Neurocomputing 523 (2023) 103–115

Contents lists available at ScienceDirect

Neurocomputing

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

Enhancing recommendations with contrastive learning from collaborative knowledge graph

Yubin Ma [a], Xuan Zhang [a,b,c,*], Chen Gao [d], Yahui Tang [a], Linyu Li [a], Rui Zhu [a], Chunlin Yin [e]

[a] School of Software, Yunnan University, Kunming 650091, China
[b] Yunnan Key Laboratory of Software Engineering, Kunming 650091, China
[c] Engineering Research Center of Cyberspace, Kunming 650091, China
[d] School of Information Science and Engineering, Yunnan University, Yunnan University, Kunming 650091, China
[e] Electric Power Research Institute of Yunnan Power Grid Co., Ltd., 650091, Kunming 650091, China

ARTICLE INFO

Article history:
Received 4 July 2022
Revised 26 October 2022
Accepted 11 December 2022
Available online 15 December 2022
Communicated by Zidong Wang

Keywords:
Graph neural network
Contrastive learning
Knowledge graphs
Recommender systems

ABSTRACT

There have been excellent results using knowledge graphs in recommender systems. Knowledge graphs can be used as auxiliary information to alleviate data sparsity and strengthen the modeling of item sets and the representation of user preferences. However, users as the Core subject in the recommendation process, should be taken seriously. We believe that the user's choice of items will be affected by internal and external factors. Internal factors refer to the users' fuzzy interest sets, which initially affect the users' choices. External factors refer to the influence of similar users and similar items in the users' selection of items. Inspired by the success of contrastive learning in graph collaborative filtering, we propose the **K**nowledge **A**ugmented **U**ser **R**epresentation (KAUR) model to explore contrastive learning in collaborative knowledge graphs, learning semantic neighbors (external factors) and extract fuzzy interest sets (internal factors) from collaborative knowledge graphs. Specifically, we use the graph neural network to learn the representation of each node in the collaborative knowledge graph and regard the information of nodes and their propagated neighbors' information as positive contrastive pairs, and then use contrastive learning to enhance the node representations. To further explore the potential interests of users, we regard users (or items) with other similar users (or items) as semantic neighbors and incorporate them into contrastive learning as positive pairings as well. Then the extracted fuzzy interest sets are merged into the user representations to get better interpretability. We conduct extensive experiments on three standard datasets and the results show that our KAUR model outperforms current state-of-the-art baselines.

1. Introduction

There is a lot of knowledge on today's Internet. Generally speaking, this knowledge is not isolated but interrelated; the same is true in recommender systems. The traditional collaborative filtering algorithms [14,29,50] are the cornerstone of recommendation system research. However, collaborative filtering algorithms usually face data sparsity and cold start problems. Knowledge graphs (KGs) provide rich item-side information [30,34,35,1], which alleviates the problem of data sparsity in the recommendation process and provides a new idea for the interpretability and accuracy research of recommender systems.

The main research content of the recommender systems based on KGs is how to integrate the heterogeneous information on the item side into embedding users and items. The current KGs-based recommender system research can be roughly divided into three categories: embedding-based methods [1,48,31,17,33,2,16], path-based methods [30,49,47,12,15,10,44,38] and propagation-based methods [34,32,39,35,7,51,28,22]. DKN [31] learns entities contained in news through the knowledge graphs embedding method to capture the connections between different news for click-through rate prediction. RippleNet [30] incorporates an embedding-based approach into recommendation through preference propagation and explores possible paths from user-interacted items to candidate items. KGCN [34] and KGNN-LS [32] sample the neighbor information of each entity in the KGs and aggregate the sampled information onto item entities. Model higher-order information by extending the receptive field to capture the potential

* Corresponding author.
E-mail address: zhxuan@ynu.edu.cn (X. Zhang).

https://doi.org/10.1016/j.neucom.2022.12.032
0925-2312/

long-range interests of users. MKGAT [26] leverages multi-modal knowledge to provide better recommendation, which utilizes graph attention mechanism for information dissemination on multimodal KGs and uses the resulting aggregated embeddings for the recommendation. FG-RS [8] uses the user-item's historical interactions to model the user's interaction preferences and mines the user's fine-grained preferences from all attributes of the items. KGIN [37] extracts different intents from KGs to represent user preferences, distinguishes paths of node information and encodes the semantic and relational dependencies of paths into representations.

Although the current research has achieved good results, we find that there are still some problems: (1) The user is an important role in the recommendation process and the quality of user modeling will greatly affects the performance of the recommender systems. Although KGs can be viewed as graph structure of attribute knowledge about items, it is more or less important to users. However, KGCN [34] and KGNN-LS [32] fail to adequately model users, downplay the role of users in the recommendation process and ignore explicit collaboration signals and interactions. Therefore, we explore some factors that can enhance user representations from the KGs. (2) In real life, a user's preference for an item is not only affected by similar items, but also by similar users. However, MKGAT [26] and KGIN [37] learn collaboration signals from graph structures and relational paths, respectively, and fail to consider the effects of similar users fully. Moreover, it is far from enough to learn the influence of the interaction graph between the users and the items or the physical structure of the KGs, so we need to learn this potential influence through the semantic structure to improve the user modeling further.

Applying contrastive learning to graph collaborative filtering is also a recent research hotspot. SGL [42] generates multiple views for each node through the random walk, node missing, relationship missing, and maximizes the similarity of different views of the same node and minimizes the consistency of views of different nodes. MCCLK [55] considers three different views for KG-aware recommendations to mine comprehensive graph features and structural information in a self-supervised manner. These methods are effective in the recommendation domain. However, the current way of constructing a comparison structure cannot effectively utilize graph nodes' high-order relationship and ignores collaboration and semantic information.

To solve the above problems, this paper aims to strengthen user modeling and propose a Knowledge Augmented User Representation (KAUR) model. Specifically, it mainly contains two structures: (1) Collaborative knowledge graph message dissemination and aggregation. We perform LightGCN [13] operation on the collaborative knowledge graph (CKG) to obtain the high-order structure and semantic information in the CKG to learn the representation of each node. Of course, this is not enough. Compared with the traditional contrastive learning of different views, we are inspired by NCL [19] and find that the output of the k-th layer of graph neural network (GNN) is the aggregated information of k-hop neighbors containing nodes. Therefore, we regard each node of the CKG and its k-hop neighbor aggregated information as node-level positive alignment, and use contrastive learning to minimize the distance between them and enhance the representation learning of nodes. (2) Fuzzy interest set extraction and user representation enhancement. Unlike KGIN [37], we believe that when users choose items in most cases, their preferences are vague and their preferences will be affected by internal and external factors. The internal factors refer to the user's own fuzzy interest set, which drives the user to choose different items. The semantics of fuzzy interest sets are opaque. Considering the importance of relation composition, we extract fuzzy interest sets from the relation set of CKGs, and emphasize the maximum difference of each element in the fuzzy interest sets. External factors refer to users being influenced by similar users (or items) when making item selections. To mine the potential influence of similar users (or items), we define them as semantic neighbors. Semantic neighbors refer to semantically similar neighbors that may not be directly connected in the graph structure. Since centroids exist for the set of semantic neighbors of users (or items), we use contrastive learning to obtain correlations between users (or items) and the centroids.

Our contributions to this work can be summarized as follows:

- We apply node-level contrastive learning to collaborative knowledge graphs, minimizing the distance between nodes and high-order neighbor information to enhance node representations.
- We propose a knowledge-augmented user representation model KAUR, which explores user fuzzy interest sets and semantic neighbors from both internal and external aspects.
- We conduct extensive experiments on three public datasets and demonstrate that our model outperforms current state-of-the-art baselines.

The rest of this paper is organized as follows. Section 2 provides a comprehensive overview of related work, including knowledge graph-based recommendation, contrastive learning, and graph collaborative filtering. In Section 3, we introduce the relevant concepts of this work and define our task. In Section 4, we introduce the implementation details of KAUP. In Section 5, we present the experimental results and analyze them in detail. In Section 6, we summarize our work and give directions for future research.

2. Related work

This section discusses existing work on KGs-based recommendation, contrastive learning, and graph collaborative filtering, which are closely related to our work.

2.1. KGs-based recommendation

2.1.1. Embedding-based methods

Most of the early research on KGs-based recommender systems used knowledge graph embedding models (TransE [5], TransR [18], TransH [40]). CKE [48] learns the representation of items by considering the heterogeneity of entities and relationships in the knowledge graph through TransR [18]. KTUP [6] uses TransH [40] to unify the recommendation task and the knowledge graph completion task, where user preferences are induced by user-interaction and KG relations. CFKG [1] combines user behavior, item and knowledge information into a user-item KG and considers user behavior (click, purchase) as a relationship between entities. Huang et al. [17] integrate an RNN-based network with a key-value memory network (KV-MN) and use a knowledge base to enhance the semantic representation of KV-MN to improve the performance of sequential recommendations. MKR [33] utilizes the knowledge graph embedding task to assist the recommendation task, modeling the problem as multi-task learning. And a cross-compression unit is designed to automatically share latent features and learn higher-order interactions. TransMKR [16] introduces multi-task learning into point of interest recommendation, using TransR [18] to improve MKR's knowledge graph embedding module to quantify the relationship between point of interests and their attributes. KGFlex [2] employs lowdimensional embeddings of knowledge in KGs to represent item features and simulates user-item interactions by combining user-relevant subsets of item features. Embedding-based methods deal with each entity and relation learning individually, focusing on strict semantic relevance, i.e., this method is more suitable for link prediction rather than recommendation tasks.

