electronics

MDPI

Article

Knowledge-Refined Causality-Inspired Preference Modeling for Recommender Systems

Hao Yang [1,*], Shuo Shen [1], Mingjie Yang [1], Junwei Wang [1], Yipeng Xiong [1], Wenli Wang [2] and Xiaowei Zhang [2]

[1] Digitalization Division, State Grid Gansu Electric Power Company, Lanzhou 730050, China; ss530802944@foxmail.com (S.S.); z1094741571@foxmail.com (M.Y.); waurora1005@foxmail.com (J.W.); xxdollie1125@foxmail.com (Y.X.)

[2] School of Information Science and Engineering, Lanzhou University, Lanzhou 730000, China; wwenli2024@lzu.edu.cn (W.W.); zhangxw@lzu.edu.cn (X.Z.)

* Correspondence: yanghao3648@gmail.com

Abstract

Recommender systems have become a key technology for intelligent information services by learning user preferences from large-scale behavioral data and heterogeneous knowledge sources. However, existing recommendation methods still face two critical limitations: noisy knowledge graphs may introduce misleading signals during representation learning, and it remains challenging to identify item attributes that are relevant to users' observed preferences. To address these issues, we propose CaRecK (causality-inspired recommendation with knowledge graph), a knowledge-refined preference modeling framework that combines causality-inspired attribute relevance estimation with preference-aware knowledge aggregation. Specifically, we develop an attribute selection mechanism to identify preference-relevant attributes for user representation learning. Furthermore, a preference-aware strategy is designed to adaptively select informative neighbors and integrate semantic information for learning item representations. Extensive experiments on two real-world benchmark datasets demonstrate that CaRecK achieves competitive performance in both CTR prediction and Top-*K* recommendation tasks.

Keywords: recommender systems; knowledge graphs; user preference

1. Introduction

Recommender systems have become an indispensable technique for alleviating information overload by helping users efficiently discover items that match their interests [1–3]. However, historical user–item interactions often fail to capture the rich semantic factors underlying user preferences. Consequently, knowledge graphs (KGs) have emerged as an effective source of auxiliary information by providing structured representations of entities, attributes, and relations, which have been widely exploited to enhance user and item representation learning. Despite their effectiveness, existing KGs-enhanced recommendation methods still face two fundamental challenges.

The first challenge arises from the heterogeneous and densely connected structure of knowledge graphs used for item representation learning. Although publicly available KGs provide rich semantic information, an item may be associated with numerous relation–entity pairs, and not all of these neighbors are relevant to a particular user's preference. Moreover, irrelevant or redundant connections may be propagated during multi-hop knowledge aggregation and introduce noise into the learned item representations. Existing methods [4,5] typically aggregate KG neighbors using attention mechanisms, but their

Check for updates

Academic Editor: Xianzhi Wang

Received: 20 August 2026
Revised: 14 October 2026
Accepted: 21 October 2026
Published: 24 October 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

aggregation strategies generally operate on a fixed neighborhood. Therefore, they do not explicitly adapt the selected knowledge to the preferences of each user.

The second challenge is to account for differences in the relevance of item attributes when modeling user preferences [6]. Existing recommender systems typically infer user interests by aggregating attributes from historically interacted items [7,8]. However, the prevalence of an attribute among a user's historically interacted items does not necessarily indicate its relevance to that user's preferences. For example, a user who frequently interacts with fantasy movies may be associated with attributes such as "Fantasy", "Adventure", and "English". While "Fantasy" and "Adventure" may directly reflect the user's interests, "English" may merely be a common characteristic of the interacted items. Most methods tend to assign high importance to frequently observed attributes, thereby incorporating less informative signals into user representations. Although causal recommendation methods [9–11] attempt to distinguish causal effects from observational interactions, they focus on user–item interactions and underexploit attribute-level knowledge in KG.

To address the above challenges, we propose CaRecK, a knowledge-refined causality-inspired preference modeling method for recommender systems. Specifically, a causality-inspired user preference learning module is developed to distinguish preference-relevant attributes by estimating attribute relevance. Furthermore, a preference-aware learning strategy adaptively selects informative KG neighbors and integrates heterogeneous semantic information to learn more expressive item representations. Finally, extensive experiments on two real-world benchmark datasets, Book-Crossing and Last.FM, validate the effectiveness of CaRecK in improving recommendation performance.

The main contributions of our framework are as follows:

- We propose a causality-inspired user preference module that identifies preference-relevant attributes from observed interaction patterns.
- We design a preference-aware item representation module that selects user-relevant KG neighbors and integrates their semantics through hybrid aggregation.
- Extensive experiments on two real-world datasets demonstrate that CaRecK outperforms state-of-the-art methods.

The remainder of this paper is organized as follows. Section 2 reviews the related work on knowledge graph-enhanced recommendation and causal inference modeling. Section 3 presents the preliminaries. Section 4 provides a detailed description of the proposed CaRecK framework. Section 5 reports the experimental results and analysis. Finally, Section 6 concludes the study and discusses future research directions.

2. Related Work

2.1. Knowledge Graphs-Enhanced Recommendation

Knowledge graph-enhanced recommendation incorporates entities, relations, and attributes as auxiliary information to enrich user and item representations [12]. KGIN [7] models latent user intentions as combinations of KG relations and performs intent-aware knowledge aggregation, while KGCL [13] improves representation robustness through contrastive learning over perturbed knowledge graphs. Recent studies have increasingly focused on the reliability and relevance of knowledge incorporated into recommendations. MKGCN [4] improves KG-based recommendation by constructing multiple KG views through random sampling to reduce noisy interactions. EditKG [14] solves the knowledge imbalance problem via editing KGs. These studies indicate that indiscriminate KG aggregation may introduce irrelevant or unreliable information.

2.2. Causal Inference-Inspired Recommendation

Causal inference has been introduced into recommender systems to alleviate exposure, popularity, and conformity biases [10]. KGCR [15] implements deconfounded user preference learning to resolve the negative impacts of structural information and adopt counterfactual inference to eliminate bias in user–item similarity scoring. CausalEPP [16] models the causal relationships among item quality, local popularity, and users' evolving popularity preferences, and employs backdoor adjustment and inference-time intervention for popularity debiasing. CSRec [17] separates user preference from user decisions and explicitly models the system's influence on users. Unlike existing approaches that rely on explicit causal graphs, our method uses a causality-inspired strategy to estimate item attribute relevance from observed interactions and incorporates the result into user representation learning.

3. Preliminaries

Problem Formulation

Let $\mathcal{U}$ and $\mathcal{V}$ denote the sets of users and items, respectively. The observed user–item interactions are represented by $\mathcal{Y} = y_{uv} \mid u \in \mathcal{U}, v \in \mathcal{V}$, where $y_{uv} = 1$ indicates an observed interaction between user u and item v, and $y_{uv} = 0$ otherwise [18]. In addition, a knowledge graph is defined as $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{S})$, where $\mathcal{E}$ and $\mathcal{R}$ denote the entity and relation sets, respectively, and $\mathcal{S} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is the set of knowledge triples.

Given the interaction data $\mathcal{Y}$ and the knowledge graph $\mathcal{G}$, the objective is to learn a recommendation function:

$$\hat{y}_{uv} = g(u, v \mid \mathcal{Y}, \mathcal{G}), \tag{1}$$

where $\hat{y}_{uv}$ denotes the predicted preference of user u for item v. The predicted scores are further used either to estimate user–item interaction probabilities in the CTR prediction task or to rank candidate items and generate a Top-K recommendation list for each user.

4. Methodology

We propose CaRecK, a knowledge-refined causality-inspired preference modeling method for recommender systems, as illustrated in Figure 1. CaRecK contains three modules: knowledge graph alignment and preprocessing, causality-inspired user representation learning, and preference-aware item representation learning.

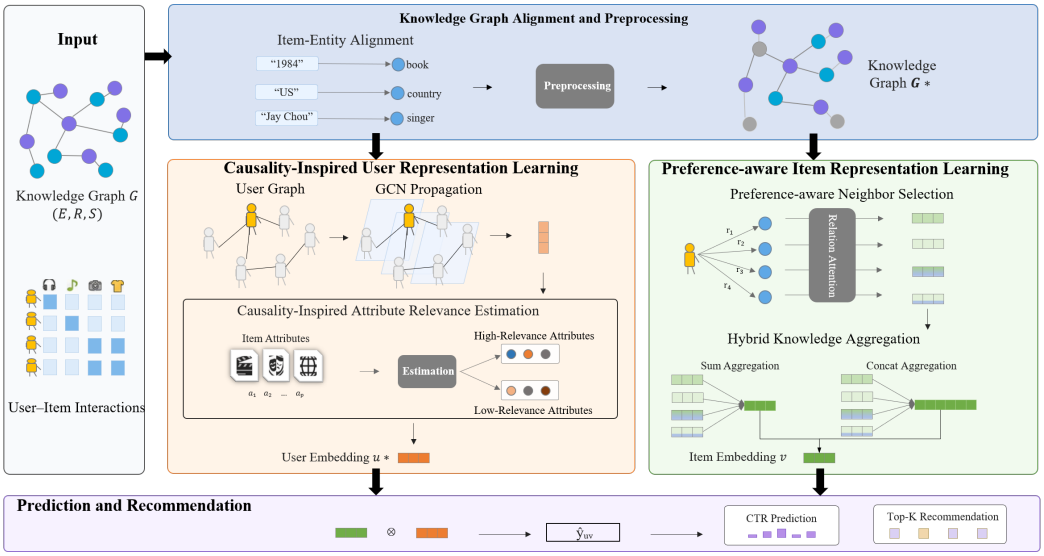


Figure 1. Illustration of the proposed CaRecK framework.

