learning, our NCGCF captures more precise user preferences to produce robust recommendations against baselines.
- NCGCF achieves the most notable improvements (e.g., 35.9% NDCG@10 and 43.8% NDCG@20) on the Amazon-Appliances dataset. Amazon-Appliances dataset is a large-scale dataset with a considerable amount of user behavior data that may be noisy and challenging to model. Nevertheless, NCGCF still outperforms all baselines. We consider the reason is that NCGCF injects causality into graph learning, enabling the model to surpass merely capturing spurious correlations among noisy data. This results in more accurate and reliable modeling of true user preferences.
- NGCF that uses graph representation learning outperforms NCF without graph learning. This is because NGCF models user-item interactions as a graph, and uses graph convolutional networks to capture more complex user-user collaborative behavior to enhance recommendations. In contrast, NCF uses a multi-layer perception to learn user and item similarities, which captures only linear user-item correlations from the interaction matrix. Moreover, GC-MC and LightGCN outperform other graph learning-based baselines (i.e., NGCF, VGAE) in most cases. This is because GC-MC and LightGCN aggregate multiple embedding propagation layers to capture higher-order connectivity within the interaction graph. Similarly, our NCGCF incorporates layer aggregation within our causal graph encoder, enabling us to capture higher-order connectivity and produce better graph representations for improved recommendation performance.
- NCGCF outperforms all graph learning-based baselines, including NGCF, VGAE, GC-MC and LightGCN. This is because NCGCF models causal relations within the graph learning process. Guided by the causal recommendation generation process, NCGCF is able to inject causal relations under the Structural Causal Model into the learning process of the graph convolutional network. This allows NCGCF to uncover the causal effect of items on users and capture user behavior patterns more accurately.

### 5.3. Study of NCGCF (RQ2)

We start by exploring how replacing our causal graph encoder with other graph representation learning methods, i.e., naive GCN [32], Graphsage [33] and Pinsage [34], impact NCGCF's performance. We then analyze the influences of core components, including causality-aware message passing and counterfactual instance-aware ELBO.

#### 5.3.1. Effect of causal graph encoder

The causal graph encoder plays a pivotal role in NCGCF by modeling the causal relations of nodes. To investigate its effectiveness, we replace our causal graph encoder with different encoders built by other graph learning methods. In particular, we use GCN [32], Graphsage [33] and Pinsage [34] to produce user and item embedding vectors for the decoder learning phase, and compare the

**Table 3**

Recommendation performance comparison: The best results are highlighted in bold, while the second-best ones are underlined. All improvements against the second-best results are significant at $p < 0.01$.