2.1.2. Path-based methods

Path-based methods explore connection patterns between nodes in a KG and extract paths with higher-order information. Most of these methods utilize multi-hop paths in KGs to improve recommendation performance. These paths can also be used as propagation paths of user preferences to represent user preferences intuitively. FMG [49] and PER [47] treat a KG as a heterogeneous information network, from which meta-path latent features are extracted to represent the connectivity between users and along different types of relational paths. In the field of course recommendation, ACKRec [12] uses graph convolution network to learn entity representations for content information and context information, and uses meta-paths over heterogeneous information network to guide the propagation of student preferences. MCRec [15] and MEIRec [10] use interactions between users and items to encode metapaths, where the dataset predefines meta-paths. PGPR [44] uses reinforcement learning to build a path between users and items, and extract recommendation results along a multihop path. TMER [9] places the user's dynamic behavior on the global knowledge graph for order-aware recommendation, and exploits the attention mechanism to explore user-item and item-item *meta*-paths for interpretable recommendations. KPRN [38] uses entity embedding and relation embedding to build relation paths, and then encodes relation paths with LSTM. However, when the KG is relatively large, it is very troublesome to manually design the *meta*-path or *meta*-graph, and the selection of the initial path is very important to the model performance, so these methods are difficult to optimize in practice.

2.1.3. GNN-based methods

In recent years, GNNs have shown great potential in text classification, node prediction and recommender systems. A GNN aims to model nodes and graph structures, a common way is to integrate multi-hop neighbor information into node representations. CKAN [39] adopts a heterogeneous propagation strategy to explicitly encode cooperative and knowledge-aware signals, and uses a knowledge aware attention mechanism to distinguish the sharing of neighbors. KGAT [35] integrates user-item interaction graph and knowledge graph and regards them as a collaborative knowledge graph, recursively propagates the embeddings from the entity's neighbors to refine the entity's embedding, and uses an attention mechanism to distinguish the importance of different neighbors. MVIN [27] learns item representations from the user view and entity view, respectively, not only aggregating high-order connection information, but also mixing the information of layer-by-layer GCN. TGCN [7] uses type aware neighbor sampling and aggregation operations to learn type-specific neighborhood representations, and performs vertical and horizontal convolution operations based on GCN networks to model multi-class unique feature interactions. [52] is a dialogue-based recommendation model that uses GNN to learn the word-side KG and item-side KG separately and then bridges the semantic gap between the two KGs based on mutual information maximization. KCAN [28] is a model for refining and refining knowledge graphs, which automatically extracts knowledge graphs into target-specific subgraphs based on a knowledge-aware attention mechanism and then uses conditional attention aggregation on the subgraphs to obtain target-specific node representations. KGNCFRRN [49] incorporates residual learning into traditional RNN networks to efficiently encode long-term relational dependencies of KGs and embed users and items into a newly designed 2D interaction graph.

2.2. Contrastive learning-based methods

After contrastive learning became popular in the field of vision, researchers applied the idea of contrastive learning to NLP, graph data mining, and recommendation systems. The contrastive learning aims to minimize the distance between positive samples while maximizing the distance between negative samples. InfoGraph [25] maximizes the mutual information between representations of graphs and substructures at different scales. GRACE [54] generates two views by corruption and learns node representations by maximizing the consistency of node representations in these two views. SGI [42] generates multiple views for each node of the user-item bipartite graph, taking node self-discrimination as a self-supervised task. MCCLK [55] constructs a global-level structural view, a local-level structural view and a semantic-level view and performs contrastive learning at the local level and the global level, and explores the semantic relationship between items in the semantic-level view. NCL[19] constructs contrastive alignments from structural neighbors and semantic neighbors, respectively, and improves graph neural collaborative filtering. KGIC [56] first constructs local and nonlocal graphs for user/item. Then an intra-graph level interactive contrastive learning is performed within each local/non-local graph, which contrasts layers of the CF and KG parts, for more consistent information leveraging. KGCL[45] is a KG-guided topological denoising framework. It creates contrast views for the KG and the user-item interaction graph respectively and improves the model robustness with augmented self-supervision signals. C-KGAT[20] performs node embedding dropout (or edge dropout) on the CKG to generate different views, so as to enhance the downstream contrastive learning. RGCL[23] designs two additional contrastive learning tasks (i.e., Node Discrimination and Edge Discrimination) to provide self-supervised signals for the two components in recommendation process. XSimGCL[46] discards the ineffective graph augmentations and instead employs a simple yet effective noise-based embedding augmentation to create views for contrastive learning. HCCF[43] uses the hypergraph-enhanced cross view contrastive learning architecture to jointly capture local and global collaboration and effectively integrates the hypergraph structure encoding with self-supervised learning to reinforce the representation quality of recommender systems. However, the application of contrastive learning to KGs-based recommendation system is less known. Additionally, in this paper, we assume that users with similar representations are within the semantic neighborhood, and incorporate these semantic neighbors into the prototype-contrastive objectives.

2.3. Graph-based collaborative filtering

Early collaborative filtering models such as matrix factorizations project user and item IDs into embedding vectors and reconstruct historical interactions to learn embedding parameters. GNNs reveal the modeling of graph structure, especially k-hop neighbors, to guide embedding learning. Graph-based collaborative filtering organizes interaction data into a user-item interaction graph and learns meaningful node representations from graph structure information. NNCF [3] utilizes interaction information to obtain interaction-based neighborhoods and integrates the neighborhood information into neural collaborative filtering methods. GC-MC [4] uses an autoencoder to generate latent features for user and item nodes in passing information on a bipartite interaction graph, and reconstructs the latent representations of users and items via a bilinear decoder. SI-News [53] utilizes an attention GCN to embed user interests from a user's social novel, and optimizes news headlines and contents through an attention mechanism to improve news representations. SpectralCF [51] explores all possible connectivity information between users and items through spectral convolution operations. NGCF [36] utilizes the user-item graph structure to propagate embeddings, effectively injecting collaboration signals into the embedding process in an explicit manner, and exploiting higher order relations on the graph to improve recommendation performance. LightGCN [13] removes feature transformation and nonlinear activation on the basis of NGCF [36]. It takes the weighted sum of the embeddings learned by all layers as the final embedding, making the whole process concise and efficient. GMCF [24] effectively leverages internal interactions for the user and item feature learning (via graph learning) and cross interactions for the preference matching (via graph matching). Different from these models, we additionally emphasize the modeling of users. On the one hand, we extract fuzzy interest sets from the CKG to model users' preferences; on the other hand, we explore semantic neighbors for users and items to conduct contrastive learning at the node level.

3. Problem formulation

We describe the notations used in this paper in Table 1, then present the related concepts and define the recommendation task.

Item Knowledge Graph. The auxiliary information on the item side and the item set together constitute an item knowledge graph $G = \{(h, r, t) | h, t \in E, r \in R\}$, which describes the real-world items and their compositional relationships. Nodes or entities in the item knowledge graph (IKG) represent items, and edges describe the relations between items. Each edge belongs to a relation type, where R is a set of relation types. Taking a movie KG as an example, (*ForrestGump*, *ActedBy*, *TomHanks*) illustrates that *TomHanks* starred in *ForrestGump*. In this paper, entities and relations in the item knowledge graph (IKG) were obtained directly from the public datasets for experiments.

Collaborative Knowledge Graph. A collaborative Knowledge Graph (CKG) encodes a user-item interaction graph and an item knowledge graph (IKG) as a unified knowledge graph. A user-item interaction graph regards user-item interactions as triples, indicating that users interact with items, and vice versa. It is defined formally as$G_{interaction} = \{(u, y_{ui}, i) | u \in U, i \in I\}$, where U and I are the users set and the items set, $y_{ui} = 1$ indicates that the user interacts with the item, and $y_{ui} = 0$ indicates no interaction. In this paper, the user-item interaction graphs are also constructed from the public datasets for experiments and connected with the item knowledge graph into a CKG $G_{ckg} = \{(h, r, t) | h, t \in E', r \in R'\}$ where$E' = E \cup U, R' = R \cup \{y_{ui}\}$.

Fuzzy Interest Set. A fuzzy interest set $F = \{f_1, f_2, f_n\}$ is shared by all users, and describes the user's attitude towards the items. Different fuzzy interests abstract different behavioral patterns of users. For example, user u_1 chooses one movie because he is a fan of the protagonist, while user u_2 chooses another movie because he is attracted by the storyline. For the sake of understanding, we regard the *protagonist* and *storyline* as fuzzy interest f_1 andf_2. Different fuzzy interests abstract different behavioral patterns of users. That is, f_1 is the main factor that drives u_1 to choose this movie and f_2 is the main factor that drives u_2 to choose the other movie. The specific construction steps of a fuzzy interest set are explained in detail in Section 4.2.1.

Table 1
Notations and Explanations.