4.1. Knowledge Graph Alignment and Preprocessing

We use the publicly released knowledge graph triples and item–entity mapping files for Book-Crossing and Last.FM [19,20]. The entity-alignment procedures reported in the source studies are summarized as follows.

Specifically, for Book-Crossing, the source study matches book titles to the tails of $(head, book.book.title, tail)$ triples. For Last.FM, the source study matches musician names to the tails of $(head, type.object.name, tail)$ triples. In each case, the head entity *head* of a matching name triple is assigned to the corresponding item; items with no match or multiple matches are excluded. The item–entity mapping is denoted by $\mu : \mathcal{V}_{\text{map}} \to \mathcal{E}$, where $\mathcal{V}_{\text{map}} \subseteq \mathcal{V}$ is the set of items with valid entity identifiers:

$$\mathcal{V}_{\text{map}} = \{v \in \mathcal{V} \mid \mu(v) \text{ is defined}\}. \tag{2}$$

For each aligned item v, let $e_v = \mu(v)$ denote its corresponding KG entity. We use the distinct tail entities of KG triples as the attributes of each aligned item. For $v \in \mathcal{V}_{\text{map}}$, the attribute set is defined as follows:

$$\mathcal{A}(v) = \{e_i \in \mathcal{E} \mid \exists r_i \in \mathcal{R},\ (e_v, r_i, e_i) \in \mathcal{S}\}. \tag{3}$$

The numbers of unique entities, relations, and triples are computed directly from the released knowledge graph files and are reported in the dataset statistics table.

4.2. Causality-Inspired User Representation Learning

User preferences are influenced by both interaction behaviors and item attributes [21]. To learn user representations, we first construct a user graph to capture collaborative information among behaviorally similar users [22]. Then, a causality-inspired strategy is introduced to estimate attribute relevance from observed interactions.

4.2.1. User Graph Construction

To capture collaborative preference patterns, we construct a weighted user graph from historical user–item interactions. Let $\mathcal{U} = \{u_1, \ldots, u_m\}$ and $\mathcal{V} = \{v_1, \ldots, v_n\}$ denote the sets of users and items, respectively. Let $\mathbf{y}_i \in \{0,1\}^n$ denote the interaction vector of user u_i.

The user graph is defined as $\mathcal{G}_u = (\mathcal{U}, \mathcal{E}_u, \mathbf{W}_u)$, where $\mathcal{E}_u \subseteq \mathcal{U} \times \mathcal{U}$ is the set of user–user edges and $\mathbf{W}_u \in \mathbb{R}^{m \times m}$ is the weighted adjacency matrix. For two distinct users u_i and u_j, their collaborative similarity is measured by considering both the number of commonly interacted items and the cosine similarity between their interaction vectors:

$$s(u_i, u_j) = \sum_{v \in \mathcal{V}} \mathbb{I}(y_{i,v} = 1)\mathbb{I}(y_{j,v} = 1), \tag{4}$$

$$W_{u,ij} = \frac{s(u_i, u_j)}{\max\limits_{1 \le p < q \le m} s(u_p, u_q)} \cdot \frac{\mathbf{y}_i^{\mathsf{T}} \mathbf{y}_j}{\|\mathbf{y}_i\|_2 \|\mathbf{y}_j\|_2}, \qquad i \neq j, \tag{5}$$

where $\mathbb{I}(\cdot)$ denotes the indicator function, $s(u_i, u_j)$ represents the number of items jointly interacted with by users u_i and u_j, and $W_{u,ij}$ denotes the corresponding edge weight. In Equation (5), p and q index two distinct users in $\mathcal{U}$, and the denominator normalizes the common-interaction count by the largest value observed among all distinct user pairs. If either user has no observed interactions, the corresponding edge weight is set to zero. If no pair of users shares an interacted item, all off-diagonal edge weights are set to zero.

To preserve each user's own information during graph propagation, self-loops are added to the user graph. Let $\mathbf{D}_u \in \mathbb{R}^{m\times m}$ be the diagonal degree matrix after adding self-loops, and let $\mathbf{P}_u \in \mathbb{R}^{m\times m}$ be the corresponding normalized propagation matrix:

$$D_{u,ii} = 1 + \sum_{j=1}^{m} W_{u,ij}, \tag{6}$$

$$\mathbf{P}_u = \mathbf{D}_u^{-\frac{1}{2}}(\mathbf{W}_u + \mathbf{I}_m)\mathbf{D}_u^{-\frac{1}{2}}, \tag{7}$$

where $\mathbf{I}_m \in \mathbb{R}^{m\times m}$ is the identity matrix. The additive term 1 in $D_{u,ii}$ arises from the unit-weight self-loop introduced for each user node.

Let $\mathbf{E}_u^{(0)} \in \mathbb{R}^{m\times d}$ denote the initial user embedding matrix, where d is the embedding dimension. Each row of $\mathbf{E}_u^{(0)}$ corresponds to one user in $\mathcal{U}$. The matrix is initialized using Xavier uniform initialization and is jointly optimized with the other trainable parameters during model training. At propagation layer t, let $\mathbf{e}_i^{(t)} \in \mathbb{R}^d$ denote the representation of user u_i, and let $\mathbf{E}_u^{(t)}$ be the matrix whose rows consist of the user representations $\mathbf{e}_1^{(t)}, \ldots, \mathbf{e}_m^{(t)}$. The user representations are propagated over the user graph as follows:

$$\mathbf{E}_u^{(t)} = \mathbf{P}_u \mathbf{E}_u^{(t-1)}, \qquad t = 1, \ldots, T. \tag{8}$$

For a target user $u = u_i$, its collaborative representation $\mathbf{u} \in \mathbb{R}^d$ is obtained by averaging $\mathbf{e}_i^{(t)}$ over the initial layer and all propagation layers:

$$\mathbf{u} = \frac{1}{T+1} \sum_{t=0}^{T} \mathbf{e}_i^{(t)}. \tag{9}$$

The collaborative representation $\mathbf{u}$ is used as the user representation in the subsequent preference-aware neighbor selection module.

4.2.2. Causality-Inspired Attribute Relevance Estimation

Although collaborative filtering can capture similarity patterns among users, it does not explicitly identify the item attributes associated with a user's preference. Therefore, we introduce a causality-inspired attribute relevance estimation mechanism. The purpose of this mechanism is to compare the observed interaction rates of items with and without a particular attribute under a common candidate item set. Accordingly, the resulting score is interpreted as an observational attribute relevance score rather than as a formally identified causal effect.

Let $\mathcal{A}(v) \subseteq \mathcal{A}$ denote the set of attributes associated with item v and $\mathcal{O}^+ = \{(u, v) \in \mathcal{U} \times \mathcal{V} \mid y_{uv} = 1\}$ denote the observed positive user–item pairs in the training set. For each positive pair, unobserved pairs satisfying $y_{uv} = 0$ are sampled as negative training instances, and the resulting set is denoted by $\mathcal{O}^-$. For each user u, we then construct the candidate item set $\mathcal{C}_u = \mathcal{C}_u^+ \cup \mathcal{C}_u^-$, where $\mathcal{C}_u^+ = \{v \in \mathcal{V} \mid (u, v) \in \mathcal{O}^+\}$ and $\mathcal{C}_u^- = \{v \in \mathcal{V} \mid (u, v) \in \mathcal{O}^-\}$. Accordingly, $y_{uv} = 1$ for $v \in \mathcal{C}_u^+$ and $y_{uv} = 0$ for $v \in \mathcal{C}_u^-$. For a given attribute a, the candidate item set $\mathcal{C}_u$ is divided into two subsets according to whether the item contains attribute a:

$$\mathcal{C}_{u,a}^{(1)} = \{v \in \mathcal{C}_u \mid a \in \mathcal{A}(v)\}, \tag{10}$$

$$\mathcal{C}_{u,a}^{(0)} = \{v \in \mathcal{C}_u \mid a \notin \mathcal{A}(v)\}. \tag{11}$$

We define the valid attribute set for user u as follows:

$$\mathcal{A}_u^{\text{valid}} = \left\{a \in \mathcal{A} \mid |\mathcal{C}_{u,a}^{(1)}| > 0 \text{ and } |\mathcal{C}_{u,a}^{(0)}| > 0\right\}. \tag{12}$$

For each $a \in \mathcal{A}_u^{\text{valid}}$, the attribute relevance score is computed as the difference between the two observed interaction rates:

$$\Delta_{u,a} = \frac{1}{|\mathcal{C}_{u,a}^{(1)}|} \sum_{v \in \mathcal{C}_{u,a}^{(1)}} y_{uv} - \frac{1}{|\mathcal{C}_{u,a}^{(0)}|} \sum_{v \in \mathcal{C}_{u,a}^{(0)}} y_{uv}. \tag{13}$$

The first term in Equation (13) represents the mean observed interaction rate of candidate items containing attribute a, whereas the second term represents the corresponding mean interaction rate of candidate items without attribute a. Therefore, $\Delta_{u,a} > 0$ indicates that user u interacts more frequently with items containing attribute a, while $\Delta_{u,a} < 0$ indicates the opposite pattern.

For the attribute a, the relevance score is first mapped to the interval $(0,1)$ using the sigmoid function. The result is then applied to the attribute embedding to obtain its weighted representation:

$$w_{u,a} = \sigma(\Delta_{u,a}) = \frac{1}{1 + \exp(-\Delta_{u,a})}, \tag{14}$$

$$\boldsymbol{\tau}_{u,a} = w_{u,a}\mathbf{a}, \tag{15}$$

where $\mathbf{a} \in \mathbb{R}^d$ denotes the embedding of attribute a, and $\boldsymbol{\tau}_{u,a} \in \mathbb{R}^d$ denotes the corresponding weighted attribute representation for user u.

