Knowledge-Rich Influence Propagation Recommendation Algorithm Based on Graph Attention Networks

Yuping Yang[1], Guifei Jiang[1,2,3](✉), and Yuzhi Zhang[1,2,3]

[1] Nankai University, Tianjin, China
yypSandra@mail.nankai.edu.cn {g.jiang,zyz}@nankai.edu.cn
[2] Haihe Laboratory of Information Technology Application Innovation (HL-IT), Tianjin, China
[3] Tianjin Key Laboratory of Operating System, Tianjin, China

Abstract. One of the biggest challenges in Recommendation Algorithms (RA) is how to obtain user and item embeddings from sparse interaction history. To take this challenge, most graph neural network based RAs explicitly incorporate high-order collaborative filtering signals on the user-item bipartite graph with either multi-layer semantics on the Knowledge Graph (KG) or multi-level neighbors on the social network. However, none of them fully integrate these three types of graph-structured data, which decreases embeddings' precision. Based on this consideration, this paper integrates the three types of data by proposing a knowledge-rich influence propagation RA based on the graph attention mechanism. Specifically, in the semantic propagation, we categorize user preferences into deep interest obtained by multiple graph attention message propagations on related KG parts, and shallow interest generated from the interaction history. Moreover, the influence weight between items is determined by the number of co-interactions and the semantic similarity. These two factors as well as social relations together decide the influence weight between users. With these influence weights, final user and item embeddings are calculated through multi-layer message propagation. The experimental results show that the proposed recommendation algorithm outperforms several compelling baselines on six scaled-down real-world datasets. This work has confirmed the effectiveness of combining these three types of data to increase RAs' coverage and accuracy.

Keywords: Recommendation Algorithm · Knowledge Graph · Social Network · Graph Attention Network · User Preference

1 Introduction

As one of the most popular recommendation Algorithms (RA), Collaborative Filtering (CF) is able to detect inter-user or inter-item similarities to make recommendations for users accordingly. Since user-item interaction history is represented as a bipartite graph, Graph Neural Networks (GNN) are utilized to predict the missing user-item interactions [3]. However, user preference inference based only on the sparse user-item interaction history yields insufficient and coarse-grained results. The widely used approach of solving the sparsity problem is to combine side information collected from the system. Side information includes attributes in the form of text or images, social relationships among users and users' comments on items and other contents [10]. In particular, as shown in Fig. 1, GNNs can be used to fully merge two kinds of graph-structured side information: triplets between items and attributes on the Knowledge Graph (KG) and user-user connections on the social network with the bipartite graph [4].

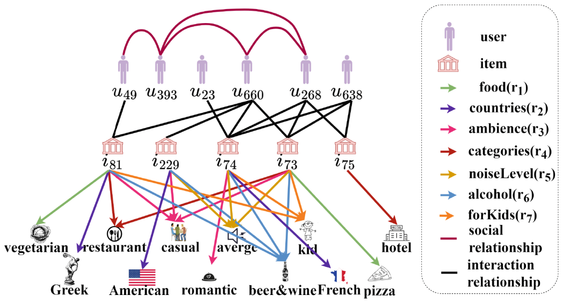


Fig. 1. An example with three kinds of graph-structured data in Yelp2018

Specifically, GNN-based collaborative filtering RAs usually combine the bipartite graph with either the KG or the social network. The former incorporates rich semantics to enhance the accuracy and interpretability of recommendations, and the latter leverages high-order social effects to modify user interests [17,21]. Yet, neither of them is fully satisfactory as they mainly rely on limited data. On the one hand, the KG-based RAs fail to account for user attribute-level interests despite that users choose items for specific reasons. On the other hand, they neglect to explicitly model high-order CF signals reflecting directly inter-user and inter-item influences [19]. In addition, they do not appreciate the importance of social diffusion. Without mixing high-order CF signals expressing user interest similarities with social influences, the social network-based RAs also fail to take advantage of available item attributes to infer user preferences.

To address these issues, this paper proposes a **K**nowledge-Rich **I**nfluence **P**ropagation **Rec**ommendation (**KIPRec**) that seamlessly fuses the KG, the social network, and the user-item bipartite graph. To obtain reasonably precise user and item embeddings, multi-layer Graph ATtention networks (GAT) are employed to aggregate neighbor messages. There are two main components in KIPRec. The first part is to explore semantic information, where user preferences are divided into shallow-level interests derived from interacted item sequences and deep-level interests obtained by aggregating semantics from their KG subgraphs. The second part is to propagate influence information. User-user connections in the social network are combined with user-user co-interactions from the bipartite graph to organize an influence graph and refine user interests. In particular, the main contributions of this paper are summarized as follows:

- A comprehensive RA is proposed by integrating three types of side information with graph structures to alleviate the sparsity problem in RAs.
- The validity of the proposed RA is fully and extensively investigated in two commonly used recommendation scenarios: Click-Through-Rate (CTR) prediction and Top-K recommendation. Specifically, for the Recall@20 metric, our RA improves by 1.92% and 5.64% on the Last-FM and Yelp2018 datasets, respectively, compared to the state-of-the-art (SOTA) KG-based RA, and it improves by 10.05% and 11.54% over the SOTA social network-based RA, respectively.
- Ablation experiments are conducted to analyze the functions of KIPRec's two main components.

The rest of this paper is organized as follows: Sect. 2 reviews the related work on GNN-based collaborative filtering RAs based on the KG or the social network. Section 3 presents the main problem addressed in this paper. The structure of proposed model is detailedly explained in Sect. 4. Extensive comparisons among KIPRec and its baselines as well as other studies are provided in Sect. 5. Finally, we conclude this paper with a discussion of the future directions in Sect. 6.

2 Related Work

By explicitly modeling high-order CF signals on the user-item bipartite graph via multiple forms of message passing, the current graph convolutional neural network-based methods such as LightGCN [5], GAT-based methods, and graph sampling aggregate network-based methods have achieved significant improvements compared with the traditional matrix decomposition methods [12] and most deep learning-based methods [26]. However, these GNN-based approaches fail to satisfactorily perform the task of acquiring useful information on heterogeneous graphs with multiple node or edge types. We will discuss the most related work in terms of the following aspects.

GNN-based RAs on the KG are divided into the following three categories: embedding-based methods, path-based methods, and propagation-based methods [6]. Items in the bipartite graph are mapped to entities in the KG, hence embedding-based approaches like MKR [14] combine the KG embedding algorithms and the recommendation module in a certain sequence. However, it might be difficult to accurately correlate the embeddings learned by these two modules. Path-based approaches, such as KPRN [18], merge the bipartite graph and the KG into a unified graph according to the correlations between items and entities. Paths between users and items are extracted from the graph to generate embeddings. Yet, path-based approaches require expert knowledge to decrease space and time consumption for processing huge volumes of paths.