Notation	Explanation
U	a set of users
u	user
I	a set of items
i	item
E	a set of entities
R	a set of KG relation types
A	a set of item entity alignment
G	Knowledge Graph
G_{ckg}	collaborative knowledge graph
$G_{interaction}$	user-item interaction graph
F	a set of fuzzy interest
k	the number of semantic neighbor groups
N_u	the user's neighbor nodes
N_i	the item's neighbor nodes
$\hat{y}_{u,i}$	the score between user u and item i

Semantic Neighbors. Semantic neighbors refer to nodes with similar features in the CKG, which correspond to similar users or similar items in real life. In particular, semantic neighbors may not be directly reachable in CKG and we extract the center of the semantic neighbor cluster and regard it as a prototype p to better capture the semantic features of users or items. We explain semantic neighbors in detail in Section 4.2.2.

Task Description. Input a CKGG_{ckg}, and output the probability prediction function $\hat{y}_{u,i}$ that the user adopting a certain item.

4. Methodology

In this section, we introduce the KAUR in detail. As shown in Fig. 1, the model framework mainly consists of two parts:

(1) **Message propagation and aggregation**. Consider each node in the CKG and its k-hop $(k = 1, \ldots, n)$ neighbor information as positive contrastive pairs. We use contrastive learning to narrow the gap between them and then aggregate the information from the neighbors to update the node representation.
(2) **Fuzzy interest extraction and user representation enhancement**. (a) *Internal factors modeling*. Generate fuzzy interest sets from relation sets in CKG, emphasizing the maximization of element differences in sets. (b) *External factors modeling*. The semantic neighbors of each node in the CKG are incorporated into contrastive learning to better capture the semantic features of users.

4.1. Message propagation and aggregation

To obtain the preliminary representation of individual node in the CKG, we use a GNN-based method for message aggregation and propagation. Specifically, similar to the operation of LightGCN [13], we abandon the use of feature variation and nonlinear activation. The graph convolution operations for users, items and entities are defined in Eq. (1), (2), (3) and (4).

$$e_u^{(k+1)} = \sum_{i \in N_u} \frac{1}{\sqrt{|N_u|}\sqrt{|N_i|}} e_i^k \tag{1}$$

$$e_i^{(k+1)} = \sum_{u \in N_i} \frac{1}{\sqrt{|N_i|}\sqrt{|N_u|}} e_u^k \tag{2}$$

$$e_h^{(k+1)} = \sum_{t \in N_h} \frac{1}{\sqrt{|N_h|}\sqrt{|N_t|}} e_t^k \tag{3}$$

$$e_t^{(k+1)} = \sum_{h \in N_t} \frac{1}{\sqrt{|N_t|}\sqrt{|N_h|}} e_h^k \tag{4}$$

where $e_x^{(k)} (x = i, u, h, t)$ represents the embedding of user, item, head entity and tail entity after $k (k = 1, 2, \cdots, n)$ layers of propagation, $N_x (x = i, u, h, t)$ represents the neighbor node connected to$x$, $\frac{1}{\sqrt{|N_u|}\sqrt{|N_i|}}$ and $\frac{1}{\sqrt{|N_h|}\sqrt{|N_t|}}$ are symmetric normalization coefficients

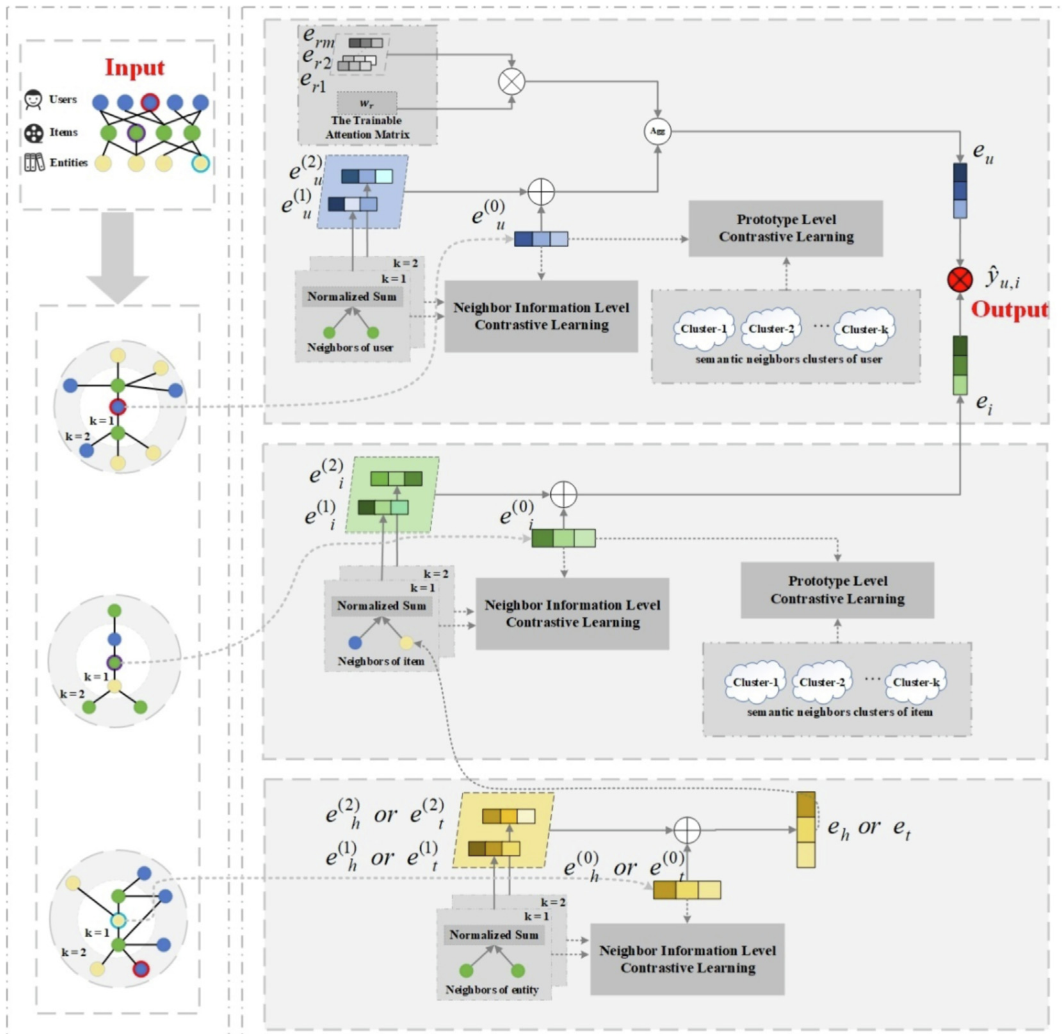


Fig. 1. Illustration of the proposed KAUR.

[41], which avoid the complex calculation when the embedding scale is too large. Using Eq. (4) as an example, N_t represents the neighbor node set (head entity set) connected by the tail entity e_t, and N_h represents the neighbor node set (tail entity set) connected by the head entity e_h.

To further enrich the node representation in the CKG, we perform contrastive learning on each node and its neighbors. The representation of neighbor information is obtained through the above graph convolution operation. The k - th layer output in GCN is the weighted sum of the node's k hop neighbor information. That is to say that the output of the k - th layer is aligned with this node as a positive contrastive pair and the distance between them is minimized. Based on InfoNCE [21], we propose the Eq. (5) as the loss function for users.

$$L^u_{ssl} = \sum_{u \in U} -\log \frac{exp((e^k_u \cdot e^0_u/\tau))}{\sum_{u' \in U} exp((e^k_u \cdot e^0_{u'}/\tau))} \tag{5}$$

where $e^{(k)}_u$ is the normalized output of the GCN layer$k$, τ is the temperature hyper parameter for *softmax*. Similarly, we compute L^i_{ssl} on the item side and L^e_{ssl} on the CKG entity side. The total loss function is a weighted sum of the three loss functions:

$$L_{ssl} = \alpha L^u_{ssl} + \beta L^i_{ssl} + \gamma L^e_{ssl} \tag{6}$$

where α, β, γ are hyper parameters used to balance the weights.

4.2. Fuzzy interest extraction and user representation enhancement

4.2.1. Fuzzy interest extraction (Internal factors)

We believe that users will be affected by an internal factor when selecting items—fuzzy interest sets, which are common to all users. For a movie-related example, some users like a movie because of actors, some users are interested in a movie because of genres, and some users are fans of directors. Different combinations of these vague interests of actors, genres and directors affect users' preferences for movies.

Considering that there is a significant set of relations in CKG, we do not consider a single relation in isolation. Because if we don't consider the interaction and combination of relations, it's impossible to refine the advanced concept of fuzzy interests. For example, the combination of relations r_1 and r_2 has an impact on fuzzy interestf_1, while relations r_3 and r_4 make a different contribution to fuzzy interestf_2. So we regard a fuzzy interest as a combination of relations and use attention strategies to create it:

$$e_f = \sum_{r \in R} \pi(r, f) e_r \tag{7}$$

where e_f and e_r are the embedding of fuzzy interest and relation in CKG. $\pi(r, f)$ is the attention score for generating fuzzy interests, which controls the proportion of relation embedding in fuzzy interests. $\pi(r, f)$ is calculated in Eq. (8)

$$\pi(r,f) = \frac{exp(w_r)}{\sum_{r' \in R} exp(w_{r'})} \tag{8}$$

where $w_{r'}$ is the trainable weight of fuzzy interest in specific relations and features. The attention score is not for a specific user, but refined to all users.

Using the above example of a movie recommendation. It can be seen that the semantic differences between the three fuzzy interests of actors, genres and directors are very obvious. Therefore, we need to maximize the differences between the elements in the fuzzy interest sets and reduce the interdependence, so that it can effectively describe the behavior patterns of users.

