Cognitive Computation (2024) 16:3391–3404
https://doi.org/10.1007/s12559-024-10321-0

RESEARCH

Disentangling User Cognitive Intent with Causal Reasoning for Knowledge-Enhanced Recommendation

Hongcai xu[1] · Junpeng Bao[1] · Qika Lin[2] · Lifang Hou[1] · Feng Chen[3]

Received: 8 February 2024 / Accepted: 29 June 2024 / Published online: 18 July 2024
© The Author(s) 2024

Abstract
The primary objective of an effective recommender system is to provide accurate, varied, and personalized recommendations that align with the user's cognitive intents. Given their ability to represent structural and semantic information effectively, knowledge graphs (KGs) are increasingly being utilized to capture auxiliary information for recommendation systems. This trend is supported by the recent advancements in graph neural network (GNN)-based models for KG-aware recommendations. However, these models often struggle with issues such as insufficient user-item interactions and the misalignment of user intent weights during information propagation. Additionally, they face a popularity bias, which is exacerbated by the disproportionate influence of a small number of highly active users and the limited auxiliary information about items. This bias significantly curtails the effectiveness of the recommendations. To address this issue, we propose a Knowledge-Enhanced User Cognitive Intent Network (KeCAIN), which incorporates item category information to capture user intents with information aggregation and eliminate popularity bias based on causal reasoning in recommendation systems. Experiments on three real-world datasets show that KeCAIN outperforms state-of-the-art baselines.

Keywords Recommendation · Knowledge graph · User cognitive intent representation · Causal reasoning

Introduction

Cognition-aware recommendations are designed to identify the personalities and intents of users while looking at their personality characteristics and preferences over time. To enrich both the user's cognitive understanding and the capabilities of the recommendation system, knowledge graphs (KGs) have been utilized to enhance recommendation accuracy. KGs contain extensive entity and relationship information that can uncover user intent or preferences [1]. The original methods improved item representations by learning embeddings from KG triples, typically as content information or as a priori [2, 3]. By integrating multi-hop paths and semantic graph structure, subsequent studies attempted to improve the user-item relationships [4, 5].

However, these methods only make recommendations based on the similarity of entities and do not take into account the user's cognitive intent to interact with the knowledge graph. Furthermore, they often exhibit poor transferability to different domains and unstable performance. Recommendation methods have been employing graph neural networks (GNNs) for developing end-to-end models recently [6–9]. These models use information aggregation schemes that effectively integrate multi-hop neighbors into the target node representation. With graph structure information, these GNN-based models achieve good recommendation performance. But in these recommender systems, they don't learn how relationships between entities or items affect the user's intent.

In recent years, some knowledge graph-based methods have been proposed to explore the user's intent behind user-item interactions [4, 10, 11]. Wang et al. [10] model the user intent and relational paths in the knowledge graph to improve recommendation performance. However, despite introducing the attention mechanism for cognitive intent calculation, the final user representation does not sufficiently define the user's preferences. The aggregation approach does not reflect the user's cognitive intent for item category information. This approach also lacks cognitive analysis between items and user intent, making it difficult to model specific items for specific users' intent based on item category information.

In most cases, the distribution of items has a long-tail effect, meaning a small number of popular items account for the majority of user-item interaction data. Recommendation

Extended author information available on the last page of the article

systems typically infer users' intents based on such biased interaction data, which may influence users' intents to some extent due to popularity bias. To overcome the problem of popularity bias, several studies have developed recommender methods that focus on long-tail items [12]. By re-weighting their interactions in the training loss [13], reducing bias amplification [14], integrating balanced training data [15], or disentangling user and item embeddings [16], the goal of these methods is to reduce the influence of popular items on the user's intent. Although these methods have demonstrated strong performance, they lack a systematic view of how item popularity affects users' intent and do not consider the popularity bias mechanism.

Recently, causal inference has demonstrated great potential for mining users' intent in user-item interaction records for personalized recommendation tasks. Generally, a causal graph is created to analyze the causal relationship and discover cognitive patterns between various task-related components [14, 17–19]. Bonner and Vasile [15] analyzed the recommendation problem from the perspective of causal inference and proposed the use of the Inverse Propensity Score (IPS) to deal with bias. Another study utilized counterfactual inference to reduce user-induced cognitive bias [17]. To counteract popularity bias in recommendation scenarios, [19] proposed the DICE model, which considers users' interest in items.

In this paper, we propose a knowledge-enhanced cognitive intent network for a recommendation that develops a causal graph to minimize popular bias and incorporates user-item interaction information between different kinds of items and user intents. The method uses a multi-task learning framework to model each influencing factor separately, which eliminates popularity bias to a certain extent and improves personalized recommendation performance.

The contributions of our work can be summarized as follows:

- We propose a method for aggregating information in recommendation systems using a knowledge-based graph neural network. This method leverages knowledge graphs and user-item interaction data to integrate user intents across different item categories into the information interaction pathway.
- We develop a causal graph model that utilizes user-item interaction data to mitigate popularity bias in recommender systems. This model allows us to systematically analyze the impact of popularity bias on recommendations by tracing the causal relationships within the graph.
- We conducted extensive experiments on three real-world industrial datasets to demonstrate the effectiveness of our model, KeCAIN.

The rest of this paper is organized as follows. "Related Work" section briefly reviews the related work of knowledge-aware cognitive recommendation systems. In the "Problem Statement and Background" section, we introduce the problem statement and background. In "Methodology" section, we propose a method (KeCAIN) including a knowledge-aware graph user intent network and cognitive recommendations based on a causal reasoning module. Afterward, we discuss experiments of knowledge-aware cognitive recommendation systems in "Experimental Analysis" section. "Conclusion" section concludes this paper.

Related Work

Cognitive Recommendation System

The recommender system is the interaction between the user (a human) and the system. There are various considerations as to why the recommender system needs to incorporate the cognitive intelligence approach: first, the user's decision-making is based on the human process [20], and to understand the decision-making process, one needs to be able to memorize the user's historical behaviors; second, it is necessary to understand the decision-making points in terms of the user's behaviors, the user's preferences, and the user's intents; and third, the recommender system provides interactions to match the user's cognition well. Nápoles et al. [21] construct a recommender system using long-term cognitive networks, a kind of recurrent neural network that allows the utilization of prior knowledge in reasoning. A method for calculating similarity is presented that integrates rating information with variables, including movie genre and release year [22]. An adaptive learning cognitive map model is presented as a solution to the issue of resource recommendation systems' closed and passive applicability [23]. Liang et al. [24] study enables goal-focused exploration in recommender systems by incorporating clear feedback from users about their personal goals.