| Dataset | Synthetic | | | Amazon-Beauty | | | Amazon-Appliances | | | Epinions | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | Precision@10 | Recall@10 | NDCG@10 | Precision@10 | Recall@10 | NDCG@10 | Precision@10 | Recall@10 | NDCG@10 | Precision@10 | Recall@10 | NDCG@10 |
| BPR | 0.5214 | 0.4913 | 0.6446 | 0.3555 | 0.3319 | 0.4111 | 0.3720 | 0.3574 | 0.4356 | 0.3022 | 0.2895 | 0.4889 |
| NCF | 0.6120 | 0.6293 | 0.7124 | 0.3618 | 0.3659 | 0.4459 | 0.3871 | 0.3789 | 0.4771 | 0.3551 | 0.3364 | 0.5432 |
| MultiVAE | 0.6248 | 0.5999 | 0.8101 | 0.4418 | 0.4112 | 0.4616 | 0.4544 | 0.4428 | 0.5998 | 0.4229 | 0.3888 | 0.5331 |
| NGCF | 0.5990 | 0.5681 | 0.7477 | 0.4512 | 0.4003 | 0.5188 | 0.4271 | 0.3778 | 0.5555 | 0.4018 | 0.3912 | 0.5012 |
| VGAE | 0.5446 | 0.5572 | 0.7778 | 0.3499 | 0.3812 | 0.4466 | 0.3681 | 0.4014 | 0.5019 | 0.3590 | 0.3460 | 0.4913 |
| GC-MC | 0.6115 | 0.6226 | 0.8116 | 0.4666 | 0.4615 | 0.5612 | 0.4718 | 0.4518 | 0.5677 | 0.4666 | 0.4218 | 0.5112 |
| LightGCN | 0.6439 | 0.6719 | 0.8223 | 0.4810 | 0.4778 | 0.5501 | 0.4844 | 0.4652 | 0.6028 | 0.4717 | 0.4544 | 0.5436 |
| CACF | 0.4482 | 0.4158 | 0.5555 | 0.3101 | 0.3005 | 0.3888 | 0.3222 | 0.3188 | 0.4215 | 0.2899 | 0.2765 | 0.3445 |
| **NCGCF** | **0.7952** | **0.6889** | **0.8538** | **0.5148** | **0.5183** | **0.6855** | **0.6510** | **0.5271** | **0.8193** | **0.4990** | **0.5030** | **0.5589** |
| **Improv.%** | +23.4% | +2.5% | +3.8% | +7.0% | +8.4% | +22.1% | +34.3% | +13.3% | +35.9% | +5.7% | +10.6% | +2.8% |
| | Precision@20 | Recall@20 | NDCG@20 | Precision@20 | Recall@20 | NDCG@20 | Precision@20 | Recall@20 | NDCG@20 | Precision@20 | Recall@20 | NDCG@20 |
| BPR | 0.6111 | 0.5536 | 0.6338 | 0.3561 | 0.3420 | 0.4062 | 0.3941 | 0.3599 | 0.4322 | 0.3332 | 0.3232 | 0.4689 |
| NCF | 0.6678 | 0.6446 | 0.7003 | 0.3699 | 0.3691 | 0.4330 | 0.3999 | 0.4033 | 0.4519 | 0.3719 | 0.3614 | 0.5255 |
| MultiVAE | 0.6779 | 0.6136 | 0.8006 | 0.4496 | 0.4200 | 0.4555 | 0.4819 | 0.4716 | 0.5911 | 0.4465 | 0.4055 | 0.5133 |
| NGCF | 0.6233 | 0.5999 | 0.7312 | 0.4612 | 0.4112 | 0.5081 | 0.4666 | 0.4258 | 0.5499 | 0.4223 | 0.4210 | 0.4811 |
| VGAE | 0.5847 | 0.5687 | 0.7613 | 0.3551 | 0.3999 | 0.4410 | 0.3771 | 0.4228 | 0.4761 | 0.3667 | 0.3598 | 0.4781 |
| GC-MC | 0.6665 | 0.6317 | 0.8091 | 0.4781 | 0.4771 | 0.5582 | 0.4892 | 0.4881 | 0.5514 | 0.4815 | 0.4451 | 0.4999 |
| LightGCN | 0.6904 | 0.6819 | 0.8108 | 0.5023 | 0.4869 | 0.5306 | 0.4919 | 0.4781 | 0.5613 | 0.4915 | 0.4718 | 0.5221 |
| CACF | 0.4567 | 0.4266 | 0.5348 | 0.3186 | 0.3211 | 0.3678 | 0.3418 | 0.3271 | 0.4103 | 0.2747 | 0.2910 | 0.3368 |
| **NCGCF** | **0.8081** | **0.6844** | **0.8603** | **0.5153** | **0.5106** | **0.7123** | **0.6367** | **0.5055** | **0.8501** | **0.5002** | **0.5034** | **0.5667** |
| **Improv.%** | +17.0% | +0.3% | +6.1% | +2.5% | +4.8% | +27.6% | +29.4% | +3.5% | +43.8% | +1.7% | +6.6% | +7.8% |

**Table 4**
Recommendation performance after replacing the causal graph encoder with different graph representation learning methods. The percentage after + or − indicates the increase (+) or decrease (−) percentage of the variant's performance compared with the original NCGCF.

| Variants | Precision@10 | Recall@10 | NDCG@10 |
|---|---|---|---|
| | Synthetic | | |
| NCGCF | 0.7952 | 0.6889 | 0.8538 |
| NCGCF-GCN | 0.5358(-32.7%) | 0.5182(-24.7%) | 0.7025( -17.7%) |
| NCGCF-Graphsage | 0.5038(-36.8%) | 0.5005(-27.4%) | 0.7022(17.8%) |
| NCGCF-Pinsage | 0.5819(-26.8%) | 0.5498(-20.2%) | 0.7446( -12.8%) |
| | Amazon-Beauty | | |
| NCGCF | 0.5148 | 0.5183 | 0.6855 |
| NCGCF-GCN | 0.4991(-3.04%) | 0.5029(-2.97%) | 0.4886(-28.68%) |
| NCGCF-Graphsage | 0.5011(-2.67%) | 0.5039(-2.78%) | 0.5243(-23.55%) |
| NCGCF-Pinsage | 0.5008(-2.72%) | 0.5043(-2.70%) | 0.5143(-25.01%) |
| | Amazon-Appliances | | |
| NCGCF | 0.6510 | 0.5271 | 0.8193 |
| NCGCF-GCN | 0.5067(-3.04%) | 0.5167(-2.97%) | 0.6614(-28.68%) |
| NCGCF-Graphsage | 0.5085(-2.67%) | 0.5184(2.78%) | 0.6670(- 23.55%) |
| NCGCF-Pinsage | 0.5083(-2.72%) | 0.5178(-2.70%) | 0.6631(-25.01%) |
| | Epinions | | |
| NCGCF | 0.4990 | 0.5030 | 0.5589 |
| NCGCF-GCN | 0.4812(-3.55%) | 0.4990(-0.79%) | 0.5013(-10.28%) |
| NCGCF-Graphsage | 0.4809(-3.62%) | 0.4989(-0.81%) | 0.4999(-10.52%) |
| NCGCF-Pinsage | 0.4871(-2.38%) | 0.4994(-0.71%) | 0.4930(-11.74%) |

