Knowledge and Information Systems (2023) 65:1685–1712
https://doi.org/10.1007/s10115-022-01808-z

REGULAR PAPER

An efficient joint framework for interacting knowledge graph and item recommendation

Haizhou Du[1] · Yue Tang[1] · Zebang Cheng[1]

Received: 16 May 2022 / Revised: 27 November 2022 / Accepted: 12 December 2022 /
Published online: 27 December 2022

Abstract
Incorporating knowledge graphs in recommendation systems is promising as knowledge graphs can be a side information for recommendation systems to alleviate the sparsity and the cold start problems. However, existing works essentially assume that side information (i.e., knowledge graphs) is completed, which may lead to sub-optimal performance. Meanwhile, semantic hierarchies implied in applications are prevalent, and many existing approaches fail to model this semantic characteristic. Modeling the semantic structure between items in recommendation systems is a crucial challenge. Therefore, it is crucial to solve the incompleteness of knowledge graphs when integrating it into recommendation system as well as to represent the hierarchical structure contained in items. In this paper, we propose Paguridae, a framework that utilizes the item recommendation task to assist link prediction task. A core idea of the Paguridae is that two tasks automatically share the potential features between items and entities. We adopt two main structures to model the hierarchy between items and entities. In order to model the hierarchy in items, we adopt graph convolutional networks as a representation learning method. In order to model the hierarchy in entities, we use Hirec model, which maps entities into the polar coordinate system. Under the framework, users can get better recommendations and knowledge graphs can be completed as these two tasks have a mutual effect. Experiments on two real-world datasets show that the Paguridae can be trained substantially, improving F1-score by 62.51% and precision by 49.31% compared to the state-of-the-art methods.

Keywords Recommendation systems · Knowledge graph embedding · Hierarchical structure · Graph convolutional network · Multi-task learning

1 Introduction

The recommendation system (RS) is designed to recommend personalized online items or information for users. It is widely used in many Web scenarios to deal with the problem of

✉ Haizhou Du
duhaizhou@shiep.edu.cn

1 School of Computer Science and Technology, Shanghai University of Electric Power, Shanghai 200090, China

information overload caused by massive information data to improve the quality of a user's life. With the rapid development of big data and deep learning, knowledge graphs (KGs) have attracted significant attention in many fields such as recommendation systems, dialog systems and search engines.

Early recommendation systems mainly adopt collaborative filtering (CF) to recommend items. Despite its success in recommendation systems, CF often suffers from data sparsity and cold start problems in real-world recommendation scenarios. To alleviate these problems, researchers suggest integrating side information into CF, such as introducing social networks [1], relational data [2], context-related comments [3] and knowledge graphs [4–8]. The knowledge graph has more static information than the user-item, so it can better solve the cold start problem. For example, it is effective in real-world movie recommendation when there is a new movie or a user who had just signed up for, because the recommendation system can recommend useful movies or other information based on the people who are closely related to the new user in the knowledge graph. In a nutshell, KG can enhance the performance of the recommendation system from three aspects: (1) From the perspective of nodes, KG introduces semantic associations between items, which can help to discover potential associations between items and improve the accuracy of recommended items. (2) From the side point of view, KG is composed of various types of relationships, which is conducive to rationally expanding users' interests, and increasing the diversity of recommended items. (3) KG connects the user's history with the recommendation path, which brings the interpretability for the recommendation systems. Taken these advantages of knowledge graph, the recommendation system can solve the problems that cannot be solved by CF method before, i.e., cold start and sparsity problems.

However, there are also drawbacks to KG. For example, existing work assumes that KGs which act as side information roles in RS are complete and there are no missing nodes or edges, limiting the benefits of knowledge transfer. Apart from this, it is common for KG to model composition, inversion, symmetric/antisymmetric, and other relational patterns. However, semantic hierarchies implied in real-world applications are more and more prevalent, and many existing approaches fail to model this hierarchical information so that this information is not used effectively. This missing information can have a significant impact on the recommendation system. From a micro point of view, previous researchers devoted themselves to exploring the transformation of relationships, such as symmetry, but they ignored the semantic information inside the data, such as hierarchy. This paper explores the impact of semantic information on the performance of link prediction from this aspect. As shown in Fig. 1, such a hierarchical graph essentially reveals the rich relations behind items. For example, two items, actors and directors, from different but closely related categories are very likely to have complementary relations concerning their functions. Such the hierarchical relations between items can significantly improve the performance of recommendations. As such, it is crucial to think about the incompleteness of knowledge graphs when adopting it into recommendation system as well as to represent hierarchical structure contained in items.