Knowledge-Aware Recommendation Systems

Recommendation algorithms based on knowledge graphs can be divided into three categories: embedding-based methods, path-based methods, and methods based on a combination of path and embedding. Embedding-based approaches enrich the representation of items and users with the help of knowledge graphs, using knowledge graph embedding algorithms such as [25, 26]. Path-based methods use connected paths of entities in the KG to construct the user-item graph and embedding for the recommendation with the representative method (FuzzyRec [27]). The joint method of embedding

and path combines the advantages of the above two methods for in-depth analysis, which is an important research direction in current recommendation algorithms. Representative methods include KGCN (Knowledge Graph Convolutional Network) [28], KGAT (Knowledge Graph Attention Network) [2], KGIN [10], and others.

Causal Reasoning in Recommendation

The issue of popularity bias has been extensively researched in the field of recommendation systems (RS). Prior research has primarily focused on the problem of bias in selection when utilizing feedback data that is missing not at random (MNAR) to ensure unbiased learning [29]. To reduce selection biases in RS, some researchers have proposed unbiased inverse propensity score (IPS) estimators based on causal inference estimation approaches and positive-unlabeled learning [29]. However, the IPS estimator's variance can fluctuate significantly [30–33]. To address this issue, other studies have proposed doubly robust (DR) estimators for unbiased RS learning [34]. Causal inference, which systematically analyzes the relationship between cause and effect, is a popular approach in this field [35]. Additionally, some studies have employed causal intervention techniques to tackle the problem of popularity bias [18, 19].

Problem Statement and Background

Problem Definition

We consider a system with a set of users $U = u_1, u_2, ..., u_N$ and a set of items $I = i_1, i_2, ..., i_M$. Each user's behavior is represented by a triplet $(u, \text{Interact}, I)$, where $y_{ui} = 1$ denotes a binary relationship between user u and item i. To incorporate item information, we create a knowledge graph expressed formally as $\{(h, r, t) \mid h, t \in E, r \in R\}$, where each triplet describes a relationship r between the head and tail entities. We also define $A = \{(i, r, e) \mid i \in I, r \in R, e \in E\}$ as a collection of item-entity alignments, where (i, e) indicates that item i can be aligned with entity e based on r in the KG.

Firstly, the user's cognitive intent is learned and fused into the recommender system using relational information and the structure of the knowledge graph to examine the impact of user's cognitive intent on a recommendation. Second, the causal effect computation of counterfactual reasoning is applied to enhance the diversity of item recommendations, providing the user with new cognitive information and eliminating the popularity bias.

Causal Graph and Causal Effect

The causal graph is a directed acyclic graph defined by $G = V, E$, where V denotes the set of variables and E denotes the cause-effect relationships between those variables. The cause-effect relationship refers to a relationship between two phenomena, where one phenomenon is the reason behind the other. In the causal graph, a variable is denoted by a capital letter, such as I, and its observed value is denoted by a lowercase letter, such as i. An edge indicates that the successor node is an effect Y, and the ancestor node is a cause I. For example, in Fig. 1, $I \rightarrow Y$ denotes the presence of a direct relationship between I and Y. Moreover, the path $I \rightarrow C \rightarrow Y$ indicates that I influence Y indirectly through a mediator C.

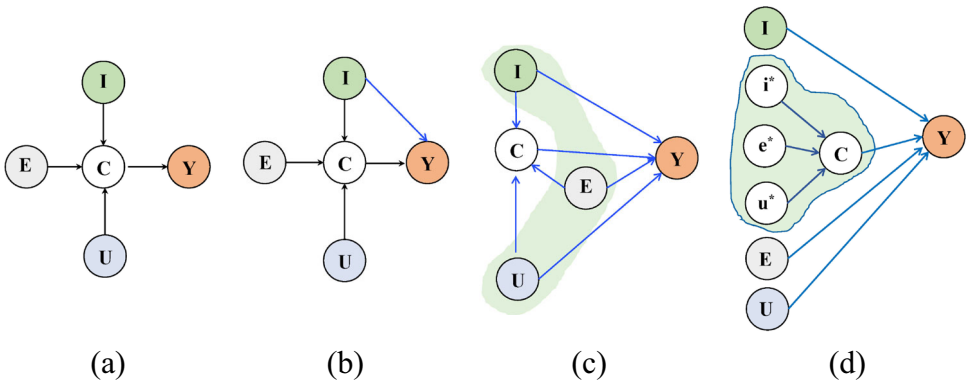
To determine the value of Y according to the causal graph, we use the values of Y's ancestor nodes, which are defined as:

$$Y_{i,C} = Y(I = i, C = c), \tag{1}$$

Here, $Y(.)$ refers to the Y' value function. Similarly, we can obtain the mediator's value by setting $c = C_i = C(I = i)$. We can then create a solution that predicts the value of Y from I by establishing $C(I)$ and $Y(I, C)$ as neural operators, such as a fully connected neural network layer.

The causal effect of I on Y refers to the extent to which a unit change in the ancestor variable I impacts the target variable Y. The total effect (TE) of $I = i$ on Y can be defined as:

Fig. 1 Causal graph



$$TE = Y_{i,C_i} - Y_{i^*,C_{i^*}}, \tag{2}$$

This equation captures the difference between two hypothetical scenarios: $I = i$ and $I = i^*$. Here, $I = i^*$ represents a condition in which the value of I is artificially suppressed to zero. When $I = i^*$, the value of C is C_{i^*}. We can further decompose TE into natural direct effect (NDE) and total indirect effect (TIE), which correspond to the influence of I on Y via the direct path $I \rightarrow Y$ and the indirect path $I \rightarrow C \rightarrow Y$ in the causal graph, respectively.

If we set C to the value of C_{i^*}, then NDE captures the value change of Y when I shift from i^* to i along the direct path $I \rightarrow Y$, which can be expressed as:

$$NDE = Y_{i,C_{i^*}} - Y_{i^*,C_{i^*}}, \tag{3}$$

Here, $Y_{i,C_{i^*}} = Y(I = i, C = C(I = i^*))$. However, since the same variable I needs to be set to different values on different paths, computing Y_i and C_i^* requires counterfactual inference, as shown in Fig. 1. In Fig. 1(b), (c), and (d), E denotes the influence of the knowledge graph, and a star on a letter represents the counterfactual influence. In our experiments, we assume that the mean is used in the calculations. Finally, we can calculate TIE by subtracting NDE from TE:

$$TIE = TE - NDE = Y_{i,C_i} - Y_{i,C_{i^*}}, \tag{4}$$

This equation captures how I affects Y through the backward path $I \rightarrow C \rightarrow Y$.

Methodology

Graph neural networks (GNNs) have revolutionized the encoding of user intents hidden within user-item interactions into the learning process of networks. This approach generates embedding vectors for users and items that encapsulate critical information. Furthermore, GNN algorithms, particularly those leveraging knowledge graphs, deepen the understanding of higher-order relationships between items, thereby enhancing the accuracy of recommendation systems. In this paper, we propose an item category information recommendation algorithm based on the knowledge graph neural network. Inspired by causal graphs, the issue of popularity bias can be explored from a causal perspective to address it in recommendation scenarios. To do this, it is necessary to identify several key factors that impact the probability of user-item interactions, including the interaction between the user, item, and knowledge graph, the distribution of item popularity, the consistency of users, and the density of the knowledge graph. Based on cognitive analysis, a causal effect graph of the recommendation process can be constructed, as shown in Fig. 1(c). Here, U represents user consistency, E represents knowledge graph density, I denotes item popularity in the recommendation scenario, C represents the interaction between the user, item, and knowledge item, and Y represents the final recommendation score.