performance of NCGCF before and after the replacements. We present the experimental results in Table 4. We find that both GCN, Graphsage and Pinsage-based encoders downgrade the performance of NCGCF compared to NCGCF equipped with our proposed causal graph encoder. For instance, NCGCF with a GCN-based encoder downgrades the NDCG@10 by 28.68% on the Amazon-Beauty. This is because GCN, Graphsage and Pinsage cannot capture the causal relations of nodes in the interaction graph, leading to insufficient representations of users and items. On the contrary, our causal graph encoder captures the intrinsic causal relations between nodes using the causality-aware message passing; thus, it learns causality-aware user and item representations to better serve the later decoder learning. Moreover, the GCN-based encoder downgrades the NCGCF performance most severely compared with GraphSage and Pinsage-based encoders. This is because naive GCN performs transductive learning requiring full graph Laplacian, whereas GraphSage and Pinsage perform inductive learning without requiring full graph Laplacian to handle large-scale graph data well. We thus conclude that an inductive learning setting is more desired for our NCGCF, especially when facing large-scale graph data.

#### *5.3.2. Effect of causality-aware message passing*

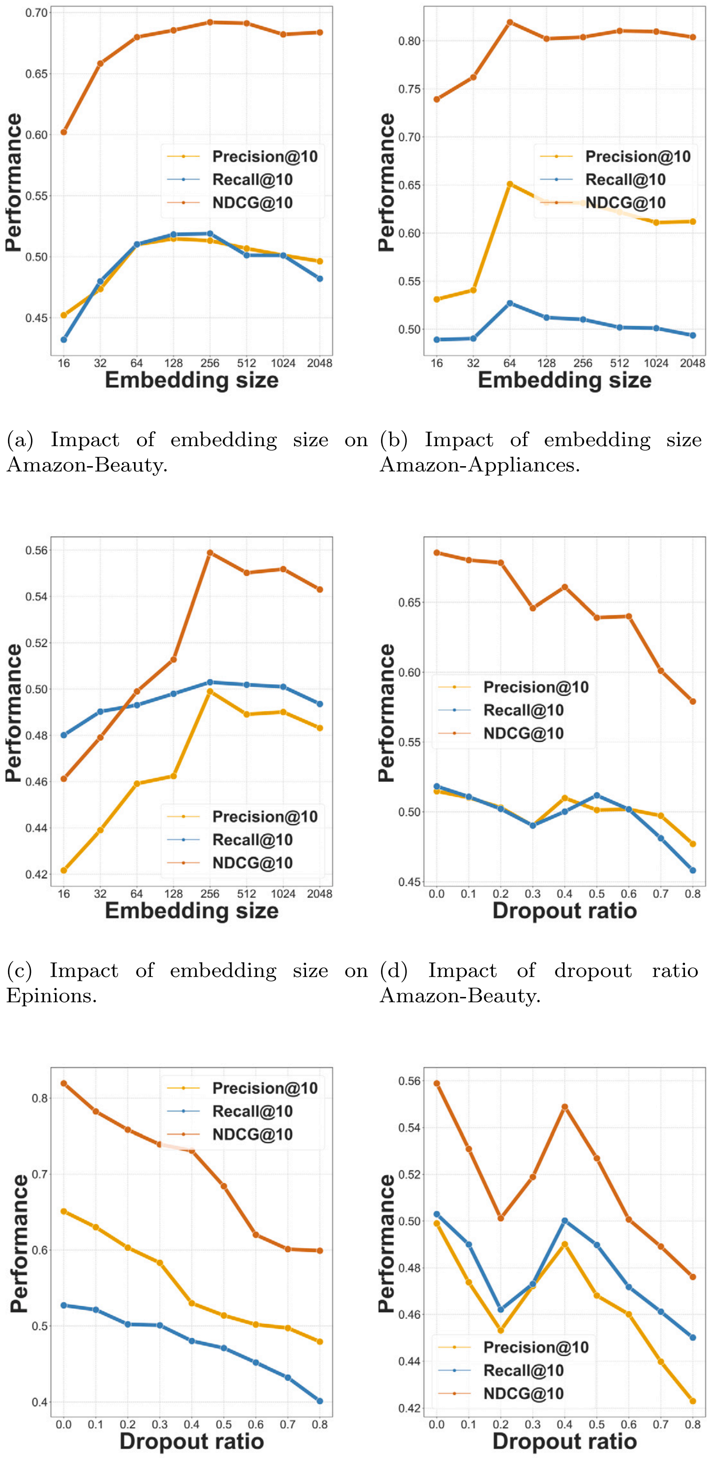
The causality-aware message passing models the dependency terms between each of the structural equations as the causal relations between nodes. We present NCGCF's performance after removing the causality-aware message passing in Table 5. We observe that removing the component downgrades NCGCF's performance, indicating the importance of causality-aware message passing in helping NCGCF achieve favorable recommendation performance. We thus conclude that modeling the causal relations between nodes within the graph-structured data is essential for graph learning-based models to uncover true user preferences for improved recommendations.