In this paper, we propose Paguridae, a framework that utilizes item recommendation task to assist link prediction task. The relationship between these two tasks is like that between hermit crabs and shrimp crabs in the ocean. In order to capture the semantic information(i.e., hierarchy structure) from item to item and entity to entity, Paguridae first captures the topological structure in RS based on a graph convolutional network (GCN) model and a hierarchy structure in recommendation systems (HIREC) model, which is an extension of HAKE [9] in the embedding layer, and then unifies the item recommendation task and link prediction task in a joint model.

The main contributions of our work can be summarized as follows:

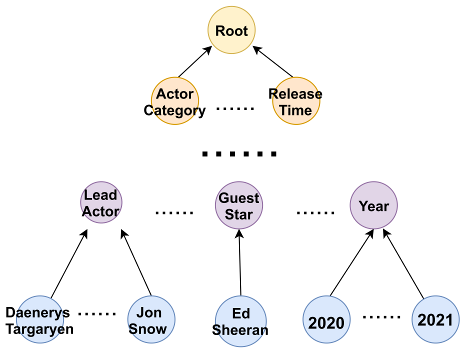


Fig. 1 The hierarchical structure in recommendation systems. The items are first categorized into different categories (e.g., the actor category and the release time of the movie), and then each category is divided into several sub-categories (e.g., Lead actor in the actor category and year in release time), while each sub-category includes multiple items (e.g., Daenerys Targaryen, Jon Snow under the Lead actor). From the hierarchical structure, we can know that Daenerys Targaryen may have something to do with Jon Snow as they have the same father node, Lead Actor

- From a data-driven perspective, we highlight the importance of hierarchical structure on items in RS, as well as entities in KG, which can better analyze the data complexities and characteristics.
- We propose Paguridae by combining the GCN. The GCN is used to capture the topological(hierarchical) structure of the items where there are no item-entity interactions.
- The Paguridae integrates the KG and adopts the framework of joint learning, which not only solves the hierarchical structure of entities in KG but also makes a better recommendation.
- We conduct plentiful experiments on two public datasets, and the results demonstrate the effectiveness of Paguridae.

For the rest of this paper, we first review some related work in Sect. 2. Then, we introduce the design and analysis of Paguridae in Sect. 3. Next, we describe the offline experimental settings and results for verifying the performance of Paguridae in terms of link prediction and item recommendation in Sect. 4. Finally, we make a conclusion for our work and discuss the limitations of the work in Sect. 5.

2 Related work

Our model Paguridae contains two tasks, one is link prediction in KG, and the other is item recommendation in RS. For each task, we will introduce their related work firstly, and then show the relation between these two tasks.

2.1 Link prediction in KG

Three kinds of methods are adopted to the link prediction task, they are based on translation model, matrix decomposition model and deep learning model [10], respectively.

2.1.1 Translation model methods

Translation-based model regards relationships as a transition between head-to-tail entities, and they are good at modeling patterns such as inverted relationships. TransE [11] first proposed the core idea of Translation-based model. It performs well in modeling 1–1 relations. However, some relations can translate one entity to various entities. For example, it may be confused when it comes to more complex relations such as 1–N, N–1 or N–N. To this end, TransH [12] learns different representations for an entity conditioned on different relations. TransR [13] models entities and relations in entity space and relation space, respectively. TransMS [14] regards the head entity, the relative entity and the tail entity as the subject, predicate and object in the sentence, respectively. In this way, it considers the semantics between the header entities and the relationship, as well as the semantics between the relationship and the tail entities, which was not done in previous models. TransL [15] introduced local joins into the translation embedding model. The model uses a common approach to combine all entities in local joins, using different weights to distinguish the contribution degree of different relationships.