Differently, propagation-based approaches apply GNNs directly to the KG, deriving node semantic embeddings by message propagation. Typical procedures for this sort of algorithms are outlined below. RippleNet [13] initializes item embeddings and refines user embeddings based on related multi-layer triplets. Starting from the user-interacted items, these triplets are obtained by aligning head and tail entities in triplets on the KG. KGCN [15] first randomly initializes user embeddings but constructs item embeddings for the target user by aggregating the neighbor messages from the items' receptive fields with GAT. By correlating intents with relations on the KG, KGIN [17] provides an explicit interpretation of intentions and a finer granularity of user and item embeddings. Besides exploring semantics in the KG, CKAN [19] additionally employs CF signals on the item side. CKAN first refreshes entity embeddings on the KG using a multi-layer GAT, then considers embeddings of the user-interacted items on the KG as the user embedding, and finally pools embeddings of all items on the KG that share at least one users on the bipartite graph into the item embedding. In this paper, the semantic propagation component adopts a KGCN-like technique to construct semantic propagation trees and refine embeddings, which provides a reasonable explanation of users' preferences and items' features. CKAN does not involve the interest similarities between users with co-interactions.

GNN-based RAs on the social network aim to rationally leverage the bipartite graph and the social network with diverse topologies to strengthen user interests. GATs are employed to evaluate the impacts of social connections and CF signals on each user's interest, as some users are easily swayed by their peers while others tend to adhere to their own preferences [1]. GraphRec [2] only considers users' first-order CF signals and social neighbors, and Diffnet++ [21] takes into account users' high-order contexts. Furthermore, some approaches like DISGCN [9] are proposed to disentangle the correspondence between the social connection and interest similarity into multiple dimensions, and some self-supervised approaches such as SEPT [24] are designed to eliminate fragile and noisy social relationships.

In addition, GSIRec [8] is equipped with a knowledge assistance module as the bridge between knowledge-aware recommendation and social-aware recommendation. However, this multi-task method may encounters the problem of negative transfer when knowledge gained in one task lowers performance in another [27]. Social-RippleNet [7] additionally utilizes the KG to adjust item embeddings and applies the social network to modify user preferences. Yet, users' profiles would be more transparent if the KG were used to extend the tags that define users.

3 Problem Formulation

In this paper, the user-item bipartite graph is denoted as a binary-valued symmetric matrix $Y \in \mathbb{R}^{m \times n}$ as usual, where $y_{ui} = y_{iu} = 1$ indicates there is an observed interaction between a user $u \in U$ and an item $i \in I$. The KG $\mathcal{G} = \{(\mathcal{h}, \mathcal{r}, t) \mid \mathcal{h}, t \in \mathcal{E}, \mathcal{r} \in \mathcal{R}\}$ is a directed graph made up of the entity set

$\mathcal{E}$ and relation set $\mathcal{R}$, where $(\hbar, \imath, t)$ refers to a triplet containing a head entity $\hbar$, a tail entity t and a relation $\imath$ connecting them. Most items in the bipartite graph have corresponding entities in the KG, i.e., $I \subseteq \mathcal{E}$. The social network is an undirected graph represented by $S \in \mathbb{R}^{m \times m}$, with $s_{uu'} = s_{u'u} = 1$ indicating the existence of a following or followed relationship for users $u, u' \in U$. Given a bipartite graph Y, a KG $\mathcal{G}$ and a social network S, our model optimizes parameters in Θ to learn embeddings of user u_p and item i_q, which in turn predict the likelihood $\hat{y}_{pq}$ of the interaction between them via the function $\mathcal{F}$: $\hat{y}_{pq} = \mathcal{F}(u_p, i_q \mid Y, \mathcal{G}, S, \Theta)$. We will specify Θ in Sect. 4.4.

4 Methodology

This paper focuses on implicit binary-valued user behaviors without considering the effect of time. As mentioned before, the goal is to predict the probability of an interaction between a user-item pair (p, q). To this end, Fig. 2 and Fig. 3 illustrate the overall structure of the proposed RA. It consists of four components: an initialization layer, a semantic propagation layer, an influence propagation layer, and a prediction layer. We will elaborate them in the following.

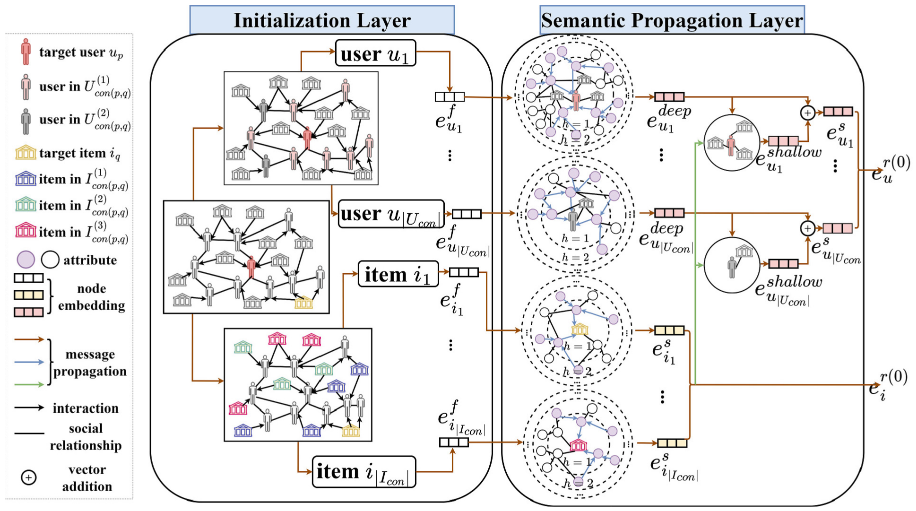


Fig. 2. The initialization layer and semantic propagation layer in KIPRec

4.1 The Initialization Layer

To begin with, we represent all nodes and edges on all graphs with trainable embeddings of a fixed length d. The notations $e^f_{u_p}$, $e^f_{i_q}$, $e^f_{\mathfrak{e}_c}$, $e^f_{r_d} \in \mathbb{R}^d$ respectively denote the free embedding of user u_p, item i_q, entity $\mathfrak{e}_c$, relation r_d. The influence propagation layer's initial embeddings are derived from the semantic propagation layer's outputs. Before proceeding to the semantic propagation layer, we need to calculate the context information for the user-item pair (p, q) in the influence propagation layer, i.e., all the users and items related to (p, q).

The inter-user and inter-item influence matrices $Y_{inf} \in \mathbb{R}^{m \times m}$ and $A_{inf} \in \mathbb{R}^{n \times n}$ are calculated by multiplying the user-item interaction matrix $Y \in \mathbb{R}^{m \times n}$ with the item-user interaction matrix $A \in \mathbb{R}^{n \times m}$. Values in Y_{inf} and A_{inf} reflect numbers of common interactions. (p, q)'s user-level context information is represented as $U_{con(p,q)} = U^1_{con(p,q)} \cup U^2_{con(p,q)} \cup U^3_{con(p,q)}$. In detail, $U^1_{con(p,q)}$ contains the users who have shared items with u_p. $U^2_{con(p,q)}$ represents the users that might share similar interests with u_p in the absence of co-interactions. And $U^3_{con(p,q)}$ denotes users who have social connections with u_p. $U^t_{con(p,q)} = \cup^K_{k=1} U^{t(k)}_{con(p,q)}$, where $U^0_{con(p,q)} = \{u_p\}$, $k \in \{1, 2, ..., K\}$ and $t \in \{1, 2, 3\}$. Each user group contains K parts, since the influence propagation procedure is repeated K times.