We compute the correlation distance minimization loss function [37] to emphasize the differences in fuzzy interests, which are the linear and nonlinear correlation of any two variables:

$$L_{distance} = \sum_{f,f' \in F, f \neq f'} \frac{dCov(e_f, e_{f'})}{\sqrt{dVar(e_f) \cdot dVar(e_{f'})}} \tag{9}$$

where $dCov(\cdot)$ is the distance covariance of the two ambiguous interests and $dVar(\cdot)$ is the distance variance of each ambiguous interest. Fuzzy interests are independent of each other, giving users better interpretability.

4.2.2. User representation enhancement (External factors)

Inspired by traditional collaborative filtering algorithms, we know that users are influenced by semantic neighbors (similar users and similar items) when selecting items. To capture the semantic features of users and items in collaborative filtering, we learn latent prototypes of users and items to identify semantic neighbors.

Similar users or items tend to fall in neighboring embedding spaces and the prototypes are the center of clusters that represent a group of semantic neighbors. We use K-means clustering algorithm to get the prototypes of users and items. As shown in Fig. 2, user/item nodes need to minimize the difference from prototype in a semantic neighbor set and maximize the difference from other prototypes. Also, based on InfoNCE [41], we propose a contrastive learning loss function between users and their prototypes:

$$L_{ssl}^{PU} = \sum_{u \in U} -\log \frac{exp(e_u \cdot p_i / \tau)}{\sum_{p_j \in P} exp(e_u \cdot p_j / \tau)} \tag{10}$$

where p_i is the prototype of user$u$, K-means algorithm is used to cluster the embedding of the user set and the users are divided into k categories. Similarly we get the contrastive learning loss function of the items and its prototype, the complete loss function is the weighted sum of the two:

$$L_{ssl}^{P} = \alpha L_{ssl}^{PU} + \beta L_{ssl}^{PI} \tag{11}$$

In this way, consider the influence of similar users with similar items, which further strengthens the user representation through graph convolution operations. It conforms to the actual situation, thereby bringing interpretability to the recommender system.

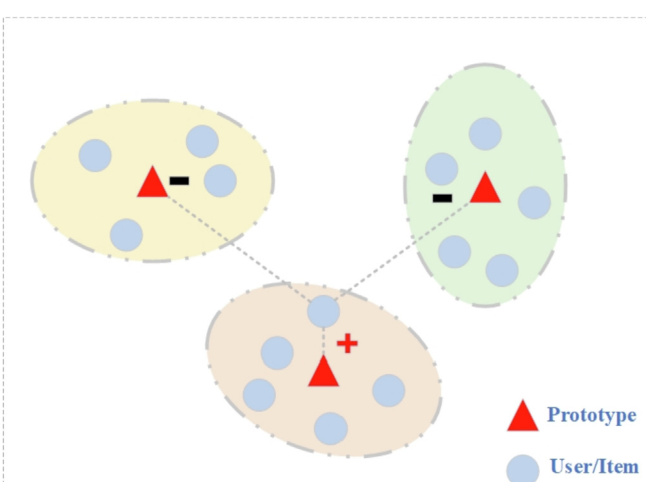


Fig. 2. Semantic Neighbor Set and Prototype.

4.3. Model prediction

After k-layer propagation, the weighted sum operation is performed on the representations of all layers, as shown in Eq. (12). The definition of Eq. (12) is inspired by LightGCN [13]. Before defining Eq. (13), we calculate the higher-level neighbor embedding of users and items in Eq. (1) and (2). The meaning of Eq. (12) is that after k layers propagation, we obtain$k+1$ embeddings to describe a user ($e_u^{(0)}, e_u^{(1)}, e_u^{(k)}$) and an item ($e_i^{(0)}, e_i^{(1)}, e_i^{(k)}$), and then sum and average the embedded values obtained from each layer to form the final representation of the user and item.

$$e_u = \frac{1}{k+1} \sum_{k=0}^{k} e_u^{(k)}, e_i = \frac{1}{k+1} \sum_{k=0}^{k} e_i^{(k)} \tag{12}$$

where e_i represents the final representation of the item, and we aggregate the fuzzy interest set with the user representation e_u to generate the final representation of the user:

$$e_u = \sum_{f \in F} \pi(u,f) e_f \odot e_u + e_u \tag{13}$$

$$\pi(u,f) = \frac{\exp\left(e_f^{\mathrm{T}} e_u\right)}{\sum_{f' \in F} \exp\left(e_{f'}^{\mathrm{T}} e_u\right))} \tag{14}$$

where $\pi(u,f)$ is the attention score that differentiates the importance of fuzzy interests, indicating that different fuzzy interests will prompt users to behave differently. In this way, the user's fuzzy interests set is integrated into the user's final representation.

Finally, we do an inner product of the user representation and the item representation to predict its matching score:

$$\hat{y}_{u,i} = e_u^{\mathrm{T}} e_i \tag{15}$$

4.4. Loss computing

To optimize the model, we employ a Bayesian personalized ranking BPR loss function. It assumes that observed interactions (indicating more user preferences) should be assigned higher prediction scores than unobserved interactions:

$$L_{BPR} = \sum_{(u,i,j) \in O} -\log \sigma(\hat{y}_{u,i} - \hat{y}_{u,j}) \tag{16}$$

where $\sigma(\cdot)$ is the sigmoid function, $O = \{(u,i,j)|(u,i) \in R^+, (u,j) \in R^-\}$ represents the training set, R^+ represents the observed (positive) interactions between user u and item$i$, and R^- is the sampled set of unobserved (negative) interactions.

The final loss function is:

$$L = L_{BPR} + L_{ssl} + L_{distance} + L_{ssl}^{P} + \lambda ||\Theta||_2^2 \tag{17}$$

where Θ is the parameter set and λ is the parameter for $L2$ regularization.

5. Experiments

In this section, we use three real datasets from different domains to evaluate the KAUR. We first introduce the three datasets in Section 5.1, and then discuss the experiment settings and results in Section 5.2 and 5.3. Furthermore, we also conduct ablation study and sensitivity analysis in Section 5.4 and 5.5, respectively.

5.1. Datasets

To evaluate the KAUR, we conduct experiments in three different scenarios: movies, books, and music. All three datasets are publicly accessible and vary in size and sparsity. Table 2 presents the statistics of the datasets.

- MovieLens-1M[1]:This is a classic movie recommendation dataset, which contains about 1 million explicit ratings (ranging from 1 to 5) for 3707 movies from 6041 users.
- Amazon-book[2]:It is a frequently used Amazon book dataset, which includes user rating data (ratings) and book metadata (description, category information, price, and brand).
- LFM-1b[3]:This is a music dataset collected from the online Last.FM system, which contains music listening events created by Last.FM users.

Table 2
Statistics of the datasets.

	MovieLens-1 M	Amazon-book	LFM-1b
#User	6041	8724	4862
#Item	3707	7392	9233
#Interaction	1,000,209	274,795	1,241,584
#Entity	79,399	19,281	28,877
#Relation Type	51	22	8
#Triplet	385,923	71,497	60,957
Density	0.04466	0.00426	0.02765

5.2. Experiment settings

5.2.1. Baselines

To demonstrate the effectiveness of our proposed model, we compare it with recent state-of-the-art KGs-based recommendation models, including KGAT, KGCN, KGNNLS, RippleNet, CFKG, MKR, and KGIN.

- KGAT [35] is a model that applies GAT to CFKG, which explicitly models higher-order connectivity in CKG in an end-to-end manner. In the propagation process, KGAT uses an attention mechanism to distinguish the importance of neighbor nodes and adaptively propagates embeddings from neighbor nodes to update node representations.
- KGCN [34] learns the structural information and semantic information of KGs and uses GCN to collect high-order neighborhood information from the IKG, which can iteratively integrate neighborhood information to enrich item embeddings.
- KGNNLS [32] applies the GNN architecture to KGs by using user-specific relation scoring functions and aggregating neighborhood information with different weights. This proposed label smoothness constraint provides a strong regularization for learning edge weights in KGs.
- RippleNet [30] is a method similar to memory network propagation, which can automatically propagate user preferences in the KG and explore users' hierarchical interests in the KG.
- CFKG [1] integrates user behavior and item knowledge into a unified graph structure and recommends reasonable predictions converted into triples, which can build explanations about recommended items by exploring paths in the graph embedding space.
- MKR [33] is a multi-task learning framework that utilizes knowledge graph embedding tasks to assist recommendation tasks. It can automatically learn higher-order interactions of item and entity features and transfer knowledge between tasks.
- KGIN [37] illustrates the connections between users and items by exploring intents, combining intents with knowledge graph relations and considering user-item relations with finer intent granularity and long-term semantics of relational paths under the GNN paradigm.

5.2.2. Evaluation metrics

To evaluate the effectiveness of top-k recommendation and preference ranking, we use the following two widely evaluated metrics: Recall@K and NDCG@K, where K=[10,20,50]. Recall@K describes a measure of the proportion of related items among all items. NDCG@K (also known as Normalized Discounted Cumulative Gain) is a measure of ranking quality that explains the position of hits by assigning higher scores to top-ranked hits. The calculation formulas are:

$$Recall@K = \frac{1}{|U|}\sum_{u\in U}\frac{|\hat{R}(u)\cap R(u)|}{|R(u)|}$$

$$NDCG@K = \frac{1}{|U|}\sum_{u\in U}\frac{\sum_{i=1}^{K}\delta(i\in R(u))\frac{1}{\log_2(i+1)}}{\sum_{i=1}^{\min(|R(u)|,K)}\frac{1}{\log_2(i+1)}}$$

where $R(u)$ represents the list of the top-k recommended items, and $\hat{R}(u)$ represents the number of items actually accessed by the users. $\delta(\cdot)$ is an indicator function, $\delta(x) = 1$ if x is true and 0 otherwise. We randomly select 80 % of interactions as training data and 10 % of interactions as validation data. The remaining 10 % of interactions are used for performance comparisons. We uniformly sample a negative item for each positive example to form the training set.