Knowledge-Aware Graph User Intent Network

The knowledge-aware graph user intent network (KGUIN) is divided into three modules, namely the user cognitive intent network module, score prediction module, and learning module, as shown in Fig. 2.

User Cognitive Intent Module

In the user's cognitive intent module, it is assumed that each user selects a certain item for a specific intent, which is referred to as the user's intent. The user intent set, denoted as P, can be used to divide (u, i) into $(u, p, i)|p \in P$, which reorganizes the original heterogeneous graph. Each user intention is matched with the relation in the knowledge graph, and the attention mechanism is used to construct a vector of user intents.

$$e_p = \sum_{r \in R} \alpha(r, p)\, e_r, \tag{5}$$

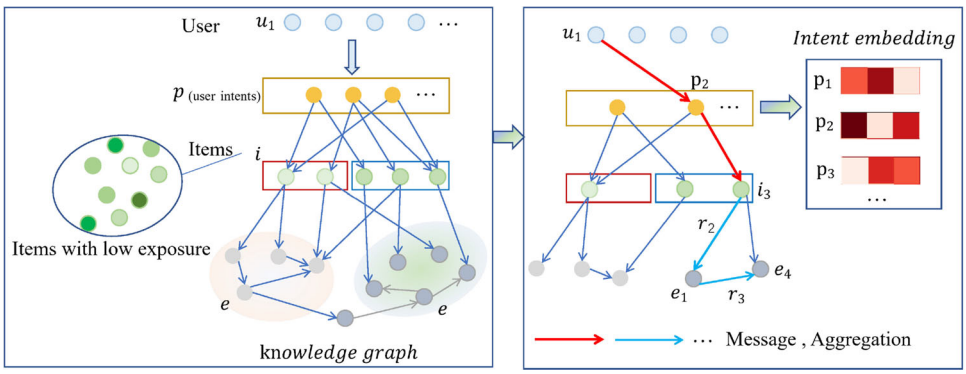


Fig. 2 The knowledge-aware graph user cognitive intent network (KGUIN) framework shows how relations 2 and 3 in the knowledge graph affect user 1's intent to purchase item 3

Where e_r denotes the ID embedding vector of relation r, which is assigned an $\alpha(r, p)$ importance score. The calculation formula for this score is:

$$\alpha(r, p) = \frac{\exp\left(w_{rp}\right)}{\sum_{r' \in R} \exp\left(w_{r'p}\right)}, \tag{6}$$

Where w_{rp} denotes a trainable weight matrix, which corresponds to a specific relationship and a specific user's cognitive intent. Then this attention score does not belong to only one user. As long as a user has a combination of these relations and intents, the score will be assigned to this user.

Different user intents should be independent of each other to provide distinct information that describes a user's behavior, and the embedding of the user intents should be as independent as possible. The distance correlation coefficient is used as a regular term to measure the independence between two intentions. The smaller the distance correlation coefficient between two intents, the more independent they are. The calculation formula for the distance correlation coefficient is as follows:

$$L_{ING} = \sum_{p, p' \in P, p \neq p'} d_{Cor}\left(e_p, e_{p'}\right), \tag{7}$$

$$d_{Cor}\left(e_p, e_{p'}\right) = \frac{d_{Cov}\left(e_p, e_{p'}\right)}{\sqrt{d_{Var}\left(e_p\right) * d_{Var}\left(e_{p'}\right)}}, \tag{8}$$

Where d_{Cor} denotes the distance correlation coefficient between user intent p. d_{Cov} and d_{Var} denote the distance covariance and distance variance of each expression vector, respectively.

In the aggregation of relational paths, the idea of collaborative filtering is used. Collaborative filtering assumes that users with similar behaviors will have similar intents for items. Therefore, it is assumed that users with similar intents have similar preferences for items. Let $N_u = (p, i)|(u, p, i) \in C$ denote the user intention history and the first-order correlation of user u. The user intention information is then integrated with all historical interaction items, enabling a representation of the user intention module.

$$e_u^{l\,(1)} = \frac{1}{|N_u|} \sum_{(p,i) \in N_u} \beta(u, p)\, e_p \odot e_i^{l\,(0)}, \tag{9}$$

The ID embedding vector of user intention module item i is denoted as $e_i^{l\,(0)}$. The Hadamard product, denoted as $\odot$, is used since each user intention should have a different driving force. Therefore, an attention score β is designed to distinguish the importance of each potential factor p for user u. $\beta(u, p)$ is calculated as follows:

$$\beta(u, p) = \frac{\exp\left(e_p^T e_u^{l\,(0)}\right)}{\sum_{p' \in P} \exp\left(e_{p'}^T e_u^{l\,(0)}\right)}, \tag{10}$$

The ID embedding vector of user u is denoted as $e_u^{l\,(0)}$. In the knowledge graph aggregation layer, $N_i = (r, v)|(i, r, v) \in G$ is used to represent the attributes of item i and its first-order linked entities, taking into account the relational context of the aggregation function. Each entity has different semantics in different relations, so the representation of item i is produced as follows:

$$e_i^{l\,(1)} = \frac{1}{|N_i|} \sum_{(r,v) \in N_i} e_r \odot e_v^{l\,(0)}, \tag{11}$$

where $e_i^{l\,(1)}$ denotes the item embedding vector after the aggregation of the first layer of adjacency information, and $e_v^{l\,(0)}$ denotes the ID embedding vector of the entity v. This expression allows for the consideration of different relationship entities e_r and their corresponding semantics, even if the final attribute entity v is the same. Personalized weighting is an operational function that follows a specific process. To elaborate, consider a scenario where there are four items, i_1, i_2, i_3, and i_4, in the item library and a user u_1 buys items i_1, i_2, and i_3. In the adjacency matrix, the corresponding vector is $(1, 1, 1, 0)$. Clustering the items based on K-means or spectral methods yields i_1 and i_2 belonging to the first category and i_3 belonging to the second category, with i_4 belonging to the third category. Consequently, for user u_1, there are two categories of items to browse, with two items in the first category and one item in the second category. Therefore, the weight of the first category of items is 2/3, and the weight of the second category of items is 1/3, resulting in a vector in the adjacency matrix represented as (2/3, 2/3, 1/3, 0). After normalization, the final vector is (2/5, 2/5, 1/5, 0) in the adjacency matrix.