#### *5.3.3. Effect of counterfactual instance-aware ELBO*

The counterfactual instance-aware ELBO augments counterfactual instances for NCGCF optimization. We present NCGCF's performance after removing the counterfactual instance-aware ELBO in Table 5. Apparently, removing the counterfactual instance-aware ELBO leads to the downgraded performance of NCGCF on both datasets. This is because our counterfactual instance-aware ELBO augments counterfactual instances, i.e., the intervened data on user preference vectors, thus facilitating better model optimization to capture user preference shifts.

### *5.4. Parameter analysis of causal graph encoder (RQ3)*

We analyze NCGCF's performance under different embedding sizes $n$ of the semi-implicit generative model in the causal graph encoder. We also investigate the node dropout ratios $p$ of the dropout layer applied in the causal graph encoder.



(a) Impact of embedding size on Amazon-Beauty.

(b) Impact of embedding size on Amazon-Appliances.

(c) Impact of embedding size on Epinions.

(d) Impact of dropout ratio on Amazon-Beauty.

(e) Impact of dropout ratio on Amazon-Appliances.

(f) Impact of dropout ratio on Epinions.

**Fig. 3.** Parameter analysis on causal graph encoder.

**Table 5**
Ablation Study on NCGCF. ¬ CM represents causality-aware message passing is removed. ¬ CI represents counterfactual instance-aware ELBO is removed. The percentage after + or − indicates the increase (+) or decrease (−) percentage of the variant's performance compared with the original NCGCF.

| Variants | Precision@10 | Recall@10 | NDCG@10 |
|---|---|---|---|
| | Synthetic | | |
| NCGCF | 0.7952 | 0.6889 | 0.8538 |
| ¬ CM | 0.5806(−31.9%) | 0.5491(−20.3%) | 0.7179(−16.0%) |
| ¬ CI | 0.7781(−2.1%) | 0.6654(−3.4%) | 0.7573(−11.2%) |
| | Amazon-Beauty | | |
| NCGCF | 0.5148 | 0.5183 | 0.6855 |
| ¬ CM | 0.5007(−2.7%) | 0.5060(−2.3%) | 0.5383(−20.7%) |
| ¬ CI | 0.5101(−0.9%) | 0.5081(−2.0%) | 0.5738(−15.9%) |
| | Amazon-Appliances | | |
| NCGCF | 0.6510 | 0.5271 | 0.8193 |
| ¬ CM | 0.6357(−2.4%) | 0.5050(−4.2%) | 0.6864(−16.2%) |
| ¬ CI | 0.6445(−1.0%) | 0.5143(−2.4%) | 0.7956(−2.9%) |
| | Epinions | | |
| NCGCF | 0.4990 | 0.5030 | 0.5589 |
| ¬ CM | 0.4695(−6.0%) | 0.4936(−1.9%) | 0.4647(−16.9%) |
| ¬ CI | 0.4794(−3.9%) | 0.5018(−0.2%) | 0.5139(−8.1%) |

### 5.4.1. Effect of embedding size

Fig. 3 (a) (b) (c) report the parameter sensitivity of our NCGCF w.r.t. embedding size $n$ with $n = \{16, 32, 64, 128, 256, 512, 1024, 2048\}$. Apparently, the performance of NCGCF on Amazon-Beauty, Amazon-Appliances and Epinions demonstrates increasing trends from $n = 16$, then reaches the peak when $n = 512$, $n = 64$ and $n = 256$, respectively. This is reasonable since $n$ controls the number of latent vectors of users and items from the semi-implicit generative model, and low-dimensional latent vectors cannot retain enough information for the encoder learning phrase. After reaching the peaks, the performance of NCGCF degrades slightly and then becomes stable. The decrease in performance is due to the introduction of redundant information as the embedding size becomes too large, which can affect the model. Additionally, we observe the largest Amazon-Appliances dataset requires the smallest embedding size of $n = 64$ to reach its peak performance compared to the other two datasets. This is because a larger embedding size brings large-scale datasets a higher computational burden, thus limiting the model's performance.

### 5.4.2. Effect of dropout ratio

We employ a node dropout layer in the causal graph encoder to prevent model overfitting. We show the influence of node dropout ratio $p$ on the three datasets in Fig. 3 (d) (e) (f). We observe that the performance of NCGCF on both Amazon-Beauty, Amazon-Appliances and Epinions exhibits a decreasing trend as we increase the node dropout ratio $p$ from 0.0 to 0.3, but recovers at $p = 0.4$. After $p = 0.4$, the performance of NCGCF decreases as the dropout ratio increases. We believe that the reduced performance could be attributed to the removal of crucial information that the model needs to learn from the data, thus impairing the NCGCF's performance. Nevertheless, the recovered performance at $p = 0.4$ indicates that NCGCF is robust to balance the loss of information and overfitting.