Finally, at most K_a, attributes with the largest relevance weights are selected to construct the user-specific attribute set:

$$A_u = \underset{a \in \mathcal{A}_u^{\text{valid}}}{\arg \operatorname{topK}}(w_{u,a}). \tag{16}$$

If $|\mathcal{A}_u^{\text{valid}}| < K_a$, all valid attributes are retained. If $\mathcal{A}_u^{\text{valid}} = \varnothing$, then $A_u = \varnothing$, and the subsequent user representation is constructed from the collaborative user representation alone.

4.2.3. User Representation Fusion

After obtaining the user-specific attribute set A_u, we integrate the weighted attribute representations with the collaborative user representation. For each selected attribute $a \in A_u$, the collaborative representation $\mathbf{u}$ and the weighted attribute representation $\boldsymbol{\tau}_{u,a}$ are first combined and then transformed through a nonlinear projection:

$$\mathbf{u}^* = \frac{1}{|\mathcal{A}_u|} \sum_{a \in \mathcal{A}_u} \text{ReLU}(\mathbf{W}_\tau(\mathbf{u} + \boldsymbol{\tau}_{u,a}) + \mathbf{b}_\tau), \qquad |\mathcal{A}_u| > 0, \tag{17}$$

where $\mathbf{u}^* \in \mathbb{R}^d$ denotes the final user representation, $\mathbf{W}_\tau \in \mathbb{R}^{d \times d}$ is a trainable transformation matrix, and $\mathbf{b}_\tau \in \mathbb{R}^d$ is a trainable bias vector. If $A_u = \varnothing$, we set $\mathbf{u}^* = \mathbf{u}$.

4.3. Preference-Aware Item Representation Learning

4.3.1. Preference-Aware Neighbor Selection

Knowledge graphs provide rich semantic information for item representation learning. However, different users may focus on different semantic aspects of the same item. Therefore, aggregating all neighboring entities without considering user preferences may introduce irrelevant semantic information. To address this issue, we design a preference-aware neighbor selection mechanism to identify user-relevant relation–entity pairs from the knowledge graph.

After entity linking, each item $v \in \mathcal{V}$ is associated with a knowledge graph entity $e_v \in \mathcal{E}$. The one-hop neighborhood of the entity corresponding to item v is defined as follows:

$$\mathcal{N}(e_v) = \{(r_i, e_i) \in \mathcal{R} \times \mathcal{E} \mid (e_v, r_i, e_i) \in \mathcal{S}\}. \tag{18}$$

For a user u, the relevance of each relation–entity pair $(r_i, e_i) \in \mathcal{N}(e_v)$ is measured by the compatibility between the user embedding $\mathbf{u}$ and the relation embedding $\mathbf{r}_i$. Based on these relevance scores, at most, K_n relation–entity pairs are retained:

$$\mathcal{N}_{K_n}(e_v, u) = \underset{(r_i,e_i)\in\mathcal{N}(e_v)}{\arg\operatorname{topK}} \left(\mathbf{u}^\top \mathbf{r}_i\right). \tag{19}$$

If $|\mathcal{N}(e_v)| \leq K_n$, all relation–entity pairs in $\mathcal{N}(e_v)$ are retained. When multiple relation–entity pairs have identical relevance scores at the Top-K_n boundary, the required number of pairs is uniformly sampled from the candidates.

$$\pi_{r_i}(u \mid e_v) = \frac{\exp\left(\mathbf{u}^\top \mathbf{r}_i\right)}{\sum_{(r_j,e_j)\in\mathcal{N}_{K_n}(e_v,u)} \exp\left(\mathbf{u}^\top \mathbf{r}_j\right)}, \quad (r_i, e_i) \in \mathcal{N}_{K_n}(e_v, u). \tag{20}$$

For the item v associated with entity e_v, the aggregated user-specific neighborhood representation is denoted by $\mathbf{s}_v^{(t)}$:

$$\mathbf{s}_v^{(t)} = \sum_{(r_i,e_i)\in\mathcal{N}_{K_n}(e_v,u)} \pi_{r_i}(u \mid e_v)\, \mathbf{e}_{e_i}^{(t-1)}, \tag{21}$$

where $\mathbf{s}_v^{(t)} \in \mathbb{R}^d$ denotes the user-specific representation of the selected knowledge graph neighbors of item v at propagation layer t, and $\mathbf{e}_{e_i}^{(t-1)} \in \mathbb{R}^d$ denotes the representation of neighboring entity e_i.

4.3.2. Hybrid Knowledge Aggregation

To enhance the expressive capability of item representations, we introduce a hybrid aggregation mechanism that combines concatenation-based and sum-based transformations. Let $\mathbf{e}_v^{(0)} \in \mathbb{R}^d$ denote the initial embedding of item v, which is initialized using the Xavier uniform initialization. At propagation layer t, $\mathbf{e}_v^{(t-1)} \in \mathbb{R}^d$ denotes the item representation propagated from the previous layer, while $\mathbf{s}_v^{(t)} \in \mathbb{R}^d$ denotes the user-specific neighborhood representation of item v obtained by aggregating the selected knowledge graph neighbors according to the preference of user u. The two representations are processed through two parallel aggregation branches, which capture complementary information.

The concatenation-based branch first concatenates the two input representations and then applies a linear transformation followed by a Rectified Linear Unit (ReLU) activation:

$$\mathbf{g}_{\text{concat}}^{(t)} = \text{ReLU}\left(\mathbf{W}_1 \left(\mathbf{e}_v^{(t-1)} \| \mathbf{s}_v^{(t)}\right) + \mathbf{b}_1\right). \tag{22}$$

The sum-based branch first combines the two representations through element-wise addition and then applies a second linear transformation and ReLU activation:

$$\mathbf{g}_{\text{sum}}^{(t)} = \text{ReLU}\left(\mathbf{W}_2 \left(\mathbf{e}_v^{(t-1)} + \mathbf{s}_v^{(t)}\right) + \mathbf{b}_2\right). \tag{23}$$

The outputs of the two branches are concatenated and projected into the d-dimensional item embedding space:

$$\mathbf{e}_v^{(t)} = \mathbf{W}_p \left(\mathbf{g}_{\text{concat}}^{(t)} \| \mathbf{g}_{\text{sum}}^{(t)}\right) + \mathbf{b}_p. \tag{24}$$

After T propagation layers, the final item representation is defined as

$$\mathbf{v} = \mathbf{e}_v^{(T)}. \tag{25}$$

All item, entity, relation, and user representations are embedded in the same d-dimensional space. Accordingly, the concatenation operation $\|$ produces a $2D$ vector, whereas the element-wise sum preserves the dimension d. The dimensions of the intermediate tensors and trainable parameters are summarized in Table 1. The matrices $\mathbf{W}_1$, $\mathbf{W}_2$, and $\mathbf{W}_p$, as well as the bias vectors $\mathbf{b}_1$, $\mathbf{b}_2$, and $\mathbf{b}_p$, are shared across all propagation layers.

Table 1. Tensor dimensions and trainable parameters in item representation learning and user–attribute fusion.

Symbol	Dimension	Description	Trainable
$\mathbf{e}_v^{(0)}$	$\mathbb{R}^d$	Initial representation of item v	Yes
$\mathbf{e}_v^{(t-1)}$	$\mathbb{R}^d$	Item representation from the previous propagation layer	No
$\mathbf{s}_v^{(t)}$	$\mathbb{R}^d$	User-preference-aware representation of the selected knowledge graph neighbors of item v	No
$\mathbf{e}_v^{(t-1)} \| \mathbf{s}_v^{(t)}$	$\mathbb{R}^{2d}$	Concatenation of the item and neighborhood representations	No
$\mathbf{W}_1$	$\mathbb{R}^{d\times 2d}$	Transformation matrix of the concatenation branch	Yes
$\mathbf{b}_1$	$\mathbb{R}^d$	Bias vector of the concatenation branch	Yes
$\mathbf{g}_{\text{concat}}^{(t)}$	$\mathbb{R}^d$	Output of the concatenation branch after ReLU activation	No
$\mathbf{e}_v^{(t-1)} + \mathbf{s}_v^{(t)}$	$\mathbb{R}^d$	Element-wise sum of the item and neighborhood representations	No
$\mathbf{W}_2$	$\mathbb{R}^{d\times d}$	Transformation matrix of the sum branch	Yes
$\mathbf{b}_2$	$\mathbb{R}^d$	Bias vector of the sum branch	Yes
$\mathbf{g}_{\text{sum}}^{(t)}$	$\mathbb{R}^d$	Output of the sum branch after ReLU activation	No
$\mathbf{g}_{\text{concat}}^{(t)} \| \mathbf{g}_{\text{sum}}^{(t)}$	$\mathbb{R}^{2d}$	Concatenation of the outputs from the two aggregation branches	No
$\mathbf{W}_p$	$\mathbb{R}^{d\times 2d}$	Projection matrix for fusing the two branch outputs	Yes
$\mathbf{b}_p$	$\mathbb{R}^d$	Bias vector of the fusion projection	Yes
$\mathbf{e}_v^{(t)}$	$\mathbb{R}^d$	Updated item representation at propagation layer t	No
$\mathbf{W}_\tau$	$\mathbb{R}^{d\times d}$	Transformation matrix for fusing collaborative and attribute-aware user representations	Yes
$\mathbf{b}_\tau$	$\mathbb{R}^d$	Bias vector of the user–attribute fusion layer	Yes

4.4. Prediction and Optimization

After obtaining the final user representation $\mathbf{u}^*$ and the preference-aware item representation $\mathbf{v}$, we estimate the interaction probability between user u and item v using an inner-product prediction function:

$$\hat{y}_{uv} = g(\mathbf{u}^*, \mathbf{v}) = \sigma\Big((\mathbf{u}^*)^T \mathbf{v}\Big), \tag{26}$$

where $\hat{y}_{uv}$ denotes the predicted interaction probability and $\sigma(\cdot)$ is the sigmoid function.