Score Prediction Module

Upon completing the aforementioned modules, we obtain the user vector e_u^k of the user's cognitive intention module, the item vector e_i^k of the user intention module, the category-weighted user vector $e_u^{c^k}$, and the category-weighted item vector $e_i^{c^k}$. By performing addition operations on these vectors, we derive the user vector e_u and item vector e_i of the layer.

$$e_u{}^{(k)} = e_u^{l\,(k)} + e_u^{c\,(k)}. \tag{12}$$

$$e_i{}^{(k)} = e_i^{l\,(k)} + e_i^{c\,(k)}. \tag{13}$$

As graph neural networks can propagate across multiple layers during training, we obtain multiple layers of user vector e_u and item vector e_i. To ensure that the embedded vectors contain rich information, we perform addition operations on the user vector and item vector generated by each layer.

$$e_u = {e_u}^{(0)} + \ldots + {e_u}^{(k)}. \tag{14}$$

$$e_i = {e_i}^{(0)} + \ldots + {e_i}^{(k)}. \tag{15}$$

The final user vector is denoted by e_u, and the final item vector is denoted by e_i. By multiplying these two vectors, we obtain the final score.

$$y_{ui} = {e_u}^T e_i. \tag{16}$$

Learning

Finally, we optimize the model using the Bayesian Personalized Ranking (BPR) loss function.

$$L_{BPR} = \sum_{(u,i,j)\in O} -Ln\sigma\left(y_{ui} - y_{uj}\right), \tag{17}$$

$$L_{KeCAIN} = L_{BPR} + \lambda_1 L_{ING} + \lambda_2 ||\Theta||_2^2, \tag{18}$$

The model's parameters are represented by Θ, where $e_u^{(0)}$, $e_v^{(0)}$, e_r, e_p, and w correspond to entities $u \in U$, items $i \in I$, relations r, intents $p \in P$, and their associated weights, respectively. The hyperparameter λ_1 controls the independence loss, while λ_2 is a regularization.

Technically, the intent embedding is created through an attentive combination of relation embeddings, where more significant relations are assigned larger attribution scores. This leads to what we call Relational Path-aware Aggregation. Unlike node-based aggregation mechanisms, we treat a relational path as an information channel and embed each channel into a representative vector. Given that user-intent-item triplets and KG (Knowledge Graph) triplets are heterogeneous, we employ distinct aggregation strategies for these two parts. This approach allows us to effectively distill the behavioral patterns of users and the relatedness of items separately.

Cognitive Recommendations Based on Causal Reasoning

This subsection focuses on the model-agnostic causal reasoning method (KGUIN-Causal), which serves as a multi-task learning framework to tackle the problem of item popularity bias through counterfactual reasoning. Based on the causal graph and causal effect, the KeCAIN model can be broken down into four modules: the KGUIN recommendation module, the user module, the item module, and the knowledge graph module. Recommender systems are influenced by various perceptual factors such as the word-of-mouth effect, promotional activities, and item quality, resulting in a long-tail distribution of user interactions with items. Models that fit this distribution inherit its biases during training, tending to recommend popular products rather than capturing user preferences and cognitive alignment with items from the perspective of user-item matches. Excessive bias may lead to the loss of personalization in the recommender system. The proposed model utilizes counterfactual theory from causal inference, incorporating factors related to users, items, and entities in the knowledge graph, thus effectively mitigating these biases.

The recommendation module, which is constructed as a multi-layer perceptron, receives representations from the user module, item module, and knowledge graph module, respectively. The user and item modules can be implemented as multi-layer perceptrons, and the knowledge graph module can be implemented as a graph neural network.

The framework of KeCAIN is illustrated in Fig. 3. In the recommendation module, the ranking score between the user and item is denoted by $\widehat{y_c} = Y_c(C(U = u, I = i, E = e))$, reflecting the extent to which item i matches the preference of the user u.

In the item module, the effect of item popularity is captured by $\widehat{y_i} = Y_i(I = i)$, where the score increases with the popularity of items. In the user module, $\widehat{y_u} = Y_u(U = u)$ demonstrates the degree to which a person interacts with an item, irrespective of their preferences.

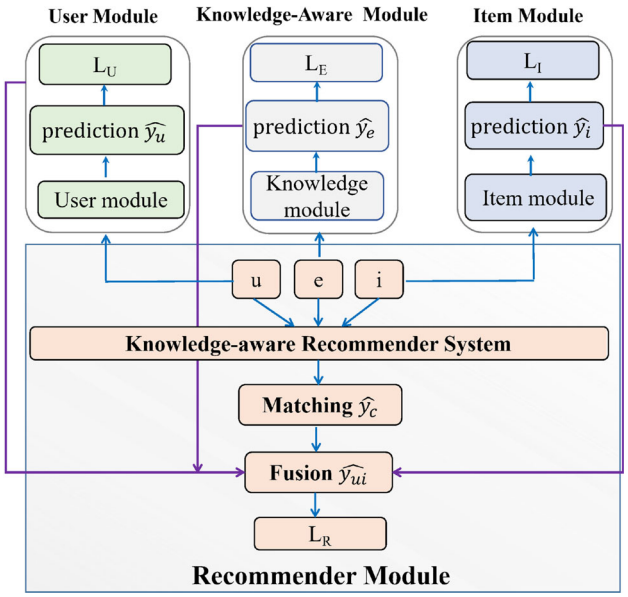


Fig. 3 The model-agnostic causal reasoning method (KUIN-Causal)

When two users are randomly recommended the same number of items, a user may click on items with higher exposure due to broader interests or herd mentality. Users with this tendency are expected to receive higher $\widehat{y_u}$ values due to the influence of popularity.

In the knowledge graph module, $\widehat{y_e} = Y_e(E = e)$ depicts how the entities and relationships in the knowledge graph impact recommendations. Popular items are likely to have denser relationships in the graph, resulting in higher scores for $\widehat{y_e}$.

Since the goal of training is to obtain the user-item score y_{ui} after removing the popularity bias, the four branches are combined into a final score as follows:

$$\hat{y_{ui}} = \hat{y_c} * \sigma\left(\hat{y_i}\right) * \sigma\left(\hat{y_u}\right) * \sigma\left(\hat{y_e}\right), \tag{19}$$

Here, the sigmoid function $\sigma(\cdot)$ is used to transform the values of $\hat{y_i}$, $\hat{y_u}$, and $\hat{y_e}$ to the range [0, 1] and adjust the final score of the original recommendation model. To construct the loss function of the KeCAIN model, we follow a multi-task training pattern:

$$L = L_R + \alpha * L_I + \beta * L_U + \gamma * L_E, \tag{20}$$

$$\begin{aligned} L_U = \sum & -y_{ui} \log\left(\sigma\left(\hat{y_u}\right)\right) \\ & - (1 - y_{ui}) \log\left(1 - \sigma\left(\hat{y_u}\right)\right), \end{aligned} \tag{21}$$

$$\begin{aligned} L_I = \sum & -y_{ui} \log\left(\sigma\left(\hat{y_i}\right)\right) \\ & - (1 - y_{ui}) \log\left(1 - \sigma\left(\hat{y_i}\right)\right), \end{aligned} \tag{22}$$

$$\begin{aligned} L_E = \sum & -y_{ui} \log\left(\sigma\left(\hat{y_e}\right)\right) \\ & - (1 - y_{ui}) \log\left(1 - \sigma\left(\hat{y_e}\right)\right), \end{aligned} \tag{23}$$

The loss of the KGUIN module is represented by L_R, while α, β, and γ are the three adjustable hyperparameters.