## 6. Related work

### 6.1. Graph collaborative filtering

Collaborative filtering (CF) [1] dominates recommendation research due to its simplicity and effectiveness. Early CF models are largely latent factor models. They use descriptive features (e.g., IDs) to calculate user similarities, assuming that users with similar historical behaviors have similar future preferences. For example, Bayesian personalized ranking (BPR) [30] learns user and item latent vectors from the interaction matrix built by implicit user feedback, e.g., clicks. The inner products between latent vectors are used as user-item similarities to predict user preference scores.

With the burgeoning of neural models [35,36], various neural networks are used for better user preference modeling. Neural collaborative filtering (NCF) [17] uses a Multi-layer perceptron (MLP) to learn a user behavior similarity function based on simple user/item one-hot encodings. Recently, benefiting from the capability to learn from relational graphs, graph CF (GCF) leverages advances in graph learning [37] to model user-item interaction graphs as well as rich auxiliary data, e.g., text and image. Those auxiliary data have demonstrated their ability to capture valuable domain knowledge [38] and intricate semantics within user-item interactions [13], thereby enhancing recommendation performance. Early GCF relies on random walk models to calculate similarities among users and items from the given graph. With the rise of graph neural networks, recent GCF methods have shifted towards graph representation learning. Graph Convolutional Network (GCN) is one of the most widely adopted graph neural networks for scrutinizing complex graph relations as user and item embeddings. Neural graph collaborative filtering (NGCF) [3] incorporates two GCNs to learn the collaborative signal of user interactions from a user-item interaction graph. Hyperbolic Graph Collaborative Filtering (HGCF) [4] offers a compelling solution by integrating GCN with hyperbolic learning techniques to acquire user and item embeddings within the hyperbolic space. By leveraging the exponential neighborhood expansion inherent in the hyperbolic space, HGCF effectively captures higher-order relationships among users and items, enhancing the learning capabilities for downstream CF models. GC-MC [5] uses a GCN-based auto-encoder to learn latent features of users and items from an interaction graph and reconstructs the rating links for matrix completion. Later, LightGCN [6] simplifies the GCN in the recommendation task by only including neighborhood aggregation for calculating user and item representations, which further boosts the efficiency of subsequent GCF approaches, e.g., [39–41].

Existing GCN-based GCF methods only capture correlation signals of user behaviors by modeling neighboring node messages, regardless of complex node and variable dependencies. This would result in the limited ability of GCF models to capture the true user preferences in the presence of spurious correlations. On the contrary, we abandon the modeling of spurious correlations to pursue the intrinsic causal relations between nodes, which estimate the causal effect of a specific item on user preferences to uncover true user interests.

### *6.2. Causal modeling for recommendation*

Recent recommendation research has largely favored causality-enhanced methods. A burst of relevant papers is proposed to address critical issues in RS, such as data bias and model explainability with causal learning. Among them, the Structural Causal Model (SCM) from Pearl et al. [42] has been intensively investigated. SCM-based recommendation builds a graphical Causal Graph by extracting structural equations on causal relations between deterministic variables in recommendations. It aims to use the Causal Graph to conduct causal reasoning for causal effect estimation. Using the Causal Graph, a large strand of relevant approaches pursue mitigating the bad effects of different data biases, e.g., exposure bias [18,43,44], popularity bias [45]. For instance, Wang et al. [18] mitigate exposure bias in the partially observed user-item interactions by regarding the bias as the confounder in the Causal Graph. They propose a deconfounded model that performs Poisson factorization on substitute confounders (i.e., an exposure matrix) and partially observed user ratings. Other approaches also achieve explainable recommendations [46,47]. Wang et al. [46] define a Causal Graph that shows how users' true intents are related to item semantics, i.e., attributes. They propose a framework that produces disentangled semantics-aware user intent embeddings, in which each model component corresponds to a specific node in the Causal Graph. The learned embeddings are able to disentangle users' true intents towards specific item semantics, which explains which item attributes are favored by users. Our work differs from existing SCM-based recommendation models by two key points: 1) In terms of problem definition, existing works do not consider leveraging SCMs to refine the learning process of graph representations. Our method offers a novel perspective on SCM-boosted graph representation learning, leveraging the expressive power of causality-aware graph embeddings to capture complex causal relations in recommendations. 2) In terms of technique, we build the first Neural Causal Model to parameterize each structural equation under the SCM as trainable neural networks for recommendations.

## 7. Discussion

### *7.1. Potential limitations*

Our research would potentially have the following limitations: 1) Reliance on domain knowledge: The Causal Graph builds the foundation of the proposed framework, which uses causal relations to reconstruct the user-item interaction graph as a causality-oriented graph. Thus, the effectiveness of the proposed framework could hinge on the quality of the Causal Graph conceptualization. However, conceptualizing a Causal Graph necessitates domain expertise within the recommendation system domain. Therefore, the rationality of domain knowledge could influence the performance of our model. The complexity could involve understanding the recommendation mechanism and tuning Causal Graph for specific recommendation scenarios, e.g., out-of-distribution (OOD) recommendations. 2) Generalization to diverse domains: The effectiveness of the proposed framework may vary across different domains, e.g., e-commerce, social networks, or content platforms. While it is impractical to build a one-fits-all framework for different recommendation domains, enhancing the generalization ability of our proposed framework remains crucial. This can be achieved by considering techniques such as transfer learning [48], domain adaptation [49], or meta-learning [50].