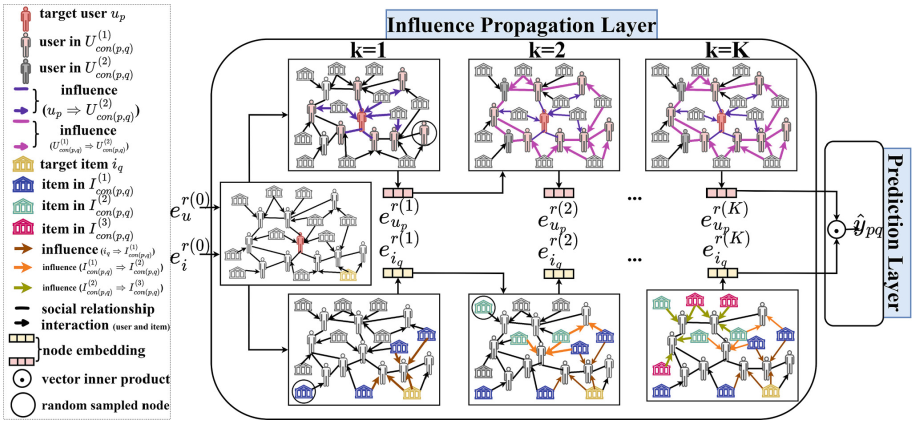

target user u_p
user in $U^{(1)}_{con(p,q)}$
user in $U^{(2)}_{con(p,q)}$
influence ($u_p \Rightarrow U^{(2)}_{con(p,q)}$)
influence ($U^{(1)}_{con(p,q)} \Rightarrow U^{(2)}_{con(p,q)}$)
target item i_q
item in $I^{(1)}_{con(p,q)}$
item in $I^{(2)}_{con(p,q)}$
item in $I^{(3)}_{con(p,q)}$
influence ($i_q \Rightarrow I^{(1)}_{con(p,q)}$)
influence ($I^{(1)}_{con(p,q)} \Rightarrow I^{(2)}_{con(p,q)}$)
influence ($I^{(2)}_{con(p,q)} \Rightarrow I^{(3)}_{con(p,q)}$)
social relationship
interaction (user and item)
node embedding
vector inner product
random sampled node
Influence Propagation Layer
k=1
k=2
k=K
$e^{r(0)}_u$
$e^{r(0)}_i$
$e^{r(1)}_{u_p}$
$e^{r(1)}_{i_q}$
$e^{r(2)}_{u_p}$
$e^{r(2)}_{i_q}$
$e^{r(K)}_{u_p}$
$e^{r(K)}_{i_q}$
$\hat{y}_{pq}$
Prediction Layer

Fig. 3. The influence propagation layer and the interaction prediction layer in KIPRec

Specifically, during the k-th influence propagation, $U^{1(k)}_{con(p,q)}$ consists of the first-order neighbors of all users in $U^{1(k-1)}_{con(p,q)}$, with column subscripts of the non-zero elements in Y_{inf}. Similarly, $U^{3(k)}_{con(p,q)}$ is formed by all users in the first-order neighbors of $U^{3(k-1)}_{con(p,q)}$ under the direction of the social matrix S. Regarding $U^{2(k)}_{con(p,q)}$, user interaction histories are frequently restricted to a narrow area, resulting in users with similar interests rarely having shared items, so we randomly choose a L fraction from $U - U^{1(k)}_{con(p,q)}$ to generate $U^{2(k)}_{con(p,q)}$.

The item-level context information of (p, q) lies in $I_{con(p,q)} = I^1_{con(p,q)} \cup I^2_{con(p,q)} \cup I^3_{con(p,q)}$. Alternatively, we use a similar method as before to obtain

$I^1_{con(p,q)}$ and $I^2_{con(p,q)}$. During the k-th influence propagation, $I^{3(k)}_{con(p,q)}$ is formed by first-order neighbors of all users in $U^{(k-1)}_{con(p,q)}$ under the direction of the interaction matrix Y. $I^3_{con(p,q)} = \cup_{k=1}^{K} I^{3(k)}_{con(p,q)}$, where $k \in \{1, 2, ..., K\}$. Figure 2 displays how to obtain the third-order context information for the target user and item.

4.2 The Semantic Propagation Layer

This section shows how to generate semantic propagation trees and update their roots' embeddings. For all users from the user contexts and items from the item contexts of the target user-item pair (p, q) described in Sect. 4.1, we establish their semantic propagation trees according to the user-item bipartite graph Y and the KG $\mathcal{G}$. Initial embeddings of these users and items are then updated through extended GAT operations on the trees.

Constructing Semantic Propagation Trees. $T^h_{q'}$ and $T^h_{p'}$ respectively represent triplet sets related to an item $i_{q'}$ in $I_{con(p,q)}$ and a user $u_{p'}$ in $U_{con(p,q)}$ during the h-th semantic propagation. From corresponding entity of $i_{q'}$ on the KG as root, related subgraph of $i_{q'}$ is extracted by aligning the tail of one triplet with the head of another for H times as described in Eq. (1), where the symbol $*$ indicates an arbitrary entity or relationship, and $(\mathscr{h}, \mathscr{r}, t)$ is a triplet on the KG $\mathcal{G}$. We limit the number of entities per layer $\mid T^h_{q'} \mid$ as w_1 to save memory space.

$$\begin{cases} T^h_{q'} = \{(\mathscr{h}, \mathscr{r}, t) \mid (\mathscr{h}, \mathscr{r}, t) \in \mathcal{G} \; and \; \mathscr{h} = i_{q'}\}, & h = 1 \\ T^h_{q'} = \left\{(\mathscr{h}, \mathscr{r}, t) \mid (\mathscr{h}, \mathscr{r}, t) \in \mathcal{G} \; and \; (*, *, \mathscr{h}) \in T^{h-1}_{q'}\right\}, & h = 2, ..., H \end{cases} \tag{1}$$

As described in Eq. (2), for each user $u_{p'}$ in $U_{con(p,q)}$, we sample w_2 attributes from the first-order entity neighbors of $u_{p'}$'s interacted items, i.e., $|T^1_{p'}| = w_2$, and connect $u_{p'}$ with these entities. $y_{p'\mathscr{h}} = 1$ means there is an interaction between $u_{p'}$ and the item corresponding to entity $\mathfrak{e}_{\mathscr{h}}$. Afterwards, the semantic propagation trees are built for users in the same way, i.e., $\mid T^h_{p'} \mid = w_1$ when $h \geq 2$. Figure 2 illustrates four semantic propagation trees with $H, w_1 = 2$ and $w_2 = 4$.