In the testing phase, the key to eliminating the popularity bias through counterfactual reasoning of the popularity analysis is to remove the influence of the item factor on the final score $\hat{y_{ui}}$ through the path $I \rightarrow Y$. Taking NDE as an example, this direct effect signifies the immediate impact on outcome Y from changes in the UI (User Interface) state, provided that the mediating variable K remains unchanged and only direct pathways affect Y. Given that K actually depends on UI, the assumption that "K remains constant when UI changes" constructs a hypothetical scenario where "user behavior is influenced only by the items and the user themselves, not by the match between the items and the user." The indirect effect, TIE, exclusively retains the influence of the match between users and items on user behavior, effectively removing bias. This can be achieved by subtracting $c \times \sigma(\hat{y_i}) \times \sigma(\hat{y_u}) \times \sigma(\hat{y_e})$ from the original formula:

$$\hat{y_c} * \sigma\left(\hat{y_i}\right) * \sigma\left(\hat{y_u}\right) * \sigma\left(\hat{y_e}\right) - c * \sigma\left(\hat{y_i}\right) * \sigma\left(\hat{y_u}\right) * \sigma\left(\hat{y_e}\right). \tag{24}$$

KeCAIN, the model introduced in this paper, stands for Knowledge Graph-guided Recommendation System, which integrates causal analysis to mitigate bias. KGUIN, short for Knowledge Graph User Intent Network, models the relationships between user intents and items using information propagation based on the knowledge graph. KGUIN-Causal applies a causal analysis module to the KGUIN framework.

Experimental Analysis

To demonstrate the effectiveness of the KeCAIN algorithm, this paper presents experimental results from various perspectives.

Dataset

This paper verifies the performance of the KeCAIN algorithm using three datasets: Amazon-Book [10], Last-FM [36], and Alibaba-iFashion [36]. The statistics for each dataset are shown in Table 1.

Evaluation Metrics

To quantify the recommendation performance of the proposed method, Recall@N [37] and NDCG@N [38] measures are utilized, with N set to 20 for top-N recommendations.

Table 1 Statistics of datasets

		Amazon-Book	Last-FM	Alibaba-iFashion
User-item interaction	Number of users	70,679	23,566	114,737
	Number of items	24,915	48,123	30,040
	Number of interactions	847,733	3,034,796	1,781,093
Knowledge Graph	Number of Entities	88,572	58,266	59,156
	Number of relations	39	9	51
	Number of Triples	2,557,746	464,567	279,155

Table 2 The overall performance evaluation results for KeCAIN and baselines on three datasets. The best and second-best performances are highlighted in bold and borderline, respectively

	Amazon-Book		Last-FM		Alibaba-iFashion	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
MF	0.1300	0.0678	0.0724	0.0617	0.1095	0.0670
CKE	0.1342	0.0698	0.0732	0.0630	0.1103	0.0676
LightGCN	0.1306	0.0688	0.0756	0.0677	0.1028	0.0638
KGAT	0.1487	0.0799	0.0873	0.0744	0.103	0.0627
KGNN-LS	0.1362	0.0560	0.0880	0.0642	0.1039	0.0557
CKAN	0.1442	0.0698	0.0812	0.0660	0.0970	0.0509
R-GCN	0.1220	0.0646	0.0743	0.0631	0.096	0.0515
KGIN	0.1687	0.0915	0.0968	0.0831	0.1144	0.0712
NCL	0.1381	0.0815	–	–	0.0713	0.0319
KGCL	–	–	0.0905	0.0769	0.1148	0.0743
MCCLK	–	–	0.0671	0.0603	0.1089	0.0707
KGRec	–	–	0.0943	0.0810	0.1188	0.0743
AdaMCL	0.1620	0.1012	–	–	0.0917	0.0423
CausE	0.1254	0.0683	0.0773	0.0675	0.1036	0.0647
IPW	0.1617	0.0881	0.1119	0.0978	0.1279	0.0799
DICE	0.1754	0.0951	0.1125	0.0983	0.1286	0.0804
KeCAIN	**0.1895**	**0.1093**	**0.1187**	**0.1038**	**0.1372**	**0.0866**
%Improv	7.44%	7.41%	5.22%	5.30%	6.27%	7.16%

Comparison with Baselines

To verify the efficacy of our proposed design, we compare the KeCAIN with baselines, including collaborative filtering methods: MF [39], knowledge-aware embedding-based method: CKE [40], LightGCN [41], KGAT [2], KGNN-LS [42], CKAN [43], R-GCN [44], NCL [45], and KGIN [10], KGCL [46], MCCLK [47], KGRec [36], AdaMCL [48], and cause-effect methods: CausE [15], IPW [35], and DICE [16].

Overall Performance Comparison

In Table 2, we present the results of all the approaches on three datasets. The proposed KeCAIN consistently surpasses all baseline models. There are three reasons for this. Firstly, knowledge and preference information that is helpful for recommendation tasks is captured via thoughtful encoding of user intent information. Second, to eliminate the impact on recommendation effectiveness caused by three confounding factors in counterfactual reasoning-the user's cognitive information, item exposure information, and information on the densities of entities and relationships in the knowledge graph-KeCAIN is equipped with causal effect computation for causal reasoning. Third, multi-perspective information characterization and information distribution of knowledge aggregation layer and user intended use. The proposed knowledge-focused user intent inference network automatically finds useful information linked to the user's intent and improves the model recommendation effect, as demonstrated by excellent results on datasets with diverse statistics. Specifically, on the Last-FM and Alibaba-iFashion datasets, the Recall@20 metric improves by 5.22% and 6.27%, respectively, compared to the DICE algorithm. On the Amazon-Book dataset, the metric improves by 7.44% over the DICE algorithm, which demonstrates the effectiveness of the proposed method.

Impact of Hyperparameters

In this section, we will explore the impact of different hyperparameters on the recommendation performance of KeCAIN, starting with the specific configuration of hyperparameters. Specifically, we will compare the number of layers for graph convolution information aggregation, the number of user intents, and c.