For implicit-feedback recommendation, observed user–item interactions are treated as positive samples, while unobserved interactions are sampled as negative instances. Let $\mathcal{O}^+$ and $\mathcal{O}^-$ denote the sets of positive and negative user–item pairs, respectively, and $\mathcal{O} = \mathcal{O}^+ \cup \mathcal{O}^-$ denote the training set. The recommendation loss is defined by binary cross-entropy:

$$\mathcal{L}_{rec} = -\sum_{(u,v)\in\mathcal{O}} [y_{uv} \log \hat{y}_{uv} + (1 - y_{uv}) \log(1 - \hat{y}_{uv})], \tag{27}$$

where $y_{uv} \in \{0, 1\}$ denotes the ground-truth interaction label, and $\hat{y}_{uv}$ is the predicted interaction probability.

The overall objective function is formulated as follows:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda \sum_{\theta \in \Theta} \|\theta\|_F^2, \tag{28}$$

where λ controls the strength of L_2 regularization. All parameters in Θ are jointly optimized during model training, where $\Theta = \left\{ \mathbf{E}_{\mathcal{U}}^{(0)}, \mathbf{E}_{\mathcal{E}}^{(0)}, \mathbf{R}, \mathbf{A}, \mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, \mathbf{b}_2, \mathbf{W}_p, \mathbf{b}_p, \mathbf{W}_\tau, \mathbf{b}_\tau \right\}$. Specifically, $\mathbf{E}_{\mathcal{U}}^{(0)}$ is the initial user embedding matrix, $\mathbf{E}_{\mathcal{E}}^{(0)}$ is the initial entity embedding matrix, $\mathbf{R} \in \mathbb{R}^{|\mathcal{R}| \times d}$ is the relation embedding matrix, and $\mathbf{A} \in \mathbb{R}^{|\mathcal{A}| \times d}$ denotes the attribute embedding matrix. For an item v linked to entity e_v, the corresponding row of $\mathbf{E}_{\mathcal{E}}^{(0)}$ is used as its initial representation.

Algorithm 1 summarizes the complete training procedure of CaRecK. The user graph and attribute relevance scores are constructed from training interactions, while knowledge selection is performed during item representation learning.

Algorithm 1 The Learning Algorithm for CaRecK

Input: Interaction matrix $\mathbf{Y}$, preprocessed knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{S})$, item–entity mapping μ, and model hyperparameters.
Output: Prediction function $g(\mathbf{u}^*, \mathbf{v})$.
Initialize all trainable parameters using Xavier uniform initialization.
Split the observed interactions into training, validation, and test sets independently for each user. Sample training negatives from unobserved user–item pairs. Construct the user graph.
Set the maximum number of epochs to $E_{\max} = 1000$ and the patience to $P = 50$.
for epoch $= 1, \ldots, E_{\max}$ **do**
  for $t = 1, \ldots, T$ **do**
    Propagate user representations according to $\mathbf{E}_u^{(t)} = \mathbf{P}_u \mathbf{E}_u^{(t-1)}$.
  end for
  Compute the collaborative representation of each user u_i as $\mathbf{u}_i = \frac{1}{T+1} \sum_{t=0}^{T} \mathbf{e}_i^{(t)}$.
  Estimate attribute relevance from the training interactions and select at most K_a valid attributes for each user.
  Fuse the collaborative user representation with the selected attribute representations to obtain $\mathbf{u}^*$.
  for each training mini-batch $\mathcal{B}$ **do**
    for each $(u, v, y_{uv}) \in \mathcal{B}$ **do**
      Obtain the aligned entity $e_v = \mu(v)$ and $\mathbf{e}_v^{(0)}$.
      for $t = 1, \ldots, T$ **do**
        Select at most K_n relevant relation–entity pairs.
        Normalize their attention scores and aggregate the selected entity representations.
        Update $\mathbf{e}_v^{(t)}$ using hybrid knowledge aggregation.
      end for
      Set $\mathbf{v} = \mathbf{e}_v^{(T)}$ and compute $g(\mathbf{u}^*, \mathbf{v}) = \sigma((\mathbf{u}^*)^\top \mathbf{v})$.
    end for
    Compute the regularized loss and update the parameters using Adam.
  end for
  Evaluate the validation AUC and retain the parameters with the highest validation AUC.
  if the validation AUC has not improved for $P = 50$ consecutive epochs **then**
    break
  end if
end for
Return the prediction function $g(\mathbf{u}^*, \mathbf{v})$.

5. Experiment

In this section, we evaluate CaRecK on two real-world datasets to answer the following questions:

RQ1: How does CaRecK perform compared to existing baseline methods?
RQ2: How does each module contribute to the recommendation performance?
RQ3: How does the setting of parameters affect the model performance?

5.1. Experimental Settings

5.1.1. Datasets

We conduct experiments on two benchmark datasets, Book-Crossing http://bookcrossing.com (accessed on 20 May 2026) and Last.FM https://www.last.fm/ (accessed on 20 May 2026), for book and music recommendation, respectively. For Book-Crossing, we use `kg.txt` and `item_index2entity_id.txt` from the `data/book` directory of the RippleNet repository (accessed on 20 May 2026) [19]. For Last.FM, we use the corresponding files from the `data/music` directory of the KGCN repository (accessed on 20 May 2026) [20]. The repository URLs are provided in the cited studies. As shown in Table 2, Book-Crossing contains 17,860 users and 14,967 items. The knowledge graph contains 24,127 entities and 10 relationships. In the Last.FM dataset, there are 1872 users and 3846 items, and Last.FM knowledge graph contains 9366 entities and 60 relationships.

We convert the interaction feedback into binary interactions by treating each observed user–item pair as positive and divide these data sets into training set, validation set, and test set, with the proportions of 60%, 20%, and 20%, respectively. The split is performed independently for each user using seed 555. After splitting, the user graph is constructed from positive training interactions, and attribute relevance is estimated from training positives and sampled training negatives. Validation and test positives are excluded from these computations. This procedure ensures that no interaction from the validation or test set is used to construct the user graph, estimate attribute relevance, or derive user representations for model training.

Table 2. Statistics of the recommendation datasets and the associated knowledge graphs.

Statistics	Book-Crossing	Last.FM
Recommendation Dataset		
Users	17,860	1872
Items	14,967	3846
Interactions	139,746	42,346
Knowledge Graph		
Entities	24,127	9366
Relations	10	60
Triples	19,876	15,518

5.1.2. Settings

We evaluate the models on two recommendation tasks. (1) For CTR prediction, the model estimates the interaction probability of a given user–item pair, and we report AUC and F1 as evaluation metrics. For F1 computation, predicted probabilities greater than or equal to 0.5 are classified as positive. (2) For Top-K recommendation, the model ranks candidate items for each user. We evaluate the ranked lists using Precision@K and Recall@K for $K \in \{1, 2, 5, 10, 20\}$. During final test evaluation, the candidate pool contains all items except the user's positive training and validation items. For model training and CTR

evaluation, negative user–item pairs are sampled uniformly at a positive-to-negative ratio of 1:1. All observed positive pairs are excluded from the negative-sampling pool.

All hyperparameters are tuned on the validation set, and the final configurations used for evaluation are summarized in Table 3. Specifically, d denotes the embedding dimension, K_n denotes the maximum number of relation–entity pairs retained by preference-aware neighbor selection, K_a denotes the maximum number of attributes retained by causality-inspired attribute relevance estimation, T denotes the number of graph convolutional aggregation layers, λ denotes the coefficient of L_2 regularization, η denotes the learning rate, and batch size denotes the number of training instances in each mini-batch.

We optimize the proposed CaRecK model using the Adam optimizer [23]. Training proceeds for at most 1000 epochs. The validation AUC is evaluated after each epoch, and early stopping is applied when it does not improve for 50 consecutive epochs. Each model is trained five times using the same data partition, and the reported performance is averaged across these runs. CaRecK and all baseline methods use the same data partition and evaluation protocol. All experiments are conducted on an NVIDIA GeForce GTX 1650 GPU.

Table 3. Hyperparameter settings of CaRecK.

Datasets	d	K_n	K_a	T	λ	η	Batch Size
Book-Crossing	64	4	4	3	2×10^{-5}	2×10^{-4}	256
Last.FM	16	8	4	4	2×10^{-5}	5×10^{-4}	128

5.1.3. Baselines

We compare CaRecK with the following representative recommendation methods:

PER [24]: PER models multiple types of relations rather than relying on a single relation type. It integrates heterogeneous semantic information through metapath-based representations, enabling effective modeling of the connectivity between users and items. As a result, PER can fully exploit the rich semantic information contained in heterogeneous information networks.

CKE [8]: CKE jointly learns embeddings from both the user–item interaction graph and the knowledge graph. It incorporates structural, textual, and visual information into a unified collaborative filtering framework, thereby enhancing item representations with multi-modal knowledge and improving recommendation performance.

DKN [25]: DKN employs a multi-channel convolutional neural network to jointly encode textual content and knowledge graph entities. By aligning words with their corresponding entities, it captures both semantic and knowledge-aware representations of news articles, which are subsequently used to predict CTR.

RippleNet [19]: RippleNet propagates user preferences over the knowledge graph by iteratively expanding users' historical interactions. Through multi-hop preference propagation, it captures users' latent interests and effectively transfers preference signals across related entities in the knowledge graph.

KGCN [20]: KGCN aggregates high-order KG neighborhood information through graph convolution to capture structural and semantic dependencies.

LightGCN [1]: LightGCN simplifies graph convolution by retaining only neighborhood aggregation on the user–item interaction graph, enabling efficient collaborative representation learning.

CKAN [21]: This represents a state-of-the-art recommendation approach based on KGs, combining collaborative signals and knowledge associations through collaboration propagation and knowledge-aware attention.