2.1.2 Matrix decomposition model methods

Matrix decomposition model usually uses the dot product of embedded vectors as a scoring function. Symmetry and (non-)reflexivity of relations can be handled, and transitivity can also be achieved with some improvements. For example, RESCAL [16] can capture the compositional relationships between entities, for it models each relation as a matrix. In order to learn the relation matrix simplify, there are some variations, for instance, DistMult [17] make a restriction for that relation matrices are diagonal, HolE [18] defines a circular correlation [19] that compresses the relational matrix into vectors. In order to model the asymmetric relation, ComplEx [20] introduces a complex value. HypER [21] uses the hyper network to generate a one-dimensional convolution filter for each relation, extracting relation-specific features from the principal entity embedding, which can be understood to be nonlinear through tensor decomposition, thus making it closer to the established decomposition model. The disadvantage of this model is that it sets most elements of the core weight tensor to 0, which amounts to demanding regularization rather than having the model learn which parameters to use through soft regularization. TuckER [22] decomposition of binary tensors based on known facts can perform multi-task learning across relations. While fully expressing, the number of parameters of this model increases linearly with the number of entities or relations in the knowledge graph. MEI [23] divides each embedded vector into a multi-partition vector, thus effectively limiting the interaction. The Tucker tensor form of each local interaction modeling and the complete interaction modeling and shielding tensor form allow MEI to control the trade-off between expression and computational costs, automatically learn the interaction mechanism from the data, and ultimately achieve the performance of advanced link prediction tasks.

2.1.3 Deep learning model methods

Deep learning models have been widely used in tasks such as prediction and classification due to their powerful representational abilities. They can combine with different input data to recognize significant patterns to learn parameters such as weights and biases. Standard methods of deep learning include ConvE [24], ConvKB [25] , and RSN [26], in which

ConvE, ConvKB are based on convolutional neural networks [27], while RSN is based on recurrent neural networks [28]. [29] proposed a heterogeneous neural network framework based on attention mechanism to process complex graph data and simultaneously aggregate multiple types of semantic information. With the development of graph convolutional neural network(GCN), deep learning methods based on GCN include CompGCN [30], TransGCN [31], and Ra-GCN [32].

However, these embedding models consider each KG triple individually and fail to capture the useful information present in the neighborhood of a node. KGEL [33] consists of an encoder of a dual weighted graph convolutional network and a decoder of a novel fully expressive tensor factorization model. These methods have witnessed great advancement of representation learning (RL)-based models for the knowledge graph relation prediction task. However, they generally rely on structure information embedded in the encyclopedic knowledge graph, while the beneficial semantic information provided by lexical knowledge graph is ignored, leading the problem of shallow understanding and coarse-grained analysis for knowledge acquisition. Therefore, HARP [34]introduces concept information derived from the lexical knowledge graph (e.g., Probase) and proposes a novel hierarchical attention model for relation prediction, which consists of entity-level attention mechanism and concept-level attention mechanism, to thoroughly integrate multiple semantic signals.

To sum up, these methods attempt to model the relation vectors in the link prediction problem from various perspectives, including symmetry, transitivity and rotation. However, semantic hierarchies implied in real-world applications are more and more prevalent, and these approaches fail to model this hierarchical information so that this information is not used effectively. Our Paguridae considered this drawback.

2.2 Item recommendation in RS

2.2.1 CF-based methods

In earlier studies, researchers focused on recommending information about similar users or items for a specific user, while this information is only detected from history interactions, which may cause cold start problem, also, they suffer from the data sparsity issue. Such algorithms include collaborative filtering algorithms [35], which are collectively referred to as similarity-based methods.