### *7.2. Future direction*

For our future work, we intend to integrate transfer learning into our proposed framework to enhance its generalization across a broader range of recommendation domains. Our plan includes exploring pre-trained models or knowledge learned from related recommendation domains and adapting them to improve the performance of our framework across diverse recommendation scenarios. We also plan to explore how user feedback and interaction can further enhance our proposed framework's performance. While NCGCF improves recommendation accuracy, it is trained on one-round recommendation data without incorporating real-time user feedback. User feedback and interaction with the recommendation system are crucial factors for real-world deployment. Future studies could explore user-centric recommendations to expand the utility of NCGCF from a user experience perspective.

## 8. Conclusion

In this work, we propose *Neural Causal Graph Collaborative Filtering (NCGCF)*, the first causality-aware graph representation learning framework for graph collaborative filtering. In particular, NCGCF injects causal relations between nodes into the graph representation learning process. The integration of causal modeling and graph representation learning offers a novel perspective to facilitate accurate recommendations. In the proposed method, we first craft a Causal Graph to describe the learning process of causality-aware graph representations. Our causal graph abandons the strong assumption of user and item independence in current recommendation models. We then construct a Neural Causal Model to parameterize each of the structural equations under the Causal Graph as trainable neural networks. The proposed Neural Causal Model completes the first Neural-Causal Connection for the causal modeling of graph convolutional networks in recommendations. Finally, we approximate the Neural Causal Model using variational inference, with a semi-implicit generative model enabling causality-aware message passing for graph learning. As a result, NCGCF effectively models complex node and variable dependencies under structural equations. Extensive evaluations of recommendation performance highlight NCGCF's ability to produce precise recommendations.

## CRediT authorship contribution statement

**Xiangmeng Wang:** Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Conceptualization. **Qian Li:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Conceptualization. **Dianer Yu:** Validation, Investigation. **Wei Huang:** Validation, Methodology. **Qing Li:** Supervision, Funding acquisition. **Guandong Xu:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Guandong Xu reports financial support was provided by Australian Research Council. Xiangmeng Wang reports a relationship with University of Technology Sydney that includes: employment. Dianer Yu reports a relationship with University of Technology Sydney that includes: employment. Qian Li reports a relationship with Curtin University that includes: employment. Wei Huang reports a relationship with RIKEN Center for AIP that includes: employment. Qing Li reports a relationship with The Hong Kong Polytechnic University that includes: employment. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

The data is publicly available and shared through links in the paper. The source code for our method is released in https://github.com/Chrystalii/CNGCF.

## Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grants No.62072257 and the Australian Research Council Under Grants DP22010371, LE220100078.

## References

[1] J.B. Schafer, D. Frankowski, J. Herlocker, S. Sen, Collaborative filtering recommender systems, in: The Adaptive Web, Springer, 2007, pp. 291–324.
[2] J. Hu, B. Hooi, S. Qian, Q. Fang, C. Xu, Mgdcf: distance learning via Markov graph diffusion for neural collaborative filtering, IEEE Trans. Knowl. Data Eng. (2024).
[3] X. Wang, X. He, M. Wang, F. Feng, T.-S. Chua, Neural graph collaborative filtering, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 165–174.
[4] J. Sun, Z. Cheng, S. Zuberi, F. Pérez, M. Volkovs, Hgcf: hyperbolic graph convolution networks for collaborative filtering, in: Proceedings of the Web Conference 2021, 2021, pp. 593–601.
[5] R.v.d. Berg, T.N. Kipf, M. Welling, Graph convolutional matrix completion, arXiv preprint, arXiv:1706.02263, 2017.
[6] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, 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, 2020, pp. 639–648.
[7] X. Wang, Q. Li, D. Yu, Q. Li, G. Xu, Counterfactual explanation for fairness in recommendation, ACM Trans. Inf. Syst. (2023).
[8] X. Wang, Q. Li, D. Yu, Q. Li, G. Xu, Reinforced path reasoning for counterfactual explainable recommendation, IEEE Trans. Knowl. Data Eng. (2024).
[9] X. Wang, Q. Li, W. Zhang, G. Xu, S. Liu, W. Zhu, Joint relational dependency learning for sequential recommendation, in: Advances in Knowledge Discovery and Data Mining: 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11–14, 2020, in: Proceedings, Part I, vol. 24, Springer, 2020, pp. 168–180.
[10] D. Yu, Q. Li, X. Wang, Z. Wang, Y. Cao, G. Xu, Semantics-guided disentangled learning for recommendation, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2022, pp. 249–261.
[11] X. Wang, Q. Li, D. Yu, G. Xu, in: Off-Policy Learning over Heterogeneous Information for Recommendation, WWW'22, Association for Computing Machinery, New York, NY, USA, 2022, pp. 2348–2359.
[12] X. Wang, Q. Li, D. Yu, Q. Li, G. Xu, Constrained off-policy learning over heterogeneous information for fairness-aware recommendation, ACM Trans. Recommender Syst. (2023).
[13] X. Wang, Q. Li, D. Yu, P. Cui, Z. Wang, G. Xu, Causal disentanglement for semantics-aware intent learning in recommendation, IEEE Trans. Knowl. Data Eng. (2022).