5.2.3. Parameter settings

To ensure the fairness of the experiments, we adopt the same operation dimension and item embeddings for our KAUR and the other baselines. We use Xavier [11] to initialize the model parameters and Adam to optimize the model. Since RecBole [50] includes most baseline implementations and corresponding parameter configurations, we select a reasonable parameter configuration and compare the original papers in each baseline with the super parameters provided by RecBole. The embeddings of all models are fixed to 64, where the batch size is set to 1024. The learning rate is explored between {0.0001, 0.0005, 0.001, 0.0015}. The size of the fuzzy interest set is controlled at 2, 3, 4, 5. The division range of semantic neighbors is in 100, 500, 800, 1000, 1500. The $L2$ normalization coefficients are set in$\{10^{-5}, 10^{-4}, ..., 10^{-1}\}$. We explore the effect of depth in the range {1, 2, 3, 4}. For the baselines, we use their default hyper-parameter settings except for the embedding dimension. To prevent overfitting, we control the number of steps for training convergence and set it to 50.

5.3. Results

Table 3 compares KAUR with the Top-k recommendations of other baselines. The best results are shown in bold, and the second-best results are underlined. Improv. Indicates the percent

[1] https://grouplens.org/datasets/movielens/.
[2] https://jmcauley.ucsd.edu/data/amazon/.
[3] https://www.cp.jku.at/datasets/LFM-1b/.

Table 3
Performance comparisons of different recommendation models (The best result is bolded and the runner-up is underlined).

		Without GNN-based methods			GNN-based methods					Improv.
Dataset	Metric	RippleNet	MKR	CFKG	KGCN	KGNNLS	KGAT	KGIN	KAUR	
MovieLens-1 M	Recall@10	0.1138	0.1361	0.1518	0.1383	0.1322	0.1625	<u>0.1708</u>	**0.1746**	2.22 %
	NDCG@10	0.1937	0.213	0.24	0.2213	0.2278	0.2531	<u>0.2627</u>	**0.2709**	3.12 %
	Recall@20	0.1963	0.2196	0.2359	0.2188	0.2289	0.2491	<u>0.262</u>	**0.266**	1.57 %
	NDCG@20	0.2005	0.2239	0.2473	0.2264	0.2327	0.2605	<u>0.2701</u>	**0.2768**	2.48 %
	Recall@50	0.3414	0.3801	0.399	0.3697	0.3864	0.4078	<u>0.4139</u>	**0.4258**	2.88 %
	NDCG@50	0.2397	0.2663	0.2849	0.266	0.2776	0.3	<u>0.3093</u>	**0.316**	2.17 %
Amazon-book	Recall@10	0.0696	0.0824	0.1156	0.0824	0.0976	0.1158	<u>0.1555</u>	**0.164**	5.47 %
	NDCG@10	0.0427	0.0501	0.074	0.0514	0.0604	0.0749	<u>0.1013</u>	**0.1095**	8.09 %
	Recall@20	0.1187	0.1515	0.1755	0.1388	0.1566	0.1813	<u>0.2287</u>	**0.2401**	4.98 %
	NDCG@20	0.0588	0.0735	0.0909	0.0687	0.0791	0.0935	<u>0.1237</u>	**0.1328**	7.36 %
	Recall@50	0.2298	0.2728	0.3138	0.2105	0.2859	0.2861	<u>0.3551</u>	**0.3711**	4.51 %
	NDCG@50	0.0875	0.1036	0.1269	0.0768	0.1123	0.1149	<u>0.1553</u>	**0.1655**	6.57 %
LFM-1b	Recall@10	0.0907	0.076	0.0829	0.0785	0.0837	0.0997	<u>0.1152</u>	**0.1282**	11.28 %
	NDCG@10	0.1895	0.1419	0.1496	0.143	0.1498	0.1906	<u>0.2231</u>	**0.2463**	10.40 %
	Recall@20	0.1479	0.1233	0.1311	0.1292	0.1318	0.1518	<u>0.1737</u>	**0.1861**	7.14 %
	NDCG@20	0.1867	0.1435	0.1496	0.1455	0.15	0.1837	<u>0.2146</u>	**0.2324**	8.29 %
	Recall@50	0.2628	0.2202	0.233	0.2342	0.2376	0.2569	<u>0.2842</u>	**0.2922**	2.81 %
	NDCG@50	0.2108	0.1662	0.174	0.1772	0.1763	0.2044	<u>0.2332</u>	**0.2479**	6.30 %

improvement of the best score compared to the second. we can find the following results:

- Overall, our KAUR yielded the best performance on all datasets. In particular, KAUR outperforms the strongest baseline KGIN, and improves by 2.22 %, 5.47 %, and 11.28 % on the Recall@10 indicator. This demonstrates the effectiveness of KAUR, which we attribute to the following two points: (1) The fuzzy interest set is extracted from the CKG, and the influence of similar users (or items) during user interaction is considered. (2) Contrastive learning is adopted, which more effectively captures information from higher-order neighbors and further strengthens node representation. The entire process reinforces the user's representation and is realistic. Additionally, we find that KAUR has a higher boost for music and books than movies, because of Amazon-book and Last.FM are sparser than MovieLens-1M.
- KAUR, KGIN, KGAT, KGNNLS, and KGCN are all models that use GNN (GAT). Among them, KAUR, KGIN, and KGAT are much better than other models that do not use GNN. However, on the MovieLens-1M and Last.FM datasets, the performance of KGNNLS is only better than that of RippleNet and MKR, and KGCN is only better than RippleNet, both of which are inferior to CKG. This is because KGCN and KGNNLS only strengthen the representation of project nodes through the entity nodes of the knowledge graph, ignoring the representation of user nodes and explicit cooperation signals, and do not make full use of the knowledge graph. KAUR, KGIN, and KGAT model users well and all effectively use of the relations set in the knowledge graph.
- As a model without GNN, CFKG outperforms RippleNet and MKR on MovieLens-1M and Last.FM datasets. Similar to KAUR, KGIN, and KGAT, CFKG integrates user behavior and item knowledge into a unified graph structure. And CFKG explores paths in the graph embedding space to construct reasonable explanations for items. To our surprise, RippleNet performs slightly better than KGCN, KGNNLS, CFKG on the Last.FM. The reason for this is likely that the average number of user interactions is much larger than the first two datasets. In this scenario, RippleNet may be more suitable. RippleNet will automatically explore possible paths from the user's historically interacted item set to candidate items, and iteratively expand the user's interests on the knowledge graph.

5.4. Ablation study

KAUR learns to strengthen nodes representation by comparing the information of nodes and neighbor nodes in CKG, and models user fuzzy interest sets and similar users (similar items). To test the effectiveness of contrastive learning, user fuzzy interest sets, and similar users (similar items), we remove the corresponding modules of these three parts from the model structure to test the performance of KAUR. We consider the following KAUR variants for comparison in Table 4.

- w/o F: This variant removes the fuzzy set of interests.
- w/o S: This variant removes similar users (similar items).
- w/o C: This variant removes contrastive learning.

As can be seen in Table 4, all our proposed techniques or modules help to improve the final performance. Taken together, the variables *w/o F* and *w/o C* have the worst performance, indicating the importance of fuzzy interest sets and contrastive learning for user representation enhancement and CKG node representation, respectively. The result of variable *w/o S* shows that it is still nec-

Table 4
Ablation Study of the KAUR.

Dataset	Metric	*w/o F*	*w/o S*	*w/o C*	KAUR
MovieLens-1M	Recall@10	0.167	0.1724	0.1708	**0.1746**
	NDCG@10	0.2647	0.2693	0.2677	**0.2709**
	Recall@20	0.2543	0.2632	0.2597	**0.266**
	NDCG@20	0.2688	0.2747	0.2726	**0.2768**
	Recall@50	0.4109	0.4204	0.4189	**0.4258**
	NDCG@50	0.3058	0.3125	0.311	**0.316**
Amazon-book	Recall@10	0.1569	0.1498	0.1499	**0.1587**
	NDCG@10	0.1065	0.1023	0.1016	**0.1072**
	Recall@20	0.2364	0.2285	0.2267	**0.2376**
	NDCG@20	0.1306	0.1261	0.1249	**0.1314**
	Recall@50	0.3688	0.3619	0.365	**0.3701**
	NDCG@50	0.1614	0.1594	0.1592	**0.1643**
LFM-1b	Recall@10	0.1147	0.1216	0.1109	**0.1282**
	NDCG@10	0.227	0.2383	0.21	**0.2463**
	Recall@20	0.1702	0.1793	0.166	**0.1861**
	NDCG@20	0.215	0.2251	0.2017	**0.2324**
	Recall@50	0.2798	0.2383	0.2724	**0.2922**
	NDCG@50	0.2331	0.2417	0.2218	**0.2479**

essary to capture the influence of semantic neighbors (similar nodes) to further enhance the user representation, which is also in line with the actual situation. Although the performance of *w/o S* is the best of the three variables, it cannot be said that *w/o S* is less important than *w/o F* and *w/o C*. This is because for convenience, our modeling of user semantic neighbors is not fine-grained enough, and the dataset also does not have sufficient connectivity. Taken together, these ablation studies confirm that all three components of KAUR are useful to augment user representations with fuzzy interest levels with semantic neighbors (similar nodes) and improve performance through contrastive learning. The three components complement each other to significantly improve recommendation results.