$$\begin{cases} T^h_{p'} = \{(u_{p'}, \mathscr{r}, t) \mid (\mathscr{h}, \mathscr{r}, t) \in \mathcal{G} \; and \; y_{p'\mathscr{h}} = 1\}, & h = 1 \\ T^h_{p'} = \left\{(\mathscr{h}, \mathscr{r}, t) \mid (\mathscr{h}, \mathscr{r}, t) \in \mathcal{G} \; and \; (*, *, \mathscr{h}) \in T^{h-1}_{p'}\right\}, & h = 2, ..., H \end{cases} \tag{2}$$

Implementing Semantic Propagation Operations. We represent an arbitrary user or item with the uniform symbol o. The semantic propagation procedure is executed $(H - v)$ times for each node in N^v_o. N^v_o includes all nodes on the v-th layer of the semantic tree with node o as the root, where $v \in \{0, 1, \ldots, H\}$. Assumed that the l-th message propagation is to be performed $(l \in \{1, \ldots, H-v\})$ and the g-th node on the v-th layer is $\mathfrak{e}_g \in N^v_o$, related triplets of $\mathfrak{e}_g$ are contained in $T^{v+1}_{o(g)} = \{(\mathscr{h}, \mathscr{r}, t) \mid (\mathscr{h}, \mathscr{r}, t) \in T^{v+1}_o \; and \; \mathscr{h} = \mathfrak{e}_g\}$. We first map the tail entity $\mathfrak{e}_t$'s embedding derived from the $(l-1)$-th message propagation onto the same level with relation $r_{\mathscr{r}}$ via the Hadamard product operation $\odot$. This not only decouples $\mathfrak{e}_t$'s features, but also preserves semantics along message propagation paths, as described in KGIN [17]. Further, the contribution of the t-th ($t \in \{1, ..., |T_{o(g)}^{v+1}|\}$) triplet to the head entity $\mathfrak{e}_g$, i.e., α_t^l, is valued in Eq. (3) as the similarity between the root o and $\mathfrak{e}_t$ by concatenating the two embeddings labeled with $||$ and then obtaining a numerical value via a two-layer perceptron.

$$\alpha_t^l = W_2\Bigg(LeakyReLU\bigg(W_1\big(e_o^{l-1} \,||\, (e_{r_\mathfrak{r}} \odot e_{\mathfrak{e}_t}^{l-1})\big) + b_1\bigg)\Bigg) + b_2 \tag{3}$$

Moreover, $\widetilde{\alpha}_t^l$ is the result of softmax normalization on α_t^l. The embedding of $\mathfrak{e}_g$ after the l-th message propagation results from aggregated relevant features and the embedding of $\mathfrak{e}_g$ after the $(l-1)$-th propagation, shown in Eq. (4).

$$e_{\mathfrak{e}_g}^l = e_{\mathfrak{e}_g}^{l-1} + \sum_{t=1}^{|T_{o(g)}^{v+1}|} \widetilde{\alpha}_t^l (e_{r_\mathfrak{r}} \odot e_{\mathfrak{e}_t}^{l-1}), \quad where \quad \widetilde{\alpha}_t^l = \frac{e^{\alpha_t^l}}{\sum_{t'=1}^{|T_{o(g)}^{v+1}|} e^{\alpha_{t'}^l}} \tag{4}$$

The symbol n denotes a user, an item or an entity. $\mathfrak{r}$ refers to a relation. In general, the initial embeddings of n and $\mathfrak{r}$ in the semantic propagation layer come from the initialization layer, i.e., $e_n^0 = e_n^f$ and $e_{r_\mathfrak{r}} = e_{r_\mathfrak{r}}^f$. After the propagation procedure is performed H times, semantics related to the root nodes ($u_{p'} \in U_{con(p,q)}$ and $i_{q'} \in I_{con(p,q)}$) on the semantic propagation trees, is distilled into their embeddings $e_{u_{p'}}^H$ and $e_{i_{q'}}^H$. Primary characteristics of $i_{q'}$ are indicated by $e_{i_{q'}}^s$ (equal to $e_{i_{q'}}^H$). However, as in Eqs. (5) and (6), interests of $u_{p'}$ are also expressed by user-item interactions $\mathcal{Y}_{u_{p'}}$ except for deep-level user interests marked by $e_{u_{p'}}^{deep}$ (equal to $e_{u_{p'}}^H$) [20]. The contribution $\widetilde{\beta}_{p'\mathfrak{v}}$ of $u_{p'}$'s interactive item $i_\mathfrak{v}$ to shallow interest embedding $e_{u_{p'}}^{shallow}$ is generated through an extended GAT operation and normalization. Eventually, deep interest $e_{u_{p'}}^{deep}$ and shallow interest $e_{u_{p'}}^{shallow}$ of $u_{p'}$ both contribute to the preliminary embedding $e_{u_{p'}}^s$.

$$\beta_{p'\mathfrak{v}} = W_4\bigg(LeakyReLU\big(W_3(e_{u_{p'}}^{deep} \,||\, e_{i_\mathfrak{v}}^s) + b_3\big)\bigg) + b_4 \tag{5}$$

$$e_{u_{p'}}^{shallow} = \sum_{\mathfrak{v} \in \mathcal{Y}_{u_{p'}}} \widetilde{\beta}_{p'\mathfrak{v}} e_{i_\mathfrak{v}}^s, \quad where \quad \widetilde{\beta}_{p'\mathfrak{v}} = \frac{e^{\beta_{p'\mathfrak{v}}}}{\sum_{\mathfrak{v}' \in \mathcal{Y}_{u_{p'}}} e^{\beta_{p'\mathfrak{v}'}}} \tag{6}$$

4.3 The Influence Propagation Layer

For further adjustment of user and item embeddings, the influence propagation is required to be performed over K iterations by absorbing relevant information from their K-nearest neighbors. The inital embedding of any user u in context $U_{con(p,q)}$ of u_p, indicated by $e_u^{r(0)}$, is equal to e_u^s. Similarly, each item i in context $I_{con(p,q)}^1 \cup I_{con(p,q)}^2$ of i_q has the condition $e_i^{r(0)} = e_i^s$. K-nearest neighbors of u_p and i_q are respectively included in $U_{con(p,q)}$ and $I_{con(p,q)}^1 \cup I_{con(p,q)}^2$.

During the k-th message propagation, item i_p needs to gain influence from its first-order neighbors in $I_{con(p,q)}^{(1)} = I_{con(p,q)}^{1(1)} \cup I_{con(p,q)}^{2(1)}$, where $k \in \{1, 2, ..., K\}$.