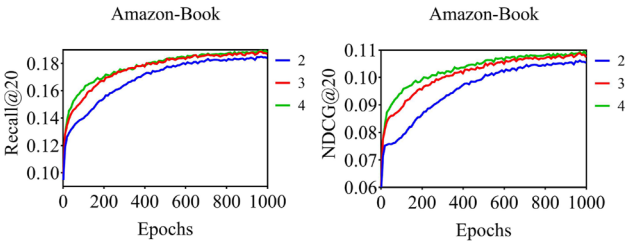


Fig. 4 Comparison of KeCAIN performance under different graph convolution layers

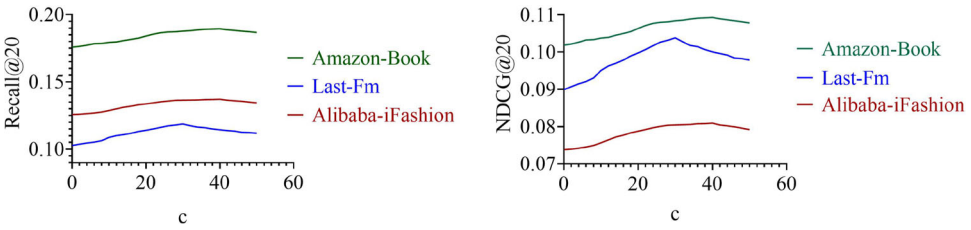


Fig. 5 Impact of hyperparameter c on recommended performance

Comparison of the Number of Graph Convolution Layer

The number of hops for graph convolution information aggregation is a crucial hyperparameter for the KeCAIN algorithm, as it determines the depth of information aggregation. The performance of the KeCAIN model for variable numbers of graph convolution information aggregation layers is shown in Fig. 4 for Amazon-Book datasets. The recommendation impact continues to get better when the graph neural network's number of information propagation layers rises, as Fig. 4 illustrates.

Effect of the Hyperparameter c

The hyperparameter c, which appears in (24) and controls the weight of the original recommendation module in the final rating, was studied to investigate its effect on the proposed model. To achieve this, c values were selected within the range of 0 to 50.

From Fig. 5, it is evident that the recommendation performance is at its lowest when the c value is 0, indicating that all model scores are derived solely from the original recommendation module. As the c value increases, the counterfactual reasoning gradually increases its contribution to the model scoring, removing the popularity bias in the recommendation process, and thus the model's performance gradually improves. However, the model's performance steadily declines as the c value increases to a certain point, indicating that counterfactual reasoning does not always enhance the model's effectiveness as a recommendation predictor. To enhance the final recommendation performance, the c value should be appropriately increased. Furthermore, it can be observed that the value of the hyperparameter c varies when the recommendation performance reaches its peak on these three datasets. For the Amazon-Book dataset, c is 40, for the Last-FM dataset, it is 30, and for the Alibaba-iFashion dataset, it is 38.

Cognitive Analysis of User Intents

The number of user cognitive intents is also a factor that affects the model recommendation effect. To explore the impact of user intentions on the model, this section will hide several of these user intents during training. The hidden user intent will still participate in the update of network parameters but will not participate in the weight calculation when the item node is finally aggregated to the user node. User intentions participate in the computation of attention, so their number is at least 2.

To further analyze the impact of masked user cognitive intentions, we extract 10 users from the training results and generate a heatmap based on the numerical distribution of

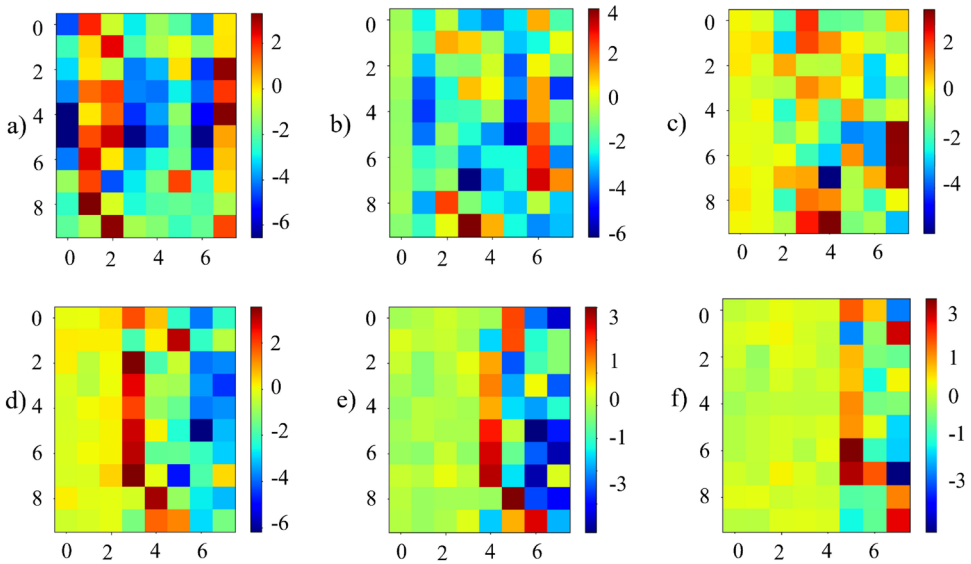


Fig. 6 Example of a heatmap of users interacting with cognitive intents under the Last-FM dataset

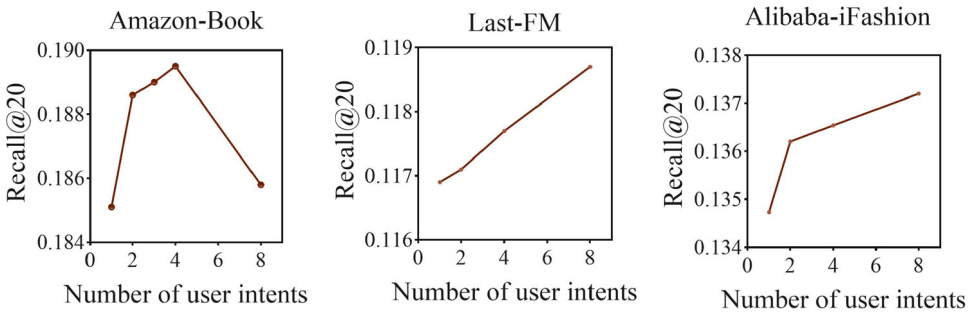


Fig. 7 Impact of user intent number on recommendation performance

Table 3 Impact of different counterfactual modules on the proposed model

	Amazon-Book		Last-FM		Alibaba-iFashion	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
w/o item module	0.1716	0.0931	0.0986	0.0861	0.1190	0.0744
w/o user module	0.1820	0.0988	0.1123	0.0981	0.1287	0.0804
w/o KG module	0.1788	0.0970	0.1106	0.0967	0.1230	0.0769
KeCAIN	0.1895	0.1093	0.1187	0.1038	0.1372	0.0866

the user intentions for each user. The y-axis represents the user, and the x-axis represents the weight of the masked user's cognitive intentions. The heatmap is shown in Fig. 6. (a) is the unmasked user intent, and (b), (c), (d), (e), and (f) denote the heat maps for masking one to five user intents respectively. As can be seen from the figure, after masking some of the user's intentions, the rest of the user's intentions are also changed, illustrating the changing perceptions of the user during their interaction with the item. Figure 7 illustrates the impact of the number of user intentions on the performance of the recommender system. From the figure, we can see that for different datasets, as the number of user intents increases, the performance of the recommender system decreases, which can also be reflected in the situation, when the user has too many intents when purchasing items, there will be "difficult to choose."

Effect of Different Modules

In the counterfactual module of the KeCAIN algorithm, there are three branching modules for multi-task training: the item module, the user module, and the knowledge graph module.