5.5. Sensitivity analysis

5.5.1. Impact of fuzzy interest set

In order to analyze the effect of the number of fuzzy interest sets, we change the size in the range of [0, 2, 3, 4, 5]. In Fig. 3, the change curves of Recall@20 and NDCG@20 on the three datasets are shown.

We can find from Fig. 3 that changes in the number of fuzzy interest sets affect the model's performance, which also shows that the modeling of fuzzy interests is effective. When$|f| = 3$, KAUR performs best on MovieLens-1 M and LFM1b. When$|f| = 5$, KAUR performs best on Amazon-book. On the whole, the higher the number of fuzzy interest sets in some cases, the better model's performance. But sometimes too much detail (the larger the number) of the fuzzy interest set division means that the elements in this set cannot effectively contain useful information. The two most extreme ways are: (1) When the number of fuzzy interests is 0, it corresponds to *w/o F* in Section 5.4. Through the ablation experiments, we know that the performance of KAUR is degraded when the fuzzy interest sets do not exist. (2) When the number of fuzzy interests is equal to the number of relations, the fuzzy interest sets correspond to the relations as well and the fuzzy interests are too clear to represent the intentions of all users. To sum up, we need to determine the number of fuzzy interests in an interval while ensuring that they are independent of each other, to improve the performance of the model.

5.5.2. Impact of the semantic neighbors

To study the effect of semantic neighbors (similar users or similar items), we vary the size in the range [0,100, 500, 800, 1500]. In Fig. 4, the change curves of Recall@20 and NDCG@20 on the three datasets are shown.

We found that setting different k values can significantly affect model performance. When $k = 0$ the experimental results correspond to *w/o S* in Section 5.4.1. We found that on Amazon-book and LFM-1b, the role of semantic neighbors is very obvious. One possible reason is that the two datasets are sparse. Similar semantic neighbors can be better divided during the clustering process, and the influence of centroid neighbors can be effectively learned. This also illustrates the importance of semantic neighbors. On the MovieLens1M dataset, sometimes introducing semantic neighbors may also have a negative impact: when$k = 100$, the results are not very ideal. This is because the MovieLens-1 M dataset is not very sparse compared to the other two datasets. When the value of k is small, the semantic neighbor division is too much detailed, which may introduce additional noise when learning the influence of centroid neighbors. We also find in Fig. 4 that the performance of KAUR degrades for large values of k for all datasets. This is because as the value of k increases, the division of semantic neighbors will be very rough, and the influence of learning semantic neighbors becomes ineffective. This situation is also in line with real life. For example, users with similar hobbies tend to be a small number of people, not the vast majority. Therefore, the choice of the k value should be determined according to the situation, in order to enhance the user representation further and improve the system performance.

5.5.3. Impact of the layer

We vary the number of propagation layers of KAUR in the range {1, 2, 3, 4} to study the effect of different layers. The performance comparisons on movie, book and music datasets are shown in Table 5.

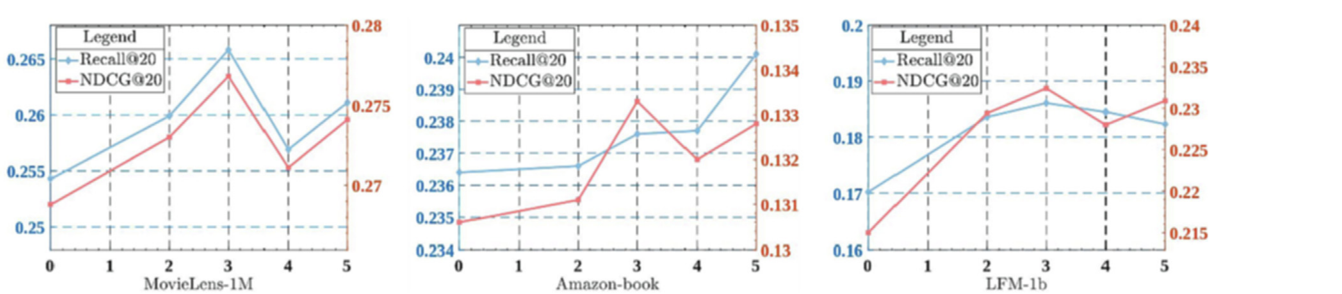


Fig. 3. Impact of fuzzy interest number.

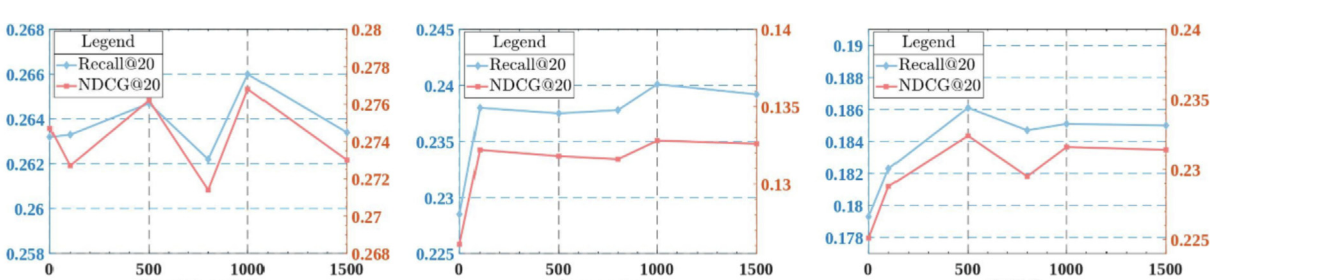


Fig. 4. Impact of the semantic neighbors.

Table 5
Ablation Study of the KAUR.

	MovieLens-1M		Amazon-book		LFM-1b	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
Layer = 1	0.2657	0.2731	0.2334	0.129	0.1819	0.2286
Layer = 2	**0.266**	**0.2768**	**0.2401**	**0.1328**	**0.1861**	**0.2324**
Layer = 3	0.2489	0.2535	0.2325	0.1299	0.1756	0.2092
Layer = 4	0.2277	0.2388	0.2227	0.1233	0.1635	0.1869

We can find from Table 5 that when Layer = 2, all three datasets achieve the best performance. This shows that increasing the appropriate number of layers can improve performance. We attribute this to higher-order neighbor information among users, items, and entities, which can enhance modeling. As the number of layer increases, the model performance decreases sequentially because more stacking used only to introduce more noise. According to different scenarios, we need to reasonably control the number of propagation layer to maximize the use of neighbor information.

5.5.4. Impact of loss weight

From Eq. (6), (11), and (17), there are four parameters of the loss functions, which are α, β, γ and λ. In order to measure the impact of various parameters on the model performance, we conducted experiments on the MovieLens-1 M dataset. According to experience, we vary α, β, γ and λ in $\{10^{-8}, 10^{-7}, 10^{-6}, 10^{-4}, 10^{-2}, 1\}$. To study the influence of a single parameter, we must control the other three parameters unchanged. For example, we set the value of parameter β and γ to10^{-7}, the value of parameter λ to10^{-4}, and observe the change of model performance by varying α. The sensitivities of these hyper-parameters are shown in Fig. 5.

From Fig. 5, we can find that when other parameters are determined, the change of α will affect the performance of the model. But this effect is less than that caused by the change of β, γ and λ. Because KAUR is designed to enhance user modeling, while α describes the weight of a subtask loss function under user enhancement, varying the value of α will not have a huge impact on the performance of the entire model. However, β and γ are the weights of loss functions in item modeling and CKG modeling, respectively. When the values of β and γ are increased, that is, too much attention is paid to the task of item modeling and CKG modeling, the model performance will be greatly reduced. λ is the parameter for $L2$ regularization. Compared with the other three parameters, it is not so important, but too large will also affect the performance of the model. To sum up, we need to choose appropriate values for parameters to ensure the best performance.