KGIN [7]: KGIN models latent user intents as combinations of KGs' relations and performs intent-aware knowledge aggregation

KGCL [13]: KGCL improves representation robustness through contrastive learning over perturbed knowledge graphs

K-NCR [26]: K-NCR enhances recommendation by aggregating neighboring entities into item representations while employing heterogeneous attention mechanisms to refine user representations. It further models user–item interactions through a two-dimensional convolutional neural network, thereby improving recommendation accuracy.

IKGRC [27]: IKGRC models the deconfounded user representation learning in the training phase and employs counterfactuals to mitigate bias in user-item similarity scoring during the inference phase.

MKGCN [4]: The model improves KG-based recommendation by constructing multiple KG views through random sampling to reduce noisy interactions. It further introduces initial residual connections into multi-layer KG convolution to alleviate over-smoothing and capture high-order connectivity. In addition, graph self-attention is used to filter noisy signals and refine recommendation results.

5.2. Performance Comparison with Baseline (RQ1)

5.2.1. CTR Prediction

To evaluate the effectiveness of CaRecK in CTR prediction, we compare it with several representative baseline methods on the Book-Crossing and Last.FM datasets. The results are reported in Table 4, where the best results are highlighted in bold, and the second-best results are underlined.

Experimental results show that CaRecK achieves the best performance among all baseline methods. On the Book-Crossing dataset, CaRecK improves AUC and F1 by 1.60% and 1.25%, respectively. On the Last.FM dataset, the improvements in AUC and F1 reach 0.55% and 0.17%, respectively. These results demonstrate the effectiveness of CaRecK in CTR prediction.

Among the baselines, DKN shows relatively limited performance. It was originally developed for news recommendation and uses textual and entity information to represent news articles. Its design may be less well-suited to the item information available on Book-Crossing and Last.FM datasets. CKE performs better than DKN in both datasets by integrating collaborative and knowledge representations, but it does not explicitly estimate attribute relevance from observed interactions. LightGCN provides a strong collaborative filtering baseline through simplified graph convolution on the user–item interaction graph. However, because it does not incorporate external knowledge or item attribute semantics, it cannot fully exploit the structured information available in the knowledge graph.

RippleNet, KGCN, KGCL, MKGCN, IKGRC, KGIN, and CKAN use knowledge graph information in different ways. RippleNet propagates user preferences through KG paths, KGCN aggregates neighboring entities for item representation learning, and KGCL introduces contrastive learning over the interaction and knowledge graphs. MKGCN constructs multiple KG views and uses residual graph convolution, while IKGRC incorporates deconfounded learning and counterfactual inference. KGIN models users' latent intents to capture intent-aware user–item and item–item relationships, and CKAN combines collaborative signals with knowledge-aware attention. CaRecK instead estimates user-specific attribute relevance from interactions and selectively aggregates informative neighbors. These results show that CaRecK achieves the highest CTR prediction performance on Book-Crossing and remains competitive with the strongest baselines on Last.FM.

Table 4. Performance comparison with baselines on the CTR prediction task. The best and second-best results for each metric are highlighted in bold and underlined, respectively. The relative improvements over the strongest baseline are also shown in bold.

Model	Book-Crossing		Last.FM	
	AUC	F1	AUC	F1
PER	0.617 ± 0.0035	0.562 ± 0.0052	0.633 ± 0.0038	0.596 ± 0.0055
CKE	0.671 ± 0.0083	0.611 ± 0.0075	0.644 ± 0.0045	0.613 ± 0.0021
DKN	0.622 ± 0.0048	0.564 ± 0.0059	0.602 ± 0.0052	0.602 ± 0.0061
RippleNet	0.713 ± 0.0075	0.649 ± 0.0134	0.741 ± 0.0090	0.702 ± 0.0023
KGCN	0.694 ± 0.0064	0.642 ± 0.0083	0.789 ± 0.0079	0.712 ± 0.0056
LightGCN	0.702 ± 0.0036	0.601 ± 0.0025	0.814 ± 0.0031	0.733 ± 0.0034
KGCL	0.745 ± 0.0035	0.668 ± 0.0026	0.824 ± 0.0016	0.751 ± 0.0024
KGIN	0.723 ± 0.0046	0.661 ± 0.0050	0.823 ± 0.0025	0.759 ± 0.0033
CKAN	0.747 ± 0.0066	0.673 ± 0.0020	0.825 ± 0.0034	0.762 ± 0.0012
K-NCR	0.711 ± 0.0095	0.661 ± 0.0056	0.759 ± 0.0045	0.702 ± 0.0031
IKGRC	0.730 ± 0.0015	0.672 ± 0.0021	0.813 ± 0.0019	0.743 ± 0.0023
MKGCN	0.732 ± 0.0025	0.680 ± 0.0018	0.812 ± 0.0022	0.741 ± 0.0013
CaRecK	**0.7590 ± 0.0052**	**0.6885 ± 0.0036**	**0.8295 ± 0.0047**	**0.7633 ± 0.0064**
IMPR	**1.60%**	**1.25%**	**0.55%**	**0.17%**

5.2.2. Top-*K* Recommendation

To further evaluate the ranking quality of CaRecK, we conduct Top-*K* recommendation experiments using Precision@*K* and Recall@*K*. Figures 2 and 3 report the Top-*K* recommendation results on the Book-Crossing and Last.FM datasets, respectively, including Precision@*K* and Recall@*K*.

CaRecK consistently achieves the best performance under different values of *K* on both datasets. This indicates that CaRecK is not only effective in CTR prediction but also capable of generating higher-quality ranked recommendation lists. Moreover, the advantage of CaRecK remains stable as *K* increases, especially in terms of Recall@*K*. This suggests that CaRecK can retrieve more relevant items when the recommendation list becomes longer, demonstrating its robustness ability in Top-*K* recommendation.

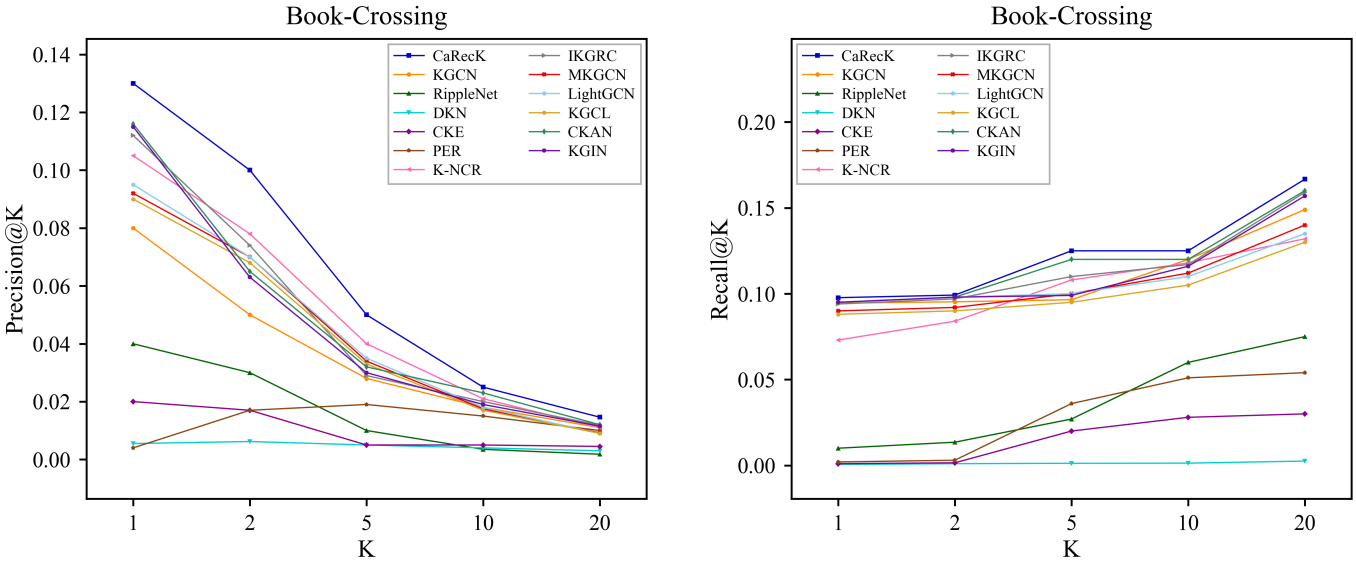


Figure 2. The results of Precision@K and Recall@K on Book-Crossing.

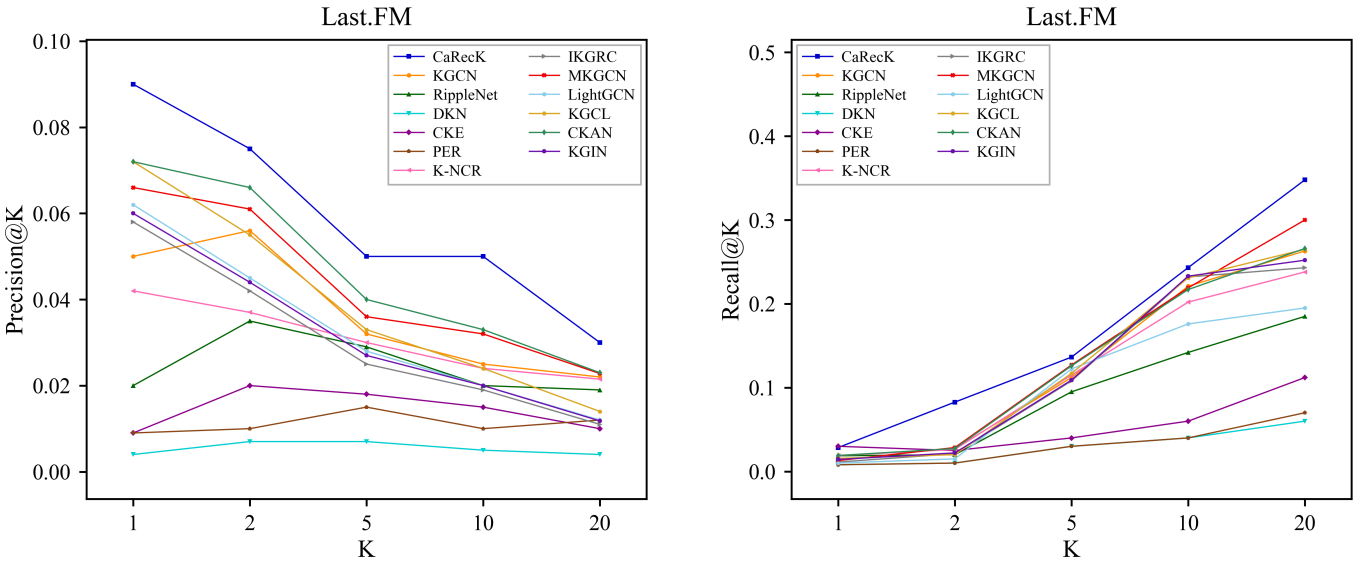


Figure 3. The results of Precision@K and Recall@K on Last.FM.