[14] E. Bareinboim, J.D. Correa, D. Ibeling, T. Icard, On pearl's hierarchy and the foundations of causal inference, in: Probabilistic and Causal Inference: The Works of Judea Pearl, 2022, pp. 507–556.

[15] K. Xia, K.-Z. Lee, Y. Bengio, E. Bareinboim, The causal-neural connection: expressiveness, learnability, and inference, Adv. Neural Inf. Process. Syst. 34 (2021) 10823–10836.

[16] J. Pearl, et al., Models, Reasoning and Inference 19(2) Cambridge University Press, Cambridge, UK, 2000.

[17] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.-S. Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 173–182.

[18] Y. Wang, D. Liang, L. Charlin, D.M. Blei, The deconfounded recommender: a causal inference approach to recommendation, arXiv preprint, arXiv:1808.06581, 2018.

[19] J. Pearl, D. Mackenzie, The Book of Why: the New Science of Cause and Effect, Basic Books, 2018.

[20] T.N. Kipf, M. Welling, Variational graph auto-encoders, arXiv preprint, arXiv:1611.07308, 2016.

[21] M. Yin, M. Zhou, Semi-implicit variational inference, in: International Conference on Machine Learning, PMLR, 2018, pp. 5660–5669.

[22] B. Altaf, U. Akujuobi, L. Yu, X. Zhang, Dataset recommendation via variational graph autoencoder, in: 2019 IEEE International Conference on Data Mining (ICDM), IEEE, 2019, pp. 11–20.

[23] X. Wang, Q. Li, D. Yu, Z. Wang, H. Chen, G. Xu, Mgpolicy: meta graph enhanced off-policy learning for recommendations, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 1369–1378.

[24] C. Zhang, K. Zhang, Y. Li, A causal view on robustness of neural networks, Adv. Neural Inf. Process. Syst. 33 (2020) 289–301.

[25] D. Liang, R.G. Krishnan, M.D. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: Proceedings of the 2018 World Wide Web Conference, 2018, pp. 689–698.

[26] O. Barkan, N. Koenigstein, Item2vec: neural item embedding for collaborative filtering, in: 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP), IEEE, 2016, pp. 1–6.

[27] R. He, J. McAuley, Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering, in: Proceedings of the 25th International Conference on World Wide Web, 2016, pp. 507–517.

[28] T.U. Haque, N.N. Saber, F.M. Shah, Sentiment analysis on large scale Amazon product reviews, in: 2018 IEEE International Conference on Innovative Research and Development (ICIRD), IEEE, 2018, pp. 1–6.

[29] J. Tang, H. Gao, H. Liu, mTrust: discerning multi-faceted trust in a connected world, in: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, ACM, 2012, pp. 93–102.

[30] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, in: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI'09, AUAI Press, Arlington, Virginia, USA, 2009, pp. 452–461.

[31] J. Zhang, X. Chen, W.X. Zhao, Causally attentive collaborative filtering, in: Proceedings of the 30th ACM International Conference on Information &, Knowledge Management, 2021, pp. 3622–3626.

[32] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint, arXiv:1609.02907, 2016.

[33] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, Adv. Neural Inf. Process. Syst. 30 (2017).

[34] R. Ying, R. He, K. Chen, P. Eksombatchai, W.L. Hamilton, J. Leskovec, Graph convolutional neural networks for web-scale recommender systems, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 974–983.

[35] T. Radhika, A. Chandrasekar, V. Vijayakumar, Q. Zhu, Analysis of Markovian jump stochastic Cohen–Grossberg bam neural networks with time delays for exponential input-to-state stability, Neural Process. Lett. 55 (8) (2023) 11055–11072.

[36] M.S. Aslam, T. Radhika, A. Chandrasekar, Q. Zhu, Improved event-triggered-based output tracking for a class of delayed networked T–S fuzzy systems, Int. J. Fuzzy Syst. (2024) 1–14.

[37] F. Xia, K. Sun, S. Yu, A. Aziz, L. Wan, S. Pan, H. Liu, Graph learning: a survey, IEEE Trans. Artif. Intell. 2 (2) (2021) 109–127.

[38] Y. Tang, Z. Pan, X. Hu, W. Pedrycz, R. Chen, Knowledge-induced multiple kernel fuzzy clustering, IEEE Trans. Pattern Anal. Mach. Intell. (2023).

[39] J. Fu, R. Gao, Y. Yu, J. Wu, J. Li, D. Liu, Z. Ye, Contrastive graph learning long and short-term interests for poi recommendation, Expert Syst. Appl. 238 (2024) 121931.