$I_{con(p,q)}^{1(1)}$ denotes items with which i_p shares users, while $I_{con(p,q)}^{2(1)}$ indicates items with similar semantics but no shared user. These items that may affect form $I_{con(p,q)}^{(1)}$. For an item $i_\mathfrak{q} \in I_{con(p,q)}^{(1)}$, we measure the semantic similarity between $i_\mathfrak{q}$ and i_q via the inner product of $e_{i_q}^{r(k-1)}$ and $e_{i_\mathfrak{q}}^{r(k-1)}$, then amplify it by multiplying with the number of their co-interactions denoted as $a_{q\mathfrak{q}}$ from A_{inf} in Eq. (7), and finally get the influence degree $\gamma_{q\mathfrak{q}}^{(k)}$. As shown in Eq. (8), the embedding of i_q after the k-th influence propagation is the total of the $(k-1)$-th outcome and the weighted sum of its first-order neighbors' embeddings during this propagation.

$$\gamma_{q\mathfrak{q}}^{(k)} = a_{q\mathfrak{q}}(e_{i_q}^{r(k-1)} \cdot e_{i_\mathfrak{q}}^{r(k-1)}) \tag{7}$$

$$e_{i_q}^{r(k)} = e_{i_q}^{r(k-1)} + \sum_{i_\mathfrak{q} \in I_{con(p,q)}^{(1)}} \widetilde{\gamma}_{q\mathfrak{q}}^{(k)} e_{i_\mathfrak{q}}^{r(k-1)}, \quad where \quad \widetilde{\gamma}_{q\mathfrak{q}}^{(k)} = \frac{e^{\gamma_{q\mathfrak{q}}^{(k)}}}{\sum_{i_{\mathfrak{q}'} \in I_{con(p,q)}^{(1)}} e^{\gamma_{q\mathfrak{q}'}^{(k)}}} \tag{8}$$

During the k-th propagation, the embedding of u_p is affected by users with social relationships, in addition to users with co-interactions or similar semantics, i.e., $U_{con(p,q)}^{(1)} = U_{con(p,q)}^{1(1)} \cup U_{con(p,q)}^{2(1)} \cup U_{con(p,q)}^{3(1)}$. Therefore in Eq. (9), the semantic similarity between u_p and $u_\mathfrak{p} \in U_{con(p,q)}^{(1)}$ is amplified by the number of co-interactions $\mathfrak{y}_{p\mathfrak{p}}$ and the social influence $\mathfrak{s}_{p\mathfrak{p}}$, where we set $\mathfrak{s}_{p\mathfrak{p}}$ to λ times the most co-interactions as described in Eq. (10) since friends tend to orient user interests more. $s_{p\mathfrak{p}} > 0$ implies that there is a social relationship between u_p and $u_\mathfrak{p}$. As described in Eq. (11), the embedding of u_p after the k-th propagation combines each neighbor's embedding based on an weight $\widetilde{\delta}_{p\mathfrak{p}}^{(k)}$ and its own embedding on the $(k-1)$-th propagation together.

$$\mathfrak{s}_{p\mathfrak{p}} = \begin{cases} 0, & if\ s_{p\mathfrak{p}} = 0 \\ \max\limits_{u_{\mathfrak{p}'} \in U_{con(p,q)}^{1(1)} \cup U_{con(p,q)}^{2(1)}} \lambda \mathfrak{y}_{p\mathfrak{p}'} (\lambda \in \mathbb{N}\ and\ 1 \le \lambda \le 10), & if\ s_{p\mathfrak{p}} > 0 \end{cases} \tag{9}$$

$$\delta_{p\mathfrak{p}}^{(k)} = (\mathfrak{y}_{p\mathfrak{p}} + \mathfrak{s}_{p\mathfrak{p}})(e_{u_p}^{r(k-1)} \cdot e_{u_\mathfrak{p}}^{r(k-1)}) \tag{10}$$

$$e_{u_p}^{r(k)} = e_{u_p}^{r(k-1)} + \sum_{u_\mathfrak{p} \in U_{con(p,q)}^{(1)}} \widetilde{\delta}_{p\mathfrak{p}}^{(k)} e_{u_\mathfrak{p}}^{r(k-1)}, \quad where \quad \widetilde{\delta}_{p\mathfrak{p}}^{(k)} = \frac{e^{\delta_{p\mathfrak{p}}^{(k)}}}{\sum_{u_{\mathfrak{p}'} \in U_{con(p,q)}^{(1)}} e^{\delta_{p\mathfrak{p}'}^{(k)}}} \tag{11}$$

4.4 The Interaction Prediction Layer

The results after K iterations of influence propagation represented by $e_{u_p}^{r(K)}$ and $e_{i_q}^{r(K)}$ are supposed as the final embeddings of u_p and i_q. They fully incorporate the semantics of the KG, social influences amongst users, and the numbers of co-interactions. The inner product of $e_{u_p}^{r(K)}$ and $e_{i_q}^{r(K)}$ predicts whether there is an interaction between them. As shown in Eq. (12), this paper applies the gradient descent to the BPR loss function to optimize parameters in Θ, including embeddings of all users, items, relationships as well as entities denoted as Θ_1 and the perceptrons' factors labeled as Θ_2, i.e., $\Theta_1 = \{e_{u_p}^f, e_{i_q}^f, e_{\mathfrak{e}_c}^f, e_{r_d}^f \mid u_p \in U, i_q \in I, \mathfrak{e}_c \in \mathcal{E}, r_d \in \mathcal{R}\}$ and $\Theta_2 = \{[W_i, b_i]_{i=1,2,3,4}\}$. We sample one item i_j without observed interaction for each observed user-item pair (u_p, i_q) in the training set, contributing to the collection $\mathcal{O}$. Moreover, the L2 regularization technique is served to avoid overfitting, where η is the related hyper-parameter.

$$\mathcal{L} = \min_{\Theta} \frac{1}{\mid \mathcal{O} \mid} \sum_{(u_p, i_q, i_j) \in \mathcal{O}} - \ln Sigmoid(\hat{y}_{pq} - \hat{y}_{pj}) + \eta \| \, \Theta \, \|_2^2 \qquad (12)$$

5 Experiments

In this section, we conduct experiments on six datasets including KGs or social networks to evaluate our proposed method and address the three questions: **RQ1**: Can KIPRec outperform the SOTA KG-based RAs and social network-based ones on real-world datasets ? **RQ2**: What exactly is the function of each component of KIPRec ? **RQ3**: How do user-user influence, item-to-item influence and multi-layer user interest affect KIPRec based on a real-world example ?

5.1 Experimental Settings

Dataset Description. Six real datasets are applied: Last-FM (music), Yelp2018 (business), Movie-Lens20M (film), Book-Crossing (book), Epinions (e-commerce) and Flickr (image). Last-FM, Movie-Lens20M and Book-Crossing are collected from CKAN [19], which contains the interaction history and the KG. Epinions and Flickr are obtained from GraphRec [2] and Diffnet++ [21] separately. KGAT [16] provides the bipartite graph and the KG of Yelp2018. We further extract its social network from the official dataset. We finally divide all users and items into four groups according to the number of their observed interactions, and narrow down all the datasets by randomly picking about 10% from each group. The statistics of all datasets are summarized in Table 1. All the datasets are randomly divided into training (60%), validation (20%) and testing (20%).