2.2.2 Content-based methods

Content-based methods take various side information such as knowledge graphs [4], the relational data [2] and the contextual reviews [36] to improve the performance of item recommendation. Knowledge not only can enrich items or users information, but also can be a good explanation of why the user chose this item, that is, to provide good interpretability. Piao and Breslin [37] extracted features from KG for factorization machines. Zhang et al. [5] view the recommendation task as the link prediction task in the knowledge graph. The purchasing relationship between users and items is modeled, and then use TransE [11] method for prediction. KGAT [6] firstly proposed a CKG structure, which is a united graph constructed by users, items and entities. KGAT models the high-order relations between users and items through embedding propagation. CoFM [7] makes an alignment of items and entities and then jointly trains TransE method and FM method by their sharing parameters or regularization. CKE [8] is a collaborative model which adopts link prediction method TransR [13] to strengthen the effectiveness of item recommendation. [38] proposed a fairness constrained approach via heuristic re-ranking to mitigate unfairness problem in the context of explainable recommendation over knowledge graphs. [39] proposed a Path Language Modeling Recommendation (PLM-Rec) framework. It alleviated the aforementioned recall bias a language model over KG paths consisting of entities and edges, which can not only capture the user behaviors but also eliminate the restriction to pre-existing KG connections. [40] adds the thought of attention mechanism and defines that users themselves do not have embedding, which depends entirely on knowledge graph embedding. KG has been applied in recommendation systems, exploiting graph neural networks (GNNs), but most existing recommendation models based on GNNs ignore the influence of node types and the loss of information during aggregation. ICKG [41] relieves the sparsity of the network and overcomes the limitation of information loss of the traditional GNN recommendation model. [42] proposed a new deep reinforcement learning-based music recommendation method with knowledge graphs, and it can make appropriate recommendations even with a small amount of user preference information by learning the optimal action of the agent. To sum up, these methods adopt KG as a side information to better recommend for a user. However, they do not consider the incompleteness of KG, which may bring a sub-optimal result. Our Paguridae solved this problem.

2.3 Relationship between two tasks—multi-task learning

Since the recommendation system also looks for top k items in multiple items, which is equivalent to looking for top k entities from multiple entities in the link prediction task, so we can regard these two tasks as the same purpose.

KTUP adopts the framework of multi-tasking learning, which can carry out recommendation task and knowledge graph completion task simultaneously [43]. The motivation of multi-task learning is that items in the RS can correspond to the entities in the KG, that is to say, the item embedding can be enriched by transferring knowledge of entities embedding in each module. MKR [44] uses a cross compress unit to be a connection, so that to transfer knowledge and share regularization between recommendation module and a KGE module. RCF [45] gives a hierarchical description of items, which contains two parts, one is the relation type embedding, and the other is relation value embedding.

Our work can also be seen as a multi-task learning framework, the difference is that Paguridae combined with hierarchical embedding, as well as considering the incompleteness of KG. Based on these, Paguridae jointly trains RS module and the KG module in an end-to-end manner.

3 Approach

We first give the problem statement in Sect. 3.1 and then show an overall architecture for our solution in Sect. 3.2. Finally, we introduce the details of the framework in the rest subsections.

3.1 Problem statement

The input in our Paguridae contains KG, user-item interactions and a set of item-entity alignments $\mathcal{A} = \{(i, e) \mid i \in \mathcal{I}, e \in \mathcal{E}\}$.

Table 1 Symbols notation

Symbols	Descriptions
h	The embedding vector of head entity
r	The embedding vector of relation
t	The embedding vector of tail entity
$\boldsymbol{i\_h}_m$	The embedding representation of head in recommendation systems, m denotes 'modulus'
$\boldsymbol{r}_m$	The embedding representation of relation in recommendation systems
$\boldsymbol{i\_t}_m$	The embedding representation of tail in recommendation systems
$\boldsymbol{i\_h}_p$	The embedding representation of head in recommendation systems, p denotes 'phase'
$\boldsymbol{r}_p$	The embedding representation of relation in recommendation systems
$\boldsymbol{i\_t}_p$	The embedding representation of tail in recommendation systems
$\boldsymbol{d}_{r,m}$	The distance function for the modulus part
$\boldsymbol{d}_{r,p}$	The distance function for the phase part

Paguridae can output not only item recommendation results but also link prediction results based on the jointly learned embeddings of users, items, entities and relations. Now let us talk about these two tasks, respectively.

3.1.1 Link prediction task in knowledge graphs

A KG can be defined as a directed, labeled multi-graph, which can be formulated as KG = $\langle \mathcal{E}, \mathcal{R}, \mathcal{G} \rangle$, where $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{G}$ represent entities, relations, and edges, respectively. Given a KG, data representation exists as triples, i.e.,$\langle h, r, t \rangle$, where h, r, and t stand for head, relation, and tail, respectively. The link prediction(LP) task is to predict the missing h or t or r for some triples.