5.5.5. Other GNN Backbones

Since we use the LightGCN-based architecture for message aggregation and propagation on the CKG, i.e., the operations on the CKG are model-independent, we will further test its performance with other GNN-based architectures. The results are shown in Fig. 6. From the figure, we can observe that using LightGCN-based architecture consistently achieves the best performance on the three datasets, which further validates the effectiveness of our proposed method. In addition, the performance drops sequentially on NGCF and SpectralCF architectures. This is because SpectralCF performs eigen-decomposition on the adjacency matrix of bipartite graphs to discover connections between users and items. It is not suitable for operating on CKGs and feigen-decomposition will lead to high computational complexity, which unsuitable for large-scale recommendation scenarios. NGCF explicitly encodes

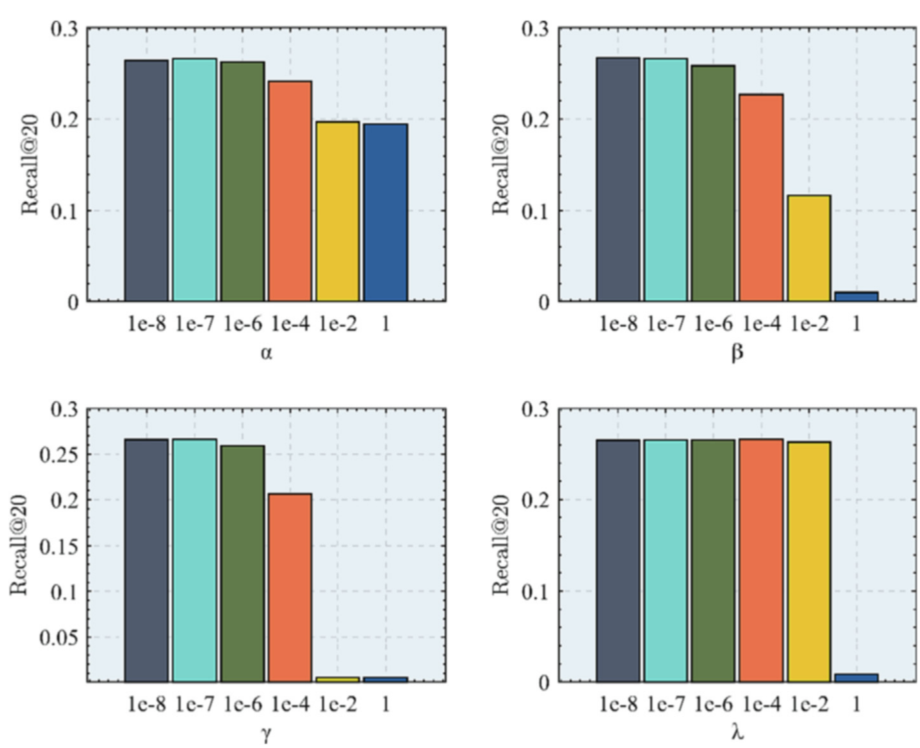


Fig. 5. Performance comparison w.r.t different α, β, γ and λ.

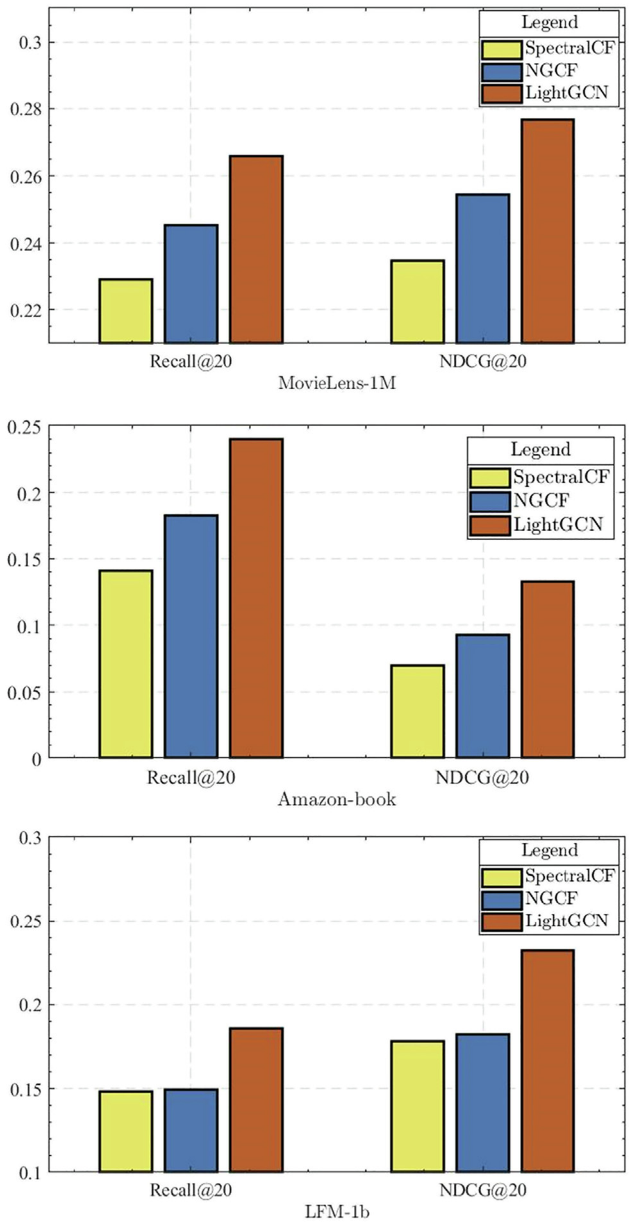


Fig. 6. Impact of Other GNN Backbones.

cooperative signals in the form of higher-order connectivity through embedding propagation. However, when designing NGCF, two components are used: feature transformation and nonlinear activation, which are experimentally proved to increase the training difficulty and reduce the recommendation performance. Compared to the first two GNN architectures, LightGCN is lighter and more efficient. It only contains the neighborhood aggregation component, which is suitable for message dissemination and aggregation in CKGs.

6. Conclusion and future work

In this work, we propose a novel Knowledge Augmented User Representation Model, KAUR. This model emphasizes the importance of users in the recommendation task, and introduces the idea of contrastive learning into CKGs. Our idea comes from the fact that in the past behavior of users interacting with items, we found that users will initially have a general preference, and users will be affected by similar users and similar items when selecting items. Therefore, we strengthen the users representation from two aspects: internal and external factors: (1) Fuzzy interest extraction. The users' intrinsic preference is modeled using the relation set in the CKG, which is common to all users. (2) User-enhanced representation. We use a clustering algorithm to find out the centroids of similar user sets and similar item sets, and compare them with user nodes or item nodes to capture potential impacts.

Currently, our user modeling is static. The influence of users' internal and external factors changes dynamically over time. In future work, we hope to introduce time series information into the process of user modeling, explore the behavior habits of users at different time stages, and reveal the changing laws of users' fuzzy interests and external influences. In addition, to explore of semantic neighbors (similar nodes), we hope to introduce the user social network and item relationship network to capture the influence of semantic neighbors in a more fine-grained way.

CRediT authorship contribution statement

Yubin Ma: Conceptualization, Methodology, Software, Writing – original draft. **Xuan Zhang:** Supervision, Writing – review & editing. **Chen Gao:** Visualization, Investigation. **Yahui Tang:** Writing – review & editing. **Linyu Li:** Writing – review & editing. **Rui Zhu:** Writing – review & editing. **Chunlin Yin:** Writing – review & editing.

Data availability

The authors do not have permission to share data.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant No. 61862063, 61502413, 61262025; the National Social Science Foundation of China under Grant No. 18BJL104; the Science Foundation of Young and Middle-aged Academic and Technical Leaders of Yunnan under Grant No. 202205AC160040; the Science Foundation of Yunnan Jinzhi Expert Workstation under Grant No. 202205AF150006; the Natural Science Foundation of Key Laboratory of Software Engineering of Yunnan Province under Grant No. 2020SE301; the Postgraduate Research and Innovation Foundation of Yunnan University under Grant (ZC-22221334); the Science Foundation of "Knowledge driven intelligent software engineering innovation team";

References

[1] Qingyao Ai, Vahid Azizi, Xu Chen, Yongfeng Zhang, Learning heterogeneous knowledge base embeddings for explainable recommendation, Algorithms 11 (9) (2018) 137.
[2] Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Antonio Ferrara, Alberto Carlo Maria Mancino, Sparse feature factorization for recommender systems with knowledge graphs, in: Fifteenth ACM Conference on Recommender Systems, 2021, pp. 154–165.
[3] Ting Bai, Ji-Rong Wen, Jun Zhang, Wayne Xin Zhao, A neural collaborative filtering model with interaction-based neighborhood, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 1979–1982.
[4] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263, 2017.