5.3. Ablation Study (RQ2)

5.3.1. Effect of Causality-Inspired Attribute Relevance Estimation Strategy

To verify the effectiveness of the proposed causality-inspired attribute estimation strategy, we conduct ablation experiments with different attribute settings in the user representation learning module. Specifically, CaRecK-att0 removes the attribute relevance estimation and directly learns user representations without incorporating attributes. CaRecK-att1 introduces one attribute for user representation learning, while CaRecK-att2 incorporates two attributes. The full CaRecK model dynamically identifies multiple preference-relevant attributes for each user and integrates them into the user representation.

Figure 4 reports the CTR prediction results on Book-Crossing and Last.FM. We observe that CaRecK-att0 generally yields the lowest AUC and F1 among the attribute variants. Introducing attribute relevance estimation improves performance, although the gain from using a single attribute is limited. Compared with CaRecK-att1, CaRecK-att2 achieves larger improvements on both datasets, suggesting that incorporating more preference-relevant attributes can provide useful information for user representation learning. The full CaRecK model achieves the best AUC and F1 on both datasets. This result supports the contribution of causality-inspired attribute estimation strategy to user representation learning.

In addition to CTR prediction, we further evaluate these variants under the Top-K recommendation setting. Figures 5 and 6 show the Precision@K and Recall@K results on Book-Crossing and Last.FM. The full CaRecK model outperforms all variants under different values of K, indicating that attribute relevance estimation not only improves interaction prediction accuracy but also enhances the ranking quality of recommendation lists.

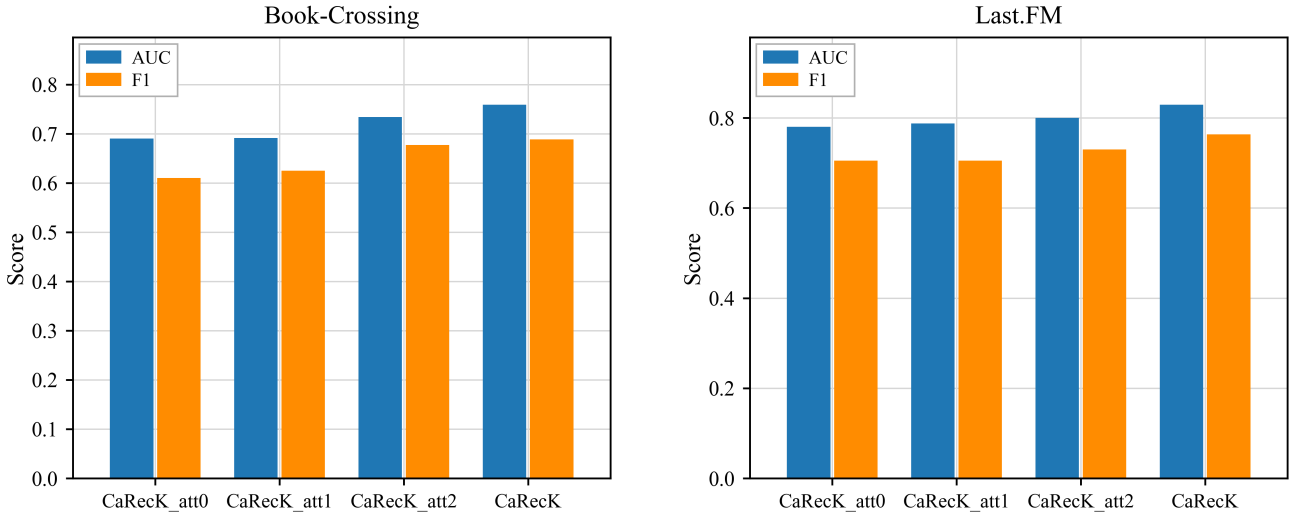


Figure 4. Experimental comparison of different causality-inspired attributes on CTR prediction task.

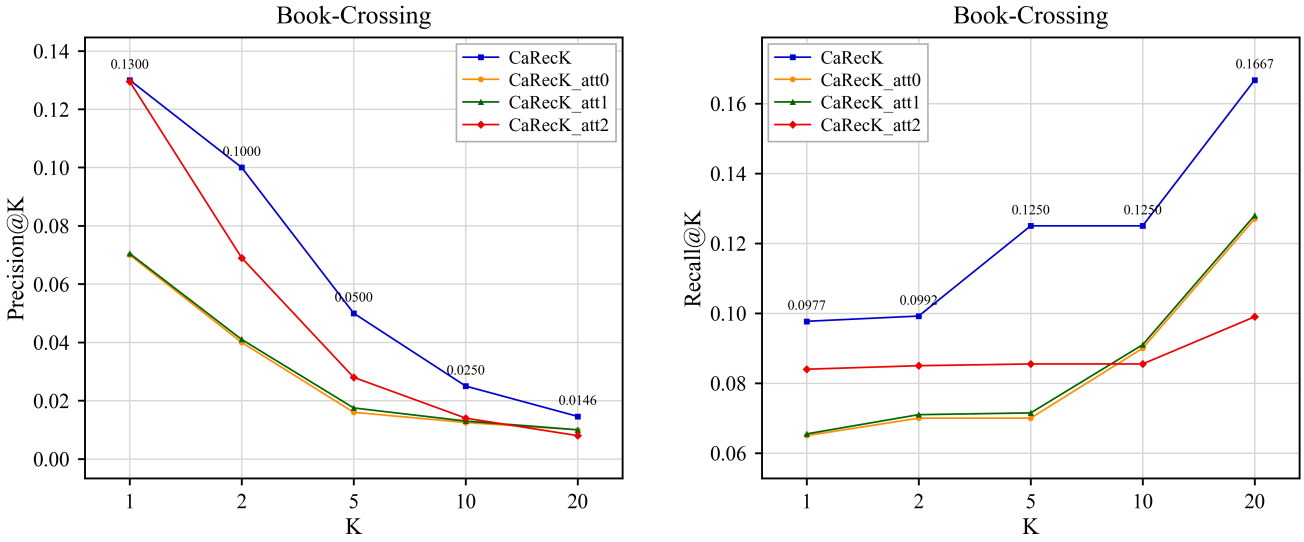


Figure 5. Precision@K and Recall@K of different causality-inspired attributes on Book-Crossing.

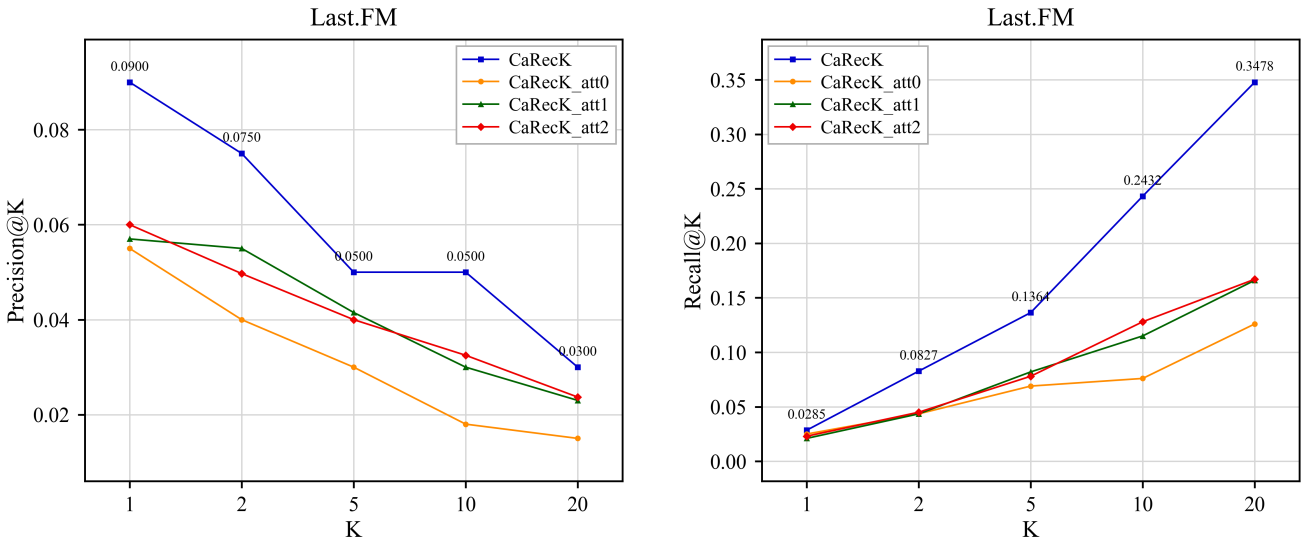


Figure 6. Precision@K and Recall@K of different causality-inspired attributes on Last.FM.