[40] X. Ni, F. Xiong, Y. Zheng, L. Wang, Graph contrastive learning with kernel dependence maximization for social recommendation, in: Proceedings of the ACM on Web Conference 2024, 2024, pp. 481–492.

[41] W. Wei, X. Ren, J. Tang, Q. Wang, L. Su, S. Cheng, J. Wang, D. Yin, C. Huang, Llmrec: large language models with graph augmentation for recommendation, in: Proceedings of the 17th ACM International Conference on Web Search and Data Mining, 2024, pp. 806–815.

[42] J. Pearl, Causality, Cambridge University Press, 2009.

[43] Q. Li, X. Wang, Z. Wang, G. Xu, Be causal: de-biasing social network confounding in recommendation, ACM Trans. Knowl. Discov. Data (2022).

[44] D. Yu, Q. Li, X. Wang, G. Xu, Deconfounded recommendation via causal intervention, Neurocomputing 529 (2023) 128–139.

[45] Y. Zhang, F. Feng, X. He, T. Wei, C. Song, G. Ling, Y. Zhang, Causal intervention for leveraging popularity bias in recommendation, arXiv preprint, arXiv:2105.06067, 2021.

[46] W. Wang, X. Lin, F. Feng, X. He, M. Lin, T.-S. Chua, Causal representation learning for out-of-distribution recommendation, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 3562–3571.

[47] D. Yu, Q. Li, X. Wang, Q. Li, G. Xu, Counterfactual explainable conversational recommendation, IEEE Trans. Knowl. Data Eng. (2023).

[48] Z. Xu, W. Pan, Z. Ming, Transfer learning in cross-domain sequential recommendation, Inf. Sci. 669 (2024) 120550.

[49] J. Fu, F. Yuan, Y. Song, Z. Yuan, M. Cheng, S. Cheng, J. Zhang, J. Wang, Y. Pan, Exploring adapter-based transfer learning for recommender systems: empirical studies and practical insights, in: Proceedings of the 17th ACM International Conference on Web Search and Data Mining, 2024, pp. 208–217.

[50] E. Yu, Z. Ye, Z. Zhang, L. Qian, M. Xie, A federated recommendation algorithm based on user clustering and meta-learning, Appl. Soft Comput. 158 (2024) 111483.

**Xiangmeng Wang** has been a Ph.D. student at the School of Computer Science, Faculty of Engineering and Information Technology, University of Technology Sydney (UTS). She received her MSc degree in Computer Application Technology from Shanghai University. Her general research interests lie primarily in explainable artificial intelligence, data analysis, and causal machine learning. Her papers have been published in top-tier conferences and journals in the field of machine learning.

**Qian Li** is a Lecturer at the School of Engineering, Computing and Mathematical Sciences (EECMS), Curtin University, Perth, Australia. Her general research interests lie primarily in optimization algorithms and causal machine learning. Her papers have been published in top-tier conferences and journals in the field of machine learning and computer vision.

**Dianer Yu** has been a Ph.D. candidate at the School of Computer Science, Faculty of Engineering and Information Technology, University of Technology Sydney (UTS). He received MSc and BSc degree in Computer Science from UTS. His general research interests lie primarily in data mining, causal inference and explainable machine learning.

**Wei Huang** is a postdoctoral researcher at RIKEN Center for Advanced Intelligence Project (AIP). He obtained a Ph.D. degree in Computer Science at the University of Technology Sydney (UTS). He received his Master and Bachelor degree in Statistical Physics from the University of Science and Technology of China. His research interests lie in explainable artificial intelligence, deep learning theory, and graph representation learning.

**Qing Li** is a Chair Professor (Data Science) and the Head of the Department of Computing, the Hong Kong Polytechnic University. Prof. Li has been actively involved in the research community by serving as an associate editor and reviewer for technical journals, and as an organizer/co-organizer of numerous international conferences. He is currently a Fellow of IEEE and IET/IEE, a member of ACM-SIGMOD and IEEE Technical Committee on Data Engineering. He is the chairperson of the Hong Kong Web Society, and also served/is serving as an executive committee (EXCO) member of IEEE-Hong Kong Computer Chapter and ACM Hong Kong Chapter. In addition, he serves as a councilor of the Database Society of Chinese Computer Federation (CCF), a member of the Big Data Expert Committee of CCF, and is a Steering Committee member of DASFAA, ER, ICWL, UMEDIA, and WISE Society.

**Guandong Xu** is a Professor in the School of Computer Science and Advanced Analytics Institute at University of Technology Sydney. He received MSc and BSc degree in Computer Science and Engineering, and PhD in Computer Science. He currently heads the Data Science and Machine Intelligence Lab, which consists of 15+ members of academics, research fellows and HDR students. From Nov 2019, he directs the newly established Smart Future Research Centre, which is an across-disciplines industry engagement and innovation platform for AI and Data Science Application towards smart wealth management and investment, energy, food, water, living, and city.