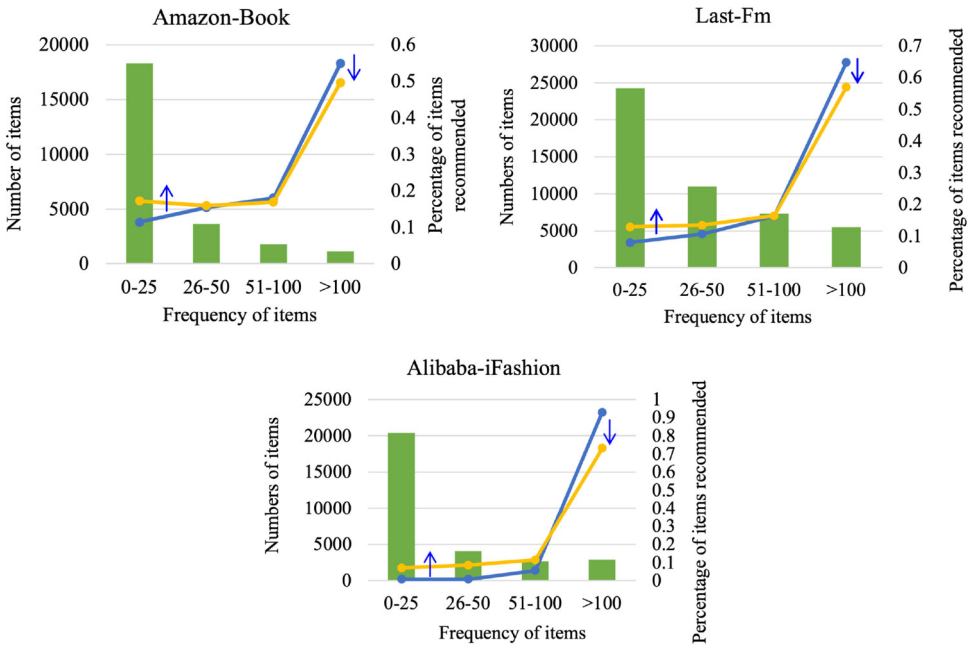


Fig. 8 Comparison before and after removal of popularity bias

Table 4 Comparing results without ic and cr module

	Amazon-Book[1]		Last-FM[2]		Alibaba-iFashion[1]	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
KeCAIN w/o cr	0.1700	0.0926	0.0974	0.0974	0.1181	0.0738
KeCAIN w/o ic	0.1837	0.1001	0.1133	0.0991	0.1295	0.0809
KeCAIN	0.1895	0.1093	0.1187	0.1038	0.1372	0.0866

This subsection will examine each of these modules separately, by removing them one at a time and showing the impact on recommendation performance. The specific recommendation performance is presented in Table 3.

Cognitive Analysis of Popularity Bias

To further analyze the cognitive effectiveness of the KeCAIN model in eliminating popularity bias, as shown in Fig. 8. The figure shows the comparison of item frequency before and after applying the causal-reasoning module in the Amazon-Book, Last-FM, and Alibaba-iFashion datasets. Before eliminating the popularity bias, the percentage of items with a frequency greater than 100 in the recommended results was 54.9%, 64.8%, and 92.9%, respectively, while the percentage of items with frequencies of 0–25 was only 11.5%, 8.05%, and 0.75% of the total recommended items, respectively. After the elimination of popularity bias, the percentage of items with a frequency greater than 100 decreased to 49.7%, 57.0%, and 73.1%, respectively, while the percentage of items with frequencies of 0–25 and 25–50 increased to 17.3%, 13.0%, and 7.0%, respectively. This result further demonstrates the effectiveness of the causal-reasoning framework in altering the distribution of items in the recommendation results, decreasing the frequency of popular items, and eliminating the popularity bias.

Ablation Study

To verify the effectiveness of counterfactual reasoning in KeCAIN, the item category information (ic), and counterfactual reasoning module (cr) are removed in this section, and as can be seen from Table 4, counterfactual reasoning has a high effect on removing the popularity bias.

Conclusion

This paper primarily focuses on the issue of the inability to mine cognitive interaction information between user cognitive intents and item categories and a causal-reasoning module in a knowledge-aware recommendation system (KeCAIN). KeCAIN incorporates multi-view information from user intents for different kinds of items, employs multi-task learning by the causal graph to evaluate the impact of various causes on the ranking score, and utilizes counterfactual inference to eliminate popularity bias. The experiments conducted demonstrate the effectiveness of KeCAIN from multiple perspectives, and the experimental results reveal that the KeCAIN algorithm outperforms most of the baseline recommendation methods, and has strong model competitiveness.

Author Contributions H.X.: conceptualization, methodology, formal analysis, writing—original draft, writing—review and editing. J.B.: conceptualization, software, supervision, funding acquisition, resources. Q.L.: conceptualization, methodology, writing—review and editing. L.H.: methodology, formal analysis. F.C.: funding acquisition, resources, writing—review and editing.

Funding This work was supported by CAAI-Huawei MindSpore Open Fund and the Science and Technology Research Program of Chongqing Municipal Education Commission: (Grant No. KJQN202200718).

Data Availability No datasets were generated or analysed during the current study.

Declarations

Competing Interests The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Lin Q, Liu J, Mao R, et al. Techs: Temporal logical graph networks for explainable extrapolation reasoning. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers). 2023. pp. 1281–93.
2. Wang X, He X, Cao Y, et al. Kgat: knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM

SIGKDD International Conference on Knowledge Discovery & Data Mining. 2019c. pp. 950–8.
3. El-Kishky A, Markovich T, Park S, et al. Twhin: Embedding the twitter heterogeneous information network for personalized recommendation. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022. pp. 2842–50.
4. Huang C, Xu H, Xu Y, et al. Knowledge-aware coupled graph neural network for social recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2021. pp. 4115–22.
5. Xu F, Wu Z, Sun Q, et al. Symbol-llm: Towards foundational symbol-centric interface for large language models. arXiv:2311.09278 [Preprint]. 2023. Available from http://arxiv.org/abs/2311.09278.
6. Pang Y, Wu L, Shen Q, et al. Heterogeneous global graph neural networks for personalized session-based recommendation. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. 2022. pp. 775–83.
7. Yang L, Liu Z, Wang Y, et al. Large-scale personalized video game recommendation via social-aware contextualized graph neural network. In: Proceedings of the ACM Web Conference 2022. 2022a. pp. 3376–86.
8. Zhou H, Tan Q, Huang X, et al. Temporal augmented graph neural networks for session-based recommendations. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021. pp. 1798–802.
9. Hsu C, Li CT. Retagnn: Relational temporal attentive graph neural networks for holistic sequential recommendation. In: Proceedings of the Web Conference 2021. 2021. pp. 2968–79.
10. Wang X, Huang T, Wang D, et al. Learning intents behind interactions with knowledge graph for recommendation. In: Proceedings of the Web Conference 2021. 2021c. pp. 878–87.
11. Zhao Y, Wang X, Chen J, et al. Time-aware path reasoning on knowledge graph for recommendation. ACM Trans Inform Syst. 2022;41(2):1–26.
12. Liang D, Charlin L, McInerney J, et al. Modeling user exposure in recommendation. In: Proceedings of the 25th International Conference on World Wide Web. 2016b. pp. 951–61.
13. Liang D, Charlin L, Blei DM. Causal inference for recommendation. In: Causation: Foundation to Application, Workshop at UAI. AUAI; 2016a.
14. Wang W, Feng F, He X, et al. Deconfounded recommendation for alleviating bias amplification. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. 2021a. pp. 1717–25.
15. Bonner S, Vasile F. Causal embeddings for recommendation. In: Proceedings of the 12th ACM Conference on Recommender Systems. 2018. pp. 104–12.
16. Cui Z, Yu F, Wu S, et al. Disentangled item representation for recommender systems. ACM Trans Intell Syst Technol (TIST). 2021;12(2):1–20.
17. Wang W, Feng F, He X, et al. Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021b. pp. 1288–97.
18. Zhang Y, Feng F, He X, et al. Causal intervention for leveraging popularity bias in recommendation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021. pp. 11–20
19. Zheng Y, Gao C, Li X, et al. Disentangling user interest and conformity for recommendation with causal embedding. In: Proceedings of the Web Conference 2021. 2021. pp. 2980–91.
20. Mao R, Du K, Ma Y, et al. Discovering the cognition behind language: Financial metaphor analysis with metapro. In: IEEE ICDM. 2023.
21. Nápoles G, Grau I, Salgueiro Y. Recommender system using long-term cognitive networks. Knowl-Based Syst. 2020;206:106372.
22. Jain G, Mahara T, Sharma S, et al. A cognitive similarity-based measure to enhance the performance of collaborative filtering-based recommendation system. IEEE Trans Comput Soc Syst. 2022;9(6):1785–93.
23. Wan H, Yu S. A recommendation system based on an adaptive learning cognitive map model and its effects. Interact Learn Environ. 2023;31(3):1821–39.
24. Liang Y, Ponnada A, Lamere P, et al. Enabling goal-focused exploration of podcasts in interactive recommender systems. In: Proceedings of the 28th International Conference on Intelligent User Interfaces. 2023. pp. 142–55.
25. Hu B, Ye Y, Zhong Y, et al. Transmkr: Translation-based knowledge graph enhanced multi-task point-of-interest recommendation. Neurocomputing. 2022;474:107–14.
26. Hui B, Zhang L, Zhou X, et al. Personalized recommendation system based on knowledge embedding and historical behavior. Appl Intell. 2022;1–3.
27. Pham P, Nguyen LT, Nguyen NT, et al. A hierarchical fused fuzzy deep neural network with heterogeneous network embedding for recommendation. Inf Sci. 2023;620:105–24.
28. Wang H, Zhao M, Xie X, et al. Knowledge graph convolutional networks for recommender systems. In: The World Wide Web Conference. 2019a. pp. 3307–13.
29. Saito Y, Yaginuma S, Nishino Y, et al. Unbiased recommender learning from missing-not-at-random implicit feedback. In: Proceedings of the 13th International Conference on Web Search and Data Mining. 2020. pp. 501–9.
30. Gupta S, Wang H, Lipton Z, et al. Correcting exposure bias for link recommendation. In: International Conference on Machine Learning. PMLR; 2021. pp. 3953–63.
31. Wang X, Zhang R, Sun Y, et al. Combating selection biases in recommender systems with a few unbiased ratings. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining. 2021d. pp. 427–35.
32. Xiao T, Wang D. A general offline reinforcement learning framework for interactive recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2021. pp. 4512–20.
33. Xiao T, Wang S. Towards off-policy learning for ranking policies with logged feedback. In: The Thirty-Sixth AAAI Conference on Artificial Intelligence. AAAI; 2022.
34. Saito Y. Doubly robust estimator for ranking metrics with post-click conversions. In: Fourteenth ACM Conference on Recommender Systems. 2020. pp. 92–100.
35. Schnabel T, Swaminathan A, Singh A, et al. Recommendations as treatments: Debiasing learning and evaluation. In: International Conference on Machine Learning. PMLR; 2016. pp. .1670–79
36. Yang Y, Huang C, Xia L, et al. Knowledge graph self-supervised rationalization for recommendation. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2023. pp. 3046–56.
37. Chen Q, Lin J, Zhang Y, et al. Towards knowledge-based recommender dialog system. In: Inui K, Jiang J, Ng V, et al (eds) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Association for Computational Linguistics, Hong Kong, China. 2019. pp. 1803–13. https://doi.org/10.18653/v1/D19-1189, https://aclanthology.org/D19-1189.
38. Rajput S, Mehta N, Singh A, et al. Recommender systems with generative retrieval. Adv Neural Inform Process Syst. 2024;36.
39. Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer. 2009;42(8):30–7.
40. Zhang F, Yuan NJ, Lian D, et al. Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. pp. 353–62.

41. He X, Deng K, Wang X, et al. 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. 639–48.
42. Wang H, Zhang F, Zhang M, et al. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2019a. pp. 968–77.
43. Wang Z, Lin G, Tan H, et al. Ckan: Collaborative knowledge-aware attentive network for recommender systems. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2020. pp. 219–28.
44. Schlichtkrull M, Kipf TN, Bloem P, et al. Modeling relational data with graph convolutional networks. In: The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15. Springer; 2018. pp. 593–607.
45. Lin Z, Tian C, Hou Y, et al. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In: Proceedings of the ACM Web Conference 2022. 2022. pp. 2320–9.
46. Yang Y, Huang C, Xia L, et al. Knowledge graph contrastive learning for recommendation. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2022b. pp. 1434–43.
47. Zou D, Wei W, Mao XL, et al. Multi-level cross-view contrastive learning for knowledge-aware recommender system. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2022. pp. 1358–68.
48. Zhu G, Lu W, Yuan C, et al. Adamcl: Adaptive fusion multi-view contrastive learning for collaborative filtering. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2023. pp. 1076–85.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Hongcai xu[1] · Junpeng Bao[1] · Qika Lin[2] · Lifang Hou[1] · Feng Chen[3]

✉ Junpeng Bao
baojp@mail.xjtu.edu.cn

Hongcai xu
xajt1822@stu.xjtu.edu.cn

Qika Lin
linqika@nus.edu.sg

Lifang Hou
tommyhou@tencent.com

Feng Chen
chenf@cqjtu.edu.cn

[1] School of Computer Science and Technology, Xi'an Jiaotong University, West Xianning Road, Xi'an 710049 , Shaanxi, China

[2] Saw Swee Hock School of Public Health, National University of Singapore, Singapore 117549, Singapore

[3] School of Mathematics and Statistics, Chongqing Jiaotong University, Xuefu Avenue No.66, Chongqing 400074, China