Evaluation Metrics. We investigate the performance of the proposed model with the Area Under Curve (AUC) metric in the CTR prediction, and the Recall (Recall@20) as well as Normalized Discounted Cumulative Gain (NDCG@20) metrics in top-K recommendation. From items other than those in the training set, the model is required to recommend 20 items to each user.

Table 1. Statistics of the datasets used in the experiments (# means the number, × means the dataset doesn't involve this part, the inter-density means the density of each user-item interaction matrix, the triplet-avg means the average triplets per item and the social-density means the density of each social matrix.)

Content		Last-FM	Yelp2018	Movie-Lens20M	Book-Crossing	Epinions	Flickr
Bipartite Graph	# users	1,892	3,884	5,734	5,931	5,183	5,791
	# items	3,846	4,553	1,695	7,483	5,589	8,212
	# interactions	21,173	13,059	28,194	17,603	9,457	32,432
	inter-density	0.291%	0.074%	0.290%	0.040%	0.033%	0.068%
Knowledge Graph	# entities	9,366	95,514	87,310	70,419	×	×
	# relations	60	42	32	25	×	×
	# triplets	15,518	1,071,279	50,188	133,892	×	×
	triplet-avg	4	235	30	18	×	×
Social Network	# links	12,532	3,418	×	×	44,613	116,570
	social-density	0.350%	0.023%	×	×	0.166%	0.348%

Baselines. Seven GNN-based RAs on the KG are chosen, including embedding-based approaches (CKE, MKR, KGAT) and propagation-based methods (RippleNet, KGCN, CKAN and KGIN), and Diffnet and Diffnet++ in the second category are selected. In addition, BPRMF is not dependent on either the KG or the social network. CKE [25] is an embedding-based approach in the co-learning category that integrates the loss functions in the KG embedding and recommendation task into a whole, while MKR [14] is a multi-task learning strategy with the cross&compress units. KGAT [16] proposes a novel GAT-based approach inspired by transR to maintain the adjacency matrix between entities. RippleNet [13] emphasizes description of user interests, on the contrary, KGCN [15] focuses on the exploration of items' semantics. KGIN [17] explains recommendation actions with intents. In contrast to the KG-based RA mentioned above, CKAN [19] also takes into account items' first-order CF signals. Diffnet [22] conveys high-order impacts between users on the social network to enhance user embeddings. Diffnet++ [21] additionally weights the contributions of the user's high-order CF signals and high-order social neighbors.

Parameter Settings. Our KIPRec model is implemented with the Pytorch deep learning framework. To be fair, the embedding size of all nodes and edges is 64, the mini-batch size is 128, and the maximum number of training epochs is 250 for all methods across all datasets. We initialize all parameters with the Xavier initializer and optimize them with the Adam optimizer. We tune the hyper-parameters within given ranges via a grid search. The training phrase is terminated if Recall@20 on the training set does not improve in the subsequent 10 epochs. Each result in this section is the average of five repeated experiments with different random seeds under the best hyper-parameter setup.

5.2 RQ1: Performance Evaluation

We compare KIPRec with its baselines with Recall@20 and NDCG@20 as shown in Table 2 and Table 3, and AUC in Fig. 4. The main experimental results are summarized as follows.

Table 2. Overall comparison of 9 KG-based methods in the top-K recommendation (The %Improv. symbol indicates the gap between the bolded best performance and the underlined suboptimal result from KIPRec's perspective.)

Model	Last-FM		Yelp2018		Movie-Lens20M		Book-Crossing	
	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG
BPRMF	0.2492	0.1446	0.0546	0.0266	0.4077	0.2086	0.0467	0.0285
CKE	0.2557	0.1444	0.0602	0.0296	0.4137	0.2108	0.0491	0.0323
MKR	0.2663	0.1469	0.0923	0.0461	0.4305	0.2139	0.0752	0.0534
RippleNet	0.2483	0.1387	0.0667	0.0282	0.4242	0.1981	0.0513	0.0366
KGCN	0.2864	0.1514	0.1370	0.0566	0.4965	0.2259	0.0853	0.0396
KGAT	0.2903	0.1702	0.1588	0.0681	0.4180	0.2098	0.0571	0.0324
CKAN	0.2449	0.1368	0.1486	0.0693	0.4670	0.2669	0.0779	0.0483
KGIN	0.3342	0.1922	0.2004	0.0875	**0.5720**	**0.3018**	**0.1476**	**0.0863**
KIPRec	**0.3406**	**0.1946**	**0.2117**	**0.0900**	0.5718	0.2877	0.1472	0.0846
%Improv.	1.92%	1.25%	5.64%	2.86%	-0.03%	-4.67%	-0.27%	-1.97%

Compared with the KG-based KGIN, Recall@20 and NDCG@20 evaluated on KIPRec are respectively improved by 1.92% and 1.25% on Last-FM, and by 5.64% and 2.86% on Yelp2018. In comparison with the social network-based Diffnet++, the two metrics on KIPRec increases by 10.05% and 12.81% on Last-FM, and by 11.54% and 7.91% on Yelp2018. Side information is reasonably utilized by KIPRec to generate more accurate user and item embeddings. In particular, the gap between KIPRec and Diffnet++ is more noticeable than that between KIPRec and KGIN, proving that the KG provides richer materials than the social network. It is feasible in KIPRec to obtain the preliminary embeddings from the KG. Moreover, BPRMF is the least effective, because it makes absolutely no use of any side information other than the bipartite graph.

On the other hand, on Movie-Lens20M and Book-Crossing including only the KG, KIPRec decreases by 0.03% and 0.27% respectively in terms of Recall@20, and by 4.67% and 1.97% relative to NDCG@20 compared with KGIN. This is because KIPRec incorporates the intention dimension associated with relationships on the KG. Yet, KIPRec achieves the similar goal, slightly less effective, by developing the multi-layer user interests structure. Figure 4 illustrates that CKAN or KIPRec is always the optimal or suboptimal model in the CTR prediction on Last-FM, Yelp2018 and Book-Crossing, since CF signals on the bipartite graph modify embeddings from the KG.