5.3.2. Effect of Different Aggregation Strategies

We further conduct ablation experiments to evaluate the effectiveness of different item aggregation strategies. As shown in Figure 7, CaRecK_sum denotes the variant that only uses the sum aggregation strategy, while CaRecK_concat denotes the variant that only uses the concatenation aggregation strategy. We evaluate these variants on both datasets under the CTR prediction setting.

The results show that CaRecK_sum consistently outperforms CaRecK_concat on both Book-Crossing and Last.FM. This indicates that sum aggregation is more effective in preserving and enhancing the original item semantics when integrating neighborhood information. However, both variants perform worse than the full CaRecK model, demonstrating that using a single aggregation strategy is insufficient to fully capture the complex semantic patterns in KG.

The superior performance of CaRecK verifies the effectiveness of the proposed hybrid aggregation design. Specifically, the sum aggregation branch strengthens item representations by incorporating preference-aware neighborhood information, while the concatenation aggregation branch preserves feature-level interactions between the target item and its neighbors. By combining these two complementary aggregation strategies, CaRecK learns more expressive item representations and achieves better CTR prediction performance.

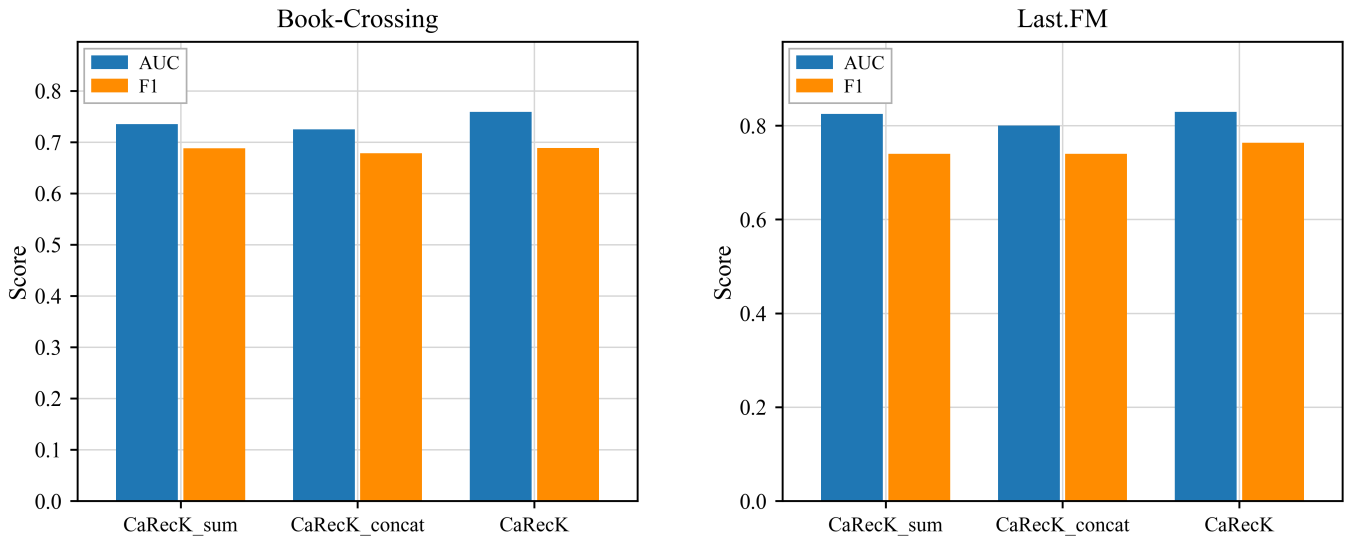


Figure 7. Ablation study of different aggregation strategies on CTR prediction.

5.3.3. Effect of User-Graph Propagation

To examine the contribution of user-graph propagation, we compare the full CaRecK model with a variant denoted as w/o User Graph. In this variant, the collaborative representation of user u_i is given by its trainable initial embedding $\mathbf{u} = \mathbf{e}_i^{(0)}$, without performing propagation over the user graph. The attribute relevance estimation, user representation fusion, and item representation learning components remain unchanged.

Table 5 reports the AUC, F1, Precision@20, and Recall@20 results on both datasets. The variant without user-graph propagation achieves lower AUC and F1 and generally lower Top-K performance than the full model. Consistent with prior work [2] on graph-based collaborative filtering, these results suggest that aggregating information from users with related interaction histories benefits both interaction prediction and item ranking.

Table 5. Effect of user-graph propagation on CTR prediction and Top@20 recommendation.

Model	AUC	F1	Precision@20	Recall@20
		Book-Crossing		
w/o User Graph	0.7344 ± 0.0034	0.6657 ± 0.0029	0.0120 ± 0.0011	0.1651 ± 0.0013
CaRecK	0.7590 ± 0.0052	0.6885 ± 0.0036	0.0146 ± 0.0023	0.1667 ± 0.0016
		Last.FM		
w/o User Graph	0.8106 ± 0.0025	0.7423 ± 0.0038	0.0285 ± 0.0012	0.3227 ± 0.0030
CaRecK	0.8295 ± 0.0047	0.7633 ± 0.0064	0.0300 ± 0.0025	0.3478 ± 0.0034

5.4. Parameter Analysis (RQ3)

5.4.1. Effect of Neighborhood Size K_n

We investigate the impact of the neighborhood size K_n, which controls the maximum number of selected relation–entity pairs. We evaluate $K_n \in \{4, 6, 8, 10\}$, together with an "All" setting that retains all candidate neighbors. Figure 8 presents the results on both datasets. We observe that the optimal value of K_n varies across datasets. On Last.FM, the performance improves as K_n increases and reaches its peak at $K_n = 8$, indicating that incorporating additional neighborhood information helps capture richer item semantics. However, a larger neighborhood size does not always yield better performance, as excessive neighbors may introduce irrelevant semantic signals and weaken preference representation learning. In contrast, it achieves the best performance when $K_n = 4$ on Book-Crossing. As the neighborhood size increases, both AUC and F1 gradually decline. A possible explanation is that the Book-Crossing KG contains more noisy or less informative neighbors, making large-scale neighborhood aggregation less effective. These results demonstrate the importance of balancing semantic enrichment and noise suppression when selecting the neighborhood size.

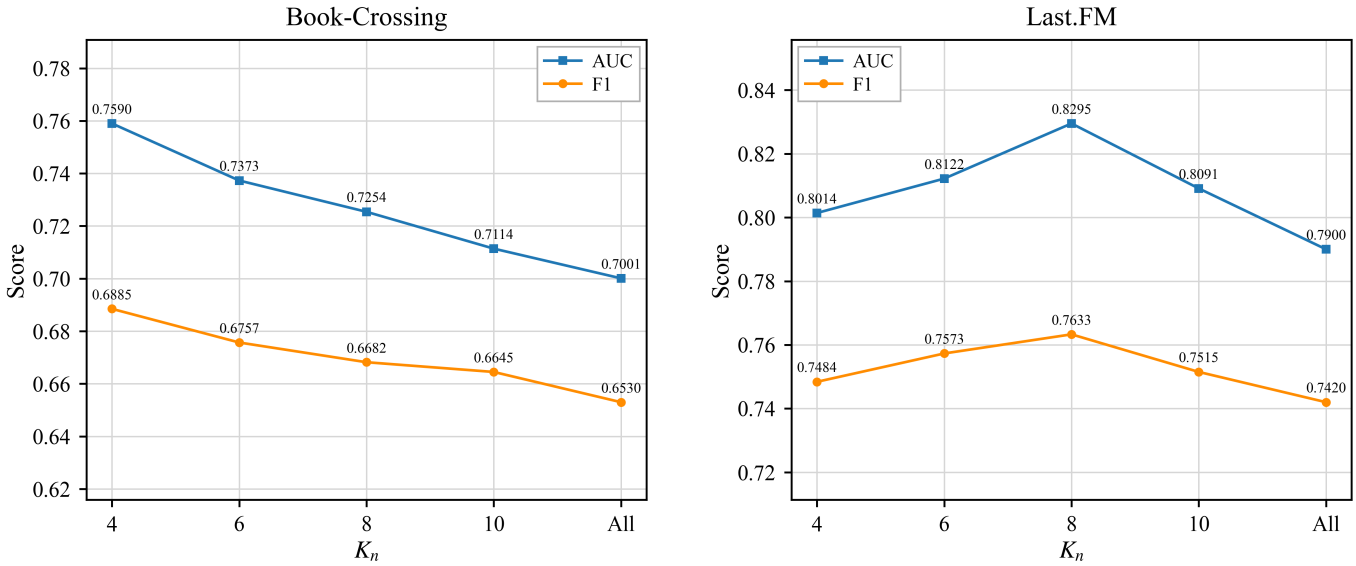


Figure 8. Effect of neighborhood size K_n.

5.4.2. Effect of the Number of Graph Convolution Layers T

We further study the impact of the number of graph convolution layers T. As shown in Figure 9, CaRecK achieves the best performance with $T = 4$ on Last.FM and $T = 3$ on Book-Crossing, indicating that an appropriate propagation depth enables the model to effectively exploit high-order KG semantic information. However, excessively increasing the number of layers does not further improve performance. This is mainly because

deeper propagation introduces more distant neighborhood information, which may cause representation over-smoothing, thereby weakening preference representation learning.