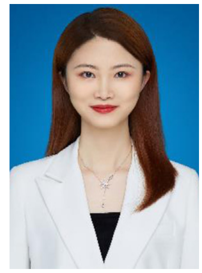
[5] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, Adv. Neural Inf. Process. Syst. 26 (2013).
[6] Yixin Cao, Xiang Wang, Xiangnan He, Hu Zikun, Tat-Seng Chua, Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences, in: The World Wide Web Conference, 2019, pp. 151–161.
[7] Bo Chen, Wei Guo, Ruiming Tang, Xin Xin, Yue Ding, Xiuqiang He, Dong Wang, Tgcn: Tag graph convolutional network for tagaware recommendation, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 155–164.
[8] H. Chen, F. Qian, J. Chen, S.u. Zhao, Y. Zhang, Fgrs: Capture user fine-grained preferences through attribute information for recommender systems, Neurocomputing 458 (2021) 195–203.
[9] Hongxu Chen, Yicong Li, Xiangguo Sun, Xu Guandong, Hongzhi Yin, Temporal meta-path guided explainable recommendation, in: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, 2021, pp. 1056–1064.
[10] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Hu Linmei, Biyu Ma, Yongliang Li, Metapath-guided heterogeneous graph neural network for intent recommendation, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 2478–2486.
[11] Xavier Glorot, Yoshua Bengio, Understanding the difficulty of training deep feedforward neural networks, Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Pages 249–256. JMLR Workshop and Conference Proceedings, 2010.
[12] Jibing Gong, Shen Wang, Jinlong Wang, Wenzheng Feng, Hao Peng, Jie Tang, Philip S. Yu, Attentional graph convolutional networks for knowledge concept recommendation in moocs in a heterogeneous view, in: Proceedings of the 43rd international ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 79–88.
[13] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, Meng Wang, Lightgcn: Simplifying and powering graph convolution network for recommendation, in: Proceedings of the 43rd international ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 639–648.
[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Hu Xia, Tat-Seng Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 173–182.
[15] Hu Binbin, Chuan Shi, Wayne Xin Zhao, Philip S. Yu, Leveraging meta-path based context for top-n recommendation with a neural co-attention model, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 1531–1540.
[16] H.u. Bojing, Y. Ye, Y. Zhong, J. Pan, H.u. Maosheng, Transmkr: Translation-based knowledge graph enhanced multi-task point-of-interest recommendation, Neurocomputing 474 (2022) 107–114.
[17] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, Edward Y. Chang, Improving sequential recommendation with knowledge-enhanced memory networks, in: The 41st international ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 505–514.
[18] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, Xuan Zhu, Learning entity and relation embeddings for knowledge graph completion, Twenty-ninth AAAI Conference on Artificial Intelligence, 2015.
[19] Zihan Lin, Changxin Tian, Yupeng Hou, Wayne Xin Zhao, Improving graph collaborative filtering with neighborhood-enriched contrastive learning, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 2320–2329.
[20] X. Ma, Z. Gao, H.u. Qian, M. Abdelhady, Contrastive knowledge graph attention network for request-based recipe recommendation, in: ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2022, pp. 3278–3282.
[21] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748, 2018.
[22] L. Sang, X.u. Min, S. Qian, W.u. Xindong, Knowledge graph enhanced neural collaborative filtering with residual recurrent network, Neurocomputing 454 (2021) 417–429.
[23] J. Shuai, K. Zhang, L. Wu, P. Sun, R. Hong, M. Wang, and Y. Li. A review-aware graph contrastive learning framework for recommendation. 2022.
[24] Su Yixin, Rui Zhang, Sarah M. Erfani, Junhao Gan, Neural graph matching based collaborative filtering, in: Proceedings of the 44th international ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 849–858.
[25] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. arXiv preprint arXiv:1908.01000, 2019.
[26] Rui Sun, Xuezhi Cao, Yan Zhao, Junchen Wan, Kun Zhou, Fuzheng Zhang, Zhongyuan Wang, and Kai Zheng. Multi-modal knowledge graphs for recommender systems. in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pages 1405–1414, 2020.
[27] Chang-You Tai, Meng-Ru Wu, Yun-Wei Chu, Shao-Yu Chu, and LunWei Ku. Mvin: Learning multiview items for recommendation. in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 99–108, 2020.
[28] Ke Tu, Peng Cui, Daixin Wang, Zhiqiang Zhang, Jun Zhou, Yuan Qi, and Wenwu Zhu. Conditional graph attention networks for distilling and refining knowledge graphs in recommendation. in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pages 1834–1843, 2021.
[29] Hongwei Wang, Jia Wang, Miao Zhao, Jiannong Cao, and Minyi Guo. Joint topic-semantic-aware social recommendation for online voting. in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pages 347–356, 2017.
[30] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. in: Proceedings of the 27th ACM international conference on information and knowledge management, pages 417–426, 2018.
[31] H. Wang, F. Zhang, Xing Xie, Minyi Guo, Dkn: Deep knowledge-aware network for news recommendation, in: Proceedings of the 2018 World Wide Web Conference, 2018, pp. 1835–1844.
[32] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, Zhongyuan Wang, 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, 2019, pp. 968–977.
[33] Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Multi-task feature learning for knowledge graph enhanced recommendation. in: The world wide web conference, pages 2000–2010, 2019.
[34] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. Knowledge graph convolutional networks for recommender systems. in: The world wide web conference, pages 3307–3313, 2019.
[35] X. Wang, X. He, Y. Cao, M. Liu, T.-S. Chua, Kgat: Knowledge graph attention network for recommendation, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 950–958.
[36] X. Wang, X. He, M. Wang, F. Feng, T.-S. Chua, Neural graph collaborative filtering, in: Proceedings of the 42nd international ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 165–174.
[37] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, Tat-Seng Chua, Learning intents behind interactions with knowledge graph for recommendation, in: In ProceedIngs of the Web Conference 2021, 2021, pp. 878–887.
[38] Xiang Wang, Dingxian Wang, Xu Canran, Xiangnan He, Yixin Cao, Tat-Seng Chua, Explainable reasoning over knowledge graphs for recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 5329–5336.
[39] Z.e. Wang, G. Lin, H. Tan, Q. Chen, X. Liu, 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. 219–228.
[40] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 28, 2014.
[41] Max Welling and Thomas N Kipf. Semi-supervised classification with graph convolutional networks. in: J. International Conference on Learning Representations (ICLR 2017), 2016.
[42] Wu Jiancan, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, Xing Xie, Self-supervised graph learning for recommendation, in: Proceedings of the 44th international ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 726–735.
[43] L. Xia, C. Huang, Y. Xu, J. Zhao, D. Yin, and J. X. Huang. Hypergraph contrastive collaborative filtering. 2022.
[44] Yikun Xian, Fu Zuohui, Shan Muthukrishnan, Gerard De Melo, Yongfeng Zhang, Reinforcement knowledge graph reasoning for explainable recommendation, in: Proceedings of the 42nd international ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 285–294.
[45] Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. Knowl edge graph contrastive learning for recommendation. arXiv preprint arXiv:2205.00976, 2022.
[46] Junliang Yu, Xin Xia, Tong Chen, Lizhen Cui, Nguyen Quoc Viet Hung, and Hongzhi Yin. Xsimgcl: Towards extremely simple graph contrastive learning for recommendation. arXiv preprint arXiv:2209.02544, 2022.
[47] Yu Xiao, Xiang Ren, Yizhou Sun, Gu Quanquan, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, Jiawei Han, Personalized entity recommendation: A heterogeneous information network approach, in: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, 2014, pp. 283–292.
[48] F. Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, Wei Ying Ma, Collaborative knowledge base embedding for recommender systems, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 353–362.
[49] H. Zhao, Quanming Yao, Jianda Li, Yangqiu Song, Dik Lun Lee, Meta-graph based recommendation fusion over heterogeneous information networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 635–644.
[50] Wayne Xin Zhao, Mu Shanlei, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Lu Yujie, Hui Wang, Changxin Tian, et al., Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 4653–4664.
[51] Lei Zheng, Lu Chun-Ta, Fei Jiang, Jiawei Zhang, Philip S. Yu, Spectral collaborative filtering, in: Proceedings of the 12th ACM Conference on Recommender Systems, 2018, pp. 311–319.

[52] Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, Yu Jingsong, Improving conversational recommender systems via knowledge graph based semantic fusion, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1006–1014.
[53] P. Zhu, D. Cheng, S. Luo, F. Yang, Y. Luo, W. Qian, A. Zhou, Si-news: Integrating social information for news recommendation with attention-based graph convolutional network, Neurocomputing 494 (2022) 33–42.
[54] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. arXiv preprint arXiv:2006.04131, 2020.
[55] Ding Zou, Wei Wei, Xian-Ling Mao, Ziyang Wang, Minghui Qiu, Feida Zhu, and Xin Cao. Multi-level cross-view contrastive learning for knowledge-aware recommender system. arXiv preprint arXiv:2204.08807, 2022.
[56] Ding Zou, Wei Wei, Ziyang Wang, Xian-Ling Mao, Feida Zhu, Rui Fang, and Dangyang Chen. Improving knowledge-aware recommendation with multi-level interactive contrastive learning. arXiv preprint arXiv:2208.10061, 2022.

Yubin Ma is currently pursing for a master's degree in software engineering at Yunnan University. His current research interests include recommender systems, graph neural network and knowledge graphs.

Xuan Zhang received the B.S. and M.S. degrees in computer science from Yunnan University, Yunnan. She received the Ph.D. degree in system analysis and integration from Yunnan University, Yunnan. She is a professor with the School of Software, Yunnan University, Yunnan, China. She is author of 3 books and more than 90 articles. She has been principal investigator for more than 30 national, provincial, and private grants and contracts. She is the core scientist of Yunnan Key Laboratory of Software Engineering and Yunnan Software Engineering Academic Team. Her research interests include knowledge engineering, recommendation system, natural language processing, computer vision,software engineering.

Chen Gao received his Master degree in software engineering from Yunnan University. He is currently a PHD student in the School of Information Science and Engineering, Yunnan University. His research interests mainly include natural language processing and knowledge graph, especially information extraction. Currently published three papers in the field of information extraction.



Yahui Tang received a B.S degree in computer science and technology from Kunming University of Science and Technology, Kunming, China, in 2016. And received a Ph.D. degree in computer science from Yunnan University, in 2022. Her main research interests include process mining, business process, and software process.

LinYu Li is currently studying for a master's degree at the School of Software, Yunnan University.His current research interests include knowledge graph completion, knowledge graph embedding,and graph neural network.

Rui Zhu (Member, IEEE) received the Ph.D. degree in software engineering from Yunnan University, Kunming, China, in 2016. He is currently a Head and Associate Professor of Artificial Intelligence with the School of Software, Yunnan University. His current research interests include intelligent transportation, blockchain technology, and machine learning.

Chunlin Yin received his M. S. degree from Yun Nan University in 2017. He is currently work for the Electric Power Research Institute of Yunnan Power Grid as an engineer. His main research interests are transfer learning.