Table 3. Overall comparison of 4 social based methods in the top-K recommendation

Model	Last-FM		Yelp2018		Epinions		Flickr	
	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG
BPRMF	0.2492	0.1446	0.0546	0.0266	0.0614	0.0234	0.0109	0.0047
Diffnet	0.2730	0.1504	0.1770	0.0772	0.1310	0.0609	0.0261	0.0129
Diffnet++	<u>0.3095</u>	<u>0.1725</u>	<u>0.1898</u>	<u>0.0834</u>	<u>0.1448</u>	0.0651	<u>0.0306</u>	<u>0.0148</u>
KIPRec	**0.3406**	**0.1946**	**0.2117**	**0.0900**	**0.1471**	<u>0.0602</u>	**0.0351**	**0.0156**
%Improv.	10.05%	12.81%	11.54%	7.91%	1.59%	-7.53 %	17%	6.85%

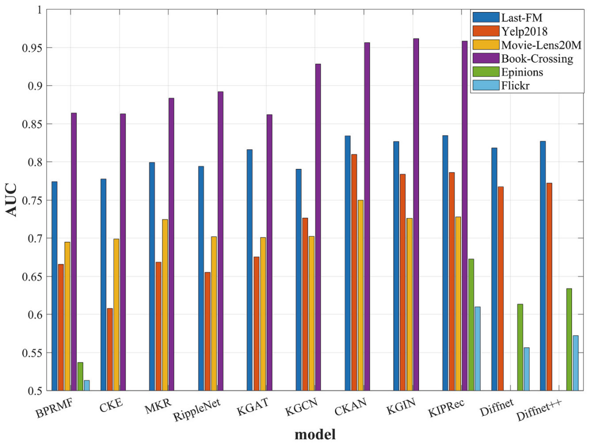


Fig. 4. The result of AUC in the CTR prediction scenario

Diffnet++ and KIPRec surpass Diffnet by integrating high-order CF signals and social influence. Recall@20 of KIPRec on Epinions and Flickr increases by 1.59% and 17%, and NDCG@20 is improved by -7.53% and 6.85% compared with Diffnet++. Inter-user and inter-item influences directly enable more accurate embeddings with less noise in KIPRec. In addition, KIPRec promotes embedding learning by broadening influence ranges through random sampling.

5.3 RQ2: Ablation Experiments

Table 4. Comparisons among variants of KIPRec

Category	Last-FM	Yelp2018	Movie-Lens20M	Book-Crossing	Setting
KIPRec$_{\text{w/o DI}}$	0.3189	0.2053	0.5607	0.1274	$e^s_{u_{p'}} = e^{shallow}_{u_{p'}}$
KIPRec$_{\text{w/o SI}}$	0.3197	0.2061	0.5683	0.1290	$e^s_{u_{p'}} = e^{deep}_{u_{p'}}$
KIPRec$_{\text{w/o Semantic}}$	0.3098	0.1938	-	-	Eq. (8):$e^{r(0)}_{i_q} = e^f_{i_q}$, Eq. (11):$e^{r(0)}_{u_p} = e^f_{u_p}$
KIPRec$_{\text{w/o Sample}}$	0.3203	0.1516	0.5432	0.1074	$L = 0, I^2_{con(p,q)} = \varnothing, U^2_{con(p,q)} = \varnothing$
KIPRec$_{\text{w/o Social}}$	0.3301	0.2040	**0.5718**	**0.1472**	Eq. (10):$\mathfrak{s}_{p\mathfrak{p}} = 0$
KIPRec$_{\text{w/o Propagation}}$	0.3018	0.1878	0.5325	0.1161	Sect. 4.3: $e^{r(K)}_{u_p} = e^{r(0)}_{u_p} = e^s_{u_p}$, $e^{r(K)}_{i_q} = e^{r(0)}_{i_q} = e^s_{i_q}$
KIPRec	**0.3406**	**0.2117**	**0.5718**	**0.1472**	-

We conduct two ablation experiments to estimate two layer's effects in KIPRec (KIPRec$_{\text{w/o Semantic}}$ and KIPRec$_{\text{w/o Propagation}}$). Their performances have significantly been decreased compared to KIPRec. As shown in Table 4, their performances indicates that both are indispensable. Further, both KIPRec$_{\text{w/o DI}}$ and KIPRec$_{\text{w/o SI}}$ outperform the KIPRec$_{\text{w/o Semantic}}$ on Last-FM and Yelp2018, but they are less effective than KIPRec, which reveals that capturing multi-layer user interests is essential. The effectiveness improvement in KIPRec$_{\text{w/o Sample}}$ or KIPRec$_{\text{w/o Social}}$ is significant over KIPRec$_{\text{w/o Propagation}}$, since RAs cannot adequately capture user interests from semantics without rich interaction history.

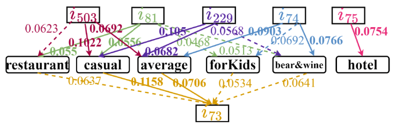
In addition, this paper explores the impact of different embedding dimensions on information capture ability through the experiment results in Table 5. It can be seen that KIPRec's recommendation effect performs best when $d = 64$ on three datasets. The optimal performances on the other three datasets are not particularly better than that of 64 dimensional setting. It is worth mentioning that larger dimensions on the large-scale dataset (Yelp2018) lead to GPU's usage exceeding the limit.

Table 5. Embedding dimension (d) settings' influence in KIPRec

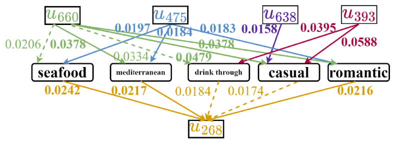
Dataset	16	32	64	128	256	512
Last-FM	0.3287	0.3324	0.3406	0.3418	0.3436	**0.3439**
Yelp2018	0.1428	0.2094	0.2117	**0.2151**	-	-
Movie-Lens20M	0.5585	0.5627	**0.5718**	0.5586	0.5677	0.5713
Book-Crossing	0.0954	0.1130	**0.1472**	0.1425	0.1346	0.1447
Epinions	0.1471	0.1442	**0.1471**	0.1471	0.1399	0.1471
Flickr	0.0325	0.0334	0.0351	0.0345	**0.0356**	0.0339

5.4 RQ3: Case Study

To illustrate the effect of KIPRec's semantic propagation layer at the attribute level, we select users and items associated with u_{268} and i_{73}, as shown in Fig. 5.



(a) i_{73} and related items



(b) u_{268} and related users

Fig. 5. A real-word example from Yelp2018 related to the semantic propagation layer (The solid line connects a user or item with the attributes that it values the most. Number on the line indicates how much the importance the attribute is placed.)

From the perspectives of whether users or items attach the same level of importance to each attribute and whether their most valued attributes are the same, we find that the semantic similarities to i_{73} and u_{475} in descending order are $i_{229} > i_{503} > i_{81} > i_{74} > i_{75}$ and $u_{475} > u_{660} > u_{393} > u_{638}$, which is consistent with the statistics in Table 6 and Table 7.

Table 6. Relations between i_{73} and its related nodes in influence propagation layer

Neighbors	i_{503}	i_{81}	i_{229}	i_{74}	i_{75}
Semantic similarity	0.9082	0.8393	0.9755	0.6429	0.5286
Co-interactions (training)	0	$1\{u_{660}\}$	$1\{u_{660}\}$	$3\{u_{660}, u_{268}, u_{638}\}$	$2\{u_{268}, u_{638}\}$
Overall similarity	0.0001	0.0018	0.1359	0.0493	0.0252

KIPRec aggregates messages from neighbors based on the overall similarity among users or items. The overall similarity is mainly determined by the semantic similarity, followed by co-interactions and social relationships. Thus, compared to i_{74}, i_{229} has a higher semantic similarity to i_{73} and thus a higher overall similarity; compared to i_{75}, i_{81} has a lower number of interactions in common with i_{73}, and thus a lower overall similarity.