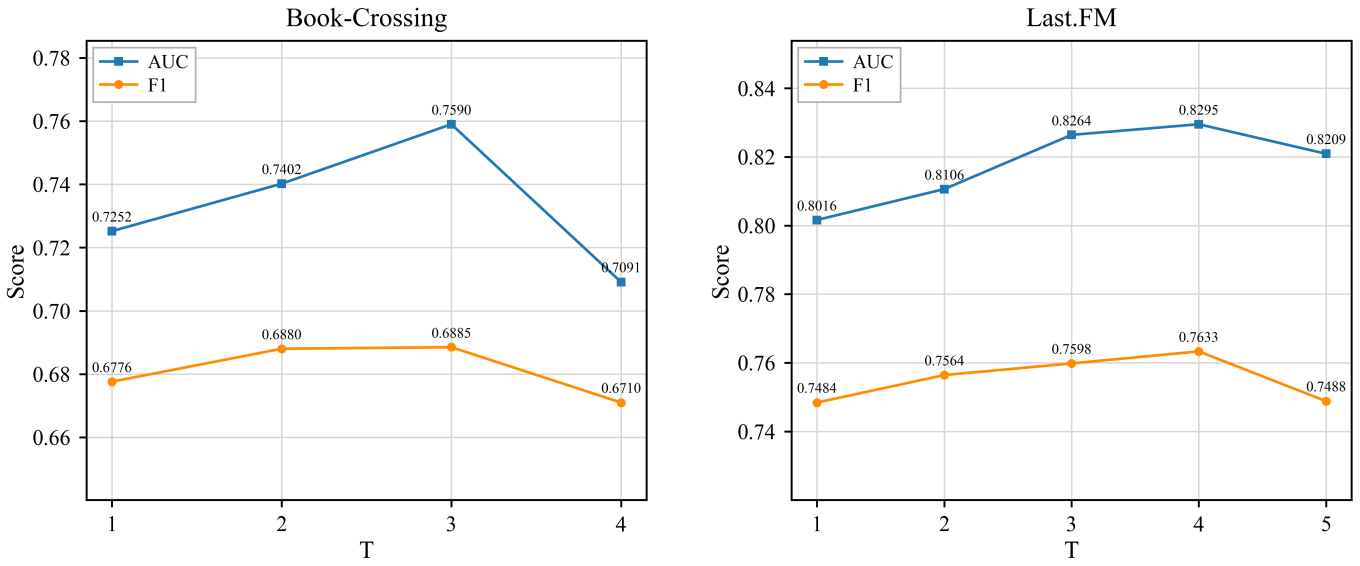


Figure 9. Effect of the graph convolution layer T.

6. Conclusions

In this paper, we proposed CaRecK, a knowledge-refined causality-inspired preference modeling framework for recommendation systems. The user representation module estimates observational attribute relevance and incorporates the most relevant attributes into user preference modeling. The item representation module selects user-relevant relation–entity pairs from the KG neighborhood and integrates their semantic information through hybrid knowledge aggregation. Experiments demonstrate that these two modules complement each other and improve both interaction prediction and top-K recommendation performance. Extensive experiments conducted on two real-world benchmark datasets show that CaRecK achieves competitive performance in both CTR prediction and Top-K recommendation tasks.

In future work, we will explore more rigorous causal reasoning strategies, such as counterfactual inference, and evaluate the proposed method on larger and more diverse datasets to further assess its scalability and generalizability.

Author Contributions: Conceptualization, H.Y. and S.S.; Methodology, S.S.; Validation, M.Y.; Formal analysis, J.W.; Investigation, Y.X.; Writing–original draft preparation, H.Y., S.S., and M.Y.; Writing–review and editing, H.Y., X.Z., and W.W.; Visualization, J.W. and Y.X.; Supervision, H.Y. and X.Z.; Funding acquisition, H.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Science and Technology Project of State Grid Gansu Electric Power Company (Grant No. 522723260004).

Data Availability Statement: Data will be made available on request.

Acknowledgments: The authors would like to thank the anonymous reviewers and editors for their valuable comments and suggestions.

Conflicts of Interest: Author Hao Yang, Shuo Shen, Mingjie Yang, Junwei Wang and Yipeng Xiong was employed by the State Grid Gansu Electric Power Company. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; Wang, M. 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*; ACM: New York, NY, USA, 2020; pp. 639–648.
2. Zhang, L.; Kang, Z.; Sun, X.; Sun, H.; Zhang, B.; Pu, D. KCRec: Knowledge-aware representation graph convolutional network for recommendation. *Knowl.-Based Syst.* **2021**, *230*, 107399. [CrossRef]
3. Chen, C.; Zhang, M.; Ma, W.; Liu, Y.; Ma, S. Jointly non-sampling learning for knowledge graph enhanced recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*; ACM: New York, NY, USA, 2020; pp. 189–198.
4. Wang, X.; Zhang, Z.; Shen, G.; Lai, S.; Chen, Y.; Zhu, S. Multi-view knowledge graph convolutional networks for recommendation. *Appl. Soft Comput.* **2025**, *169*, 112633. [CrossRef]
5. Zhou, S.; Dai, X.; Chen, H.; Zhang, W.; Ren, K.; Tang, R.; He, X.; Yu, Y. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*; ACM: New York, NY, USA, 2020; pp. 179–188.
6. Ge, Y.; Liu, S.; Fu, Z.; Tan, J.; Li, Z.; Xu, S.; Li, Y.; Xian, Y.; Zhang, Y. A survey on trustworthy recommender systems. *ACM Trans. Recomm. Syst.* **2024**, *3*, 1–68. [CrossRef]
7. Wang, X.; Huang, T.; Wang, D.; Yuan, Y.; Liu, Z.; He, X.; Chua, T.S. Learning intents behind interactions with knowledge graph for recommendation. In *Proceedings of the Web Conference 2021*; ACM: New York, NY, USA, 2021; pp. 878–887.
8. Zhang, F.; Yuan, N.J.; Lian, D.; Xie, X.; Ma, W.Y. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; ACM: New York, NY, USA, 2016; pp. 353–362.
9. Pearl, J.; Glymour, M.; Jewell, N.P. *Causal Inference in Statistics: A Primer*; John Wiley & Sons: Hoboken, NJ, USA, 2016.
10. Liang, D.; Charlin, L.; Blei, D.M. Causal inference for recommendation. *Causation Found. Appl. Workshop UAI. AUAI* **2016**, *6*, 108.
11. Gao, C.; Zheng, Y.; Wang, W.; Feng, F.; He, X.; Li, Y. Causal inference in recommender systems: A survey and future directions. *ACM Trans. Inf. Syst.* **2024**, *42*, 1–32. [CrossRef]
12. Gheewala, S.; Xu, S.; Yeom, S. In-depth survey: Deep learning in recommender systems—Exploring prediction and ranking models, datasets, feature analysis, and emerging trends. *Neural Comput. Appl.* **2025**, *37*, 10875–10947. [CrossRef]
13. Yang, Y.; Huang, C.; Xia, L.; Li, C. Knowledge graph contrastive learning for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*; ACM: New York, NY, USA, 2022; pp. 1434–1443.
14. Tang, G.; Gan, X.; Wang, J.; Lu, B.; Wu, L.; Fu, L.; Zhou, C. Editkg: Editing knowledge graph for recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*; ACM: New York, NY, USA, 2024; pp. 112–122.
15. Wei, Y.; Wang, X.; Nie, L.; Li, S.; Wang, D.; Chua, T.S. Causal inference for knowledge graph based recommendation. *IEEE Trans. Knowl. Data Eng.* **2022**, *35*, 11153–11164. [CrossRef]
16. Tan, S.; Li, D.; Jiang, R.; Wang, Z.; Yu, X.; Okumura, M. Taming recommendation bias with causal intervention on evolving personal popularity. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*; ACM: New York, NY, USA, 2025; pp. 2790–2800.
17. Liu, X.; Yuan, J.; Zhou, Y.; Li, J.; Huang, F.; Ai, W. CSRec: Rethinking Sequential Recommendation from A Causal Perspective. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*; ACM: New York, NY, USA, 2025; pp. 1562–1571.
18. Duan, G.; Yao, Y.; Ji, Z.; Gong, T.; Yan, J. Asymmetric Spectral Filtering and Behavior-Guided Graph Convolution for Multimodal Recommendation. *Electronics* **2026**, *15*, 2764. [CrossRef]
19. Wang, H.; Zhang, F.; Wang, J.; Zhao, M.; Li, W.; Xie, X.; Guo, M. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*; ACM: New York, NY, USA, 2018; pp. 417–426.
20. Wang, H.; Zhao, M.; Xie, X.; Li, W.; Guo, M. Knowledge graph convolutional networks for recommender systems. In *Proceedings of the WWW '19: The World Wide Web Conference*; ACM: New York, NY, USA, 2019; pp. 3307–3313.
21. Wang, Z.; Lin, G.; Tan, H.; Chen, Q.; Liu, X. 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*; ACM: New York, NY, USA, 2020; pp. 219–228.
22. Ahmadian, S.; Berahmand, K.; Rostami, M.; Forouzandeh, S.; Moradi, P.; Jalili, M. Recommender systems based on non-negative matrix factorization: A survey. *IEEE Trans. Artif. Intell.* **2025**, *6*, 2554–2574. [CrossRef]
23. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

24. Yu, X.; Ren, X.; Sun, Y.; Gu, Q.; Sturt, B.; Khandelwal, U.; Norick, B.; Han, J. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*; ACM: New York, NY, USA, 2014; pp. 283–292.
25. Wang, H.; Zhang, F.; Xie, X.; Guo, M. DKN: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*; ACM: New York, NY, USA, 2018; pp. 1835–1844.
26. Sang, L.; Xu, M.; Qian, S.; Wu, X. Knowledge graph enhanced neural collaborative recommendation. *Expert Syst. Appl.* **2021**, *164*, 113992. [CrossRef]
27. Bi, Q.; Yang, Y.; Ding, Y.; Pu, P. Intent Based Knowledge Graph Aware Recommender System with Causal Inference. In *Proceedings of the 2025 IEEE International Conference on Big Data (BigData)*; IEEE: Piscataway, NJ, USA, 2025; pp. 426–432.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.