Table 7. Relations between u_{268} and its related nodes in influence propagation layer

Neighbors	u_{475}	u_{393}	u_{638}	u_{660}
Semantic similarity	0.7986	0.5822	0.5690	0.7732
Co-interactions (training)	0	0	$3\{i_{73}, i_{74}, i_{75}\}$	$2\{i_{73}, i_{74}\}$
Social relationships	0	1	0	1
Overall similarity	3.20×10^{-35}	5.41×10^{-12}	1.74×10^{-32}	1.14×10^{-6}
Co-interactions (testing)	$1\{i_{484}\}$	$3\{i_{225}, i_{265}, i_{652}\}$	$1\{i_{235}\}$	$1\{i_{229}\}$

Because social influence plays a greater role in user representations, users that have social relationships with u_{268} are more similar in general to those have no social relationship. Furthermore, i_{73} and u_{268} respectively absorb user information from i_{503} and u_{475}, which are only semantically similar. The number of users' co-interactions in the testing set is essentially consistent with their overall similarity. The exception to this is that u_{660} and u_{268} have the highest overall similarity yet with one common interaction. This is because there are a few interaction records in the dataset for u_{660}.

6 Conclusion

This paper has proposed a RA based on GATs that naturally combines the knowledge graph, the bipartite graph and the social network to enrich user and item embeddings. A multi-layer user interest structure is designed at the attribute and item levels to achieve user embeddings with rich semantics. Besides the similarity from the semantic propagation layer, the inter-user and inter-item influence weights in the influence propagation layer are further acquired based on the bipartite graph and the social network. Multiple graph attention message propagation are performed in the two layers to obtain more precise user and item embeddings. Extensive experiments and comparative analysis on two datasets demonstrate the advantage of the proposed model over the SOTA baselines.

The future work is manifold. It would be interesting to study how to reduce GAT operations' time and space costs in the semantic propagation layer on large-scale graphs using sample strategies [11,23]. It is also worth investigating self-supervised methods to generate more accurate recommendations by identifying relationships among the refined item embeddings from the user-item bipartite graph, the item-item co-interaction graph, and the KG.

Acknowledgments. We are grateful to the reviewers of this paper for their constructive and insightful comments. The research reported in this paper was partially supported by the National Key R&D Program of China (NO. 2021YFB0300104), and the ANR project AGAPE ANR-18-CE23-0013.

References

1. Chen, T., Guo, J., et al.: Graph representation learning for popularity prediction problem: a survey. Discrete Math., Algorithms Appl. **14**(7), 2230003 (2022)
2. Fan, W., Ma, Y., et al.: A graph neural network framework for social recommendations. TKDE **34**(5), 2033–2047 (2020)
3. Gao, C., Wang, X., et al.: Graph neural networks for recommender system. In: WSDM, pp. 1623–1625 (2022)
4. Guo, Q., Zhuang, F., et al.: A survey on knowledge graph-based recommender systems. TKDE **34**(8), 3549–3568 (2020)
5. He, X., Deng, K., et al.: LightGCN: simplifying and powering graph convolution network for recommendation. In: SIGIR, pp. 639–648 (2020)
6. Huang, C.: Recent advances in heterogeneous relation learning for recommendation. In: IJCAI, pp. 4442–4449 (2021)
7. Jiang, W., Sun, Y.: Social-RippleNet: jointly modeling of ripple net and social information for recommendation. Appl. Intell. **53**(3), 3472–3487 (2022). https://doi.org/10.1007/s10489-022-03620-2
8. Li, A., Yang, B.: GSIRec: Learning with graph side information for recommendation. World Wide Web **24**(5), 1411–1437 (2021). https://doi.org/10.1007/s11280-021-00910-6
9. Li, N., Gao, C., et al.: Disentangled modeling of social homophily and influence for social recommendation. TKDE **35**(6), 5738–5751 (2022)
10. Liu, T., Wang, Z., et al.: Recommender systems with heterogeneous side information. In: WWW, pp. 3027–3033 (2019)
11. Liu, X., Yan, M., et al.: Sampling methods for efficient training of graph convolutional networks: a survey. IEEE/CAA J. Automatica Sinica **9**(2), 205–234 (2022)
12. Rendle, S., Freudenthaler, C., et al.: BPR: bayesian personalized ranking from implicit feedback. In: UAI, pp. 452–461 (2009)

13. Wang, H., Zhang, F., et al.: RippleNet: propagating user preferences on the knowledge graph for recommender systems. In: CIKM, pp. 417–426 (2018)
14. Wang, H., Zhang, F., et al.: Multi-task feature learning for knowledge graph enhanced recommendation. In: WWW, pp. 2000–2010 (2019)
15. Wang, H., Zhao, M., et al.: Knowledge graph convolutional networks for recommender systems. In: WWW, pp. 3307–3313 (2019)
16. Wang, X., He, X., et al.: KGAT: knowledge graph attention network for recommendation. In: SIGKDD, pp. 950–958 (2019)
17. Wang, X., Huang, T., et al.: Learning intents behind interactions with knowledge graph for recommendation. In: WWW, pp. 878–887 (2021)
18. Wang, X., Wang, D., et al.: Explainable reasoning over knowledge graphs for recommendation. In: AAAI, pp. 5329–5336 (2019)
19. Wang, Z., Lin, G., et al.: CKAN: collaborative knowledge-aware attentive network for recommender systems. In: SIGIR, pp. 219–228 (2020)
20. Wang, Z., Wang, Z., et al.: Exploring multi-dimension user-item interactions with attentional knowledge graph neural networks for recommendation. IEEE Trans. Big Data, TBD **9**(1), 212–226 (2023)
21. Wu, L., Li, J., et al.: DiffNet++: a neural influence and interest diffusion network for social recommendation. TKDE **34**(10), 4753–4766 (2020)
22. Wu, L., Sun, P., et al.: A neural influence diffusion model for social recommendation. In: SIGIR, pp. 235–244 (2019)
23. Xu, X., Feng, W., et al.: Sampling methods for efficient training of graph convolutional networks: a survey. In: ICLR (2020)
24. Yu, J., Yin, H., et al.: Socially-aware self-supervised tri-training for recommendation. In: SIGKDD, pp. 2084–2092 (2021)
25. Zhang, F., Yuan, N.J., et al.: Collaborative knowledge base embedding for recommender systems. In: SIGKDD, pp. 353–362 (2016)
26. Zhang, S., Yao, L., et al.: Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. **52**(1), 1–38 (2019)
27. Zhu, F., Wang, Y., et al.: Cross-domain recommendation: challenges, progress, and prospects. In: IJCAI, pp. 4721–4728 (2021)