3.1.2 Item recommendation task in recommendation systems

How to recommend the products that users want to interact with to specific users in the massive data is the main task of the recommendation system. The input and output in recommendation systems are as follows:

- Input: Knowledge Graphs and User-Item Interaction Matrix.
- Output: The probability $\hat{y}_{i,j}$ that one user would adopt an item.

Based on the above two prediction tasks, we can integrate them into one model through the Paguridae, so as to complete the link prediction task as well as item recommendation task simultaneously.

The mathematical notations used in this paper are summarized in Table 1.

3.2 The overview of Paguridae framework

We designed a efficient joint framework, called Paguridae, as shown in Fig. 2, which consists of four main components, i.e., input information layer, embedding layer, jointly learning layer and prediction layer.

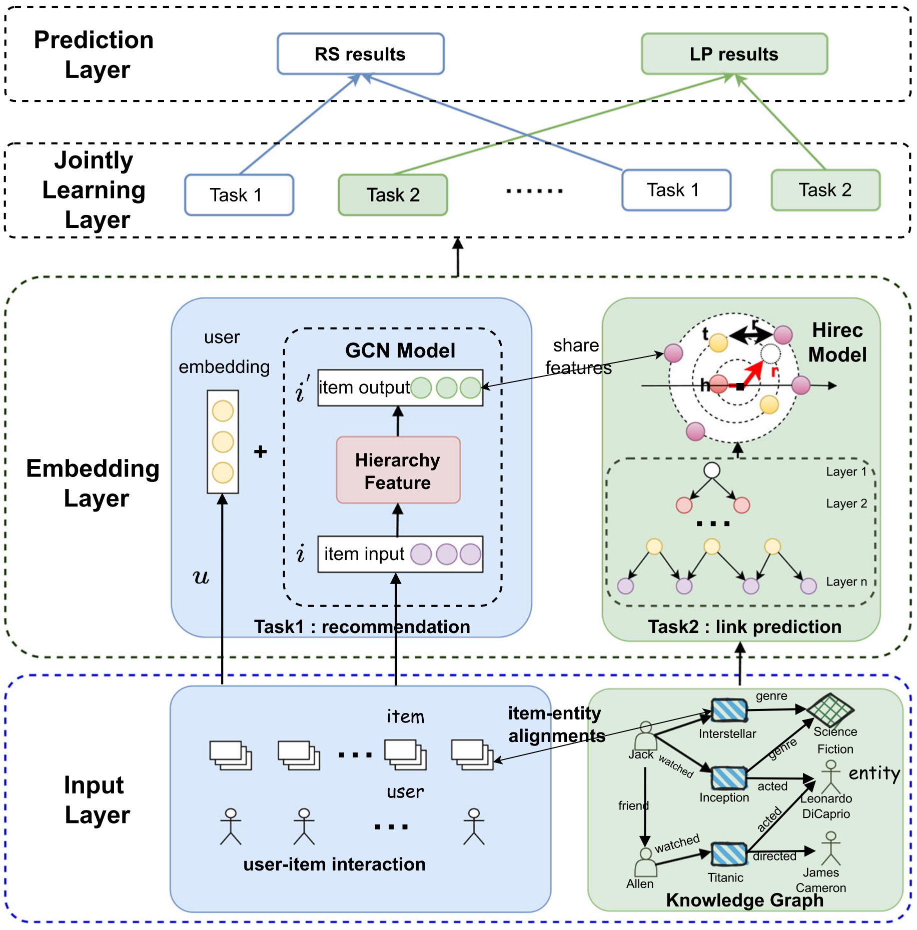


Fig. 2 The architecture of the joint model

The input information layer contains three parts, i.e., Knowledge Graphs, User-Item Interaction Matrix and a set of item-entity alignments. After this part, embedding users, items, and entities into the appropriate vector space is the first problem, so the second layer is the embedding layer. Considering the hierarchical structure of the items in recommendation systems and entities in knowledge graph, how to get the hierarchical structure is another challenge. In our Paguridae, we adopt two main structures to model the hierarchy between items and entities. In order to model the hierarchy in items, we adopt graph convolutional networks as a representation learning method. In order to model the hierarchy in entities, we use Hirec(Hierarchy structure in Recommendation Systems), which is the extension of HAKE [9] model, to capture the hierarchy between entities further. The third level, the jointly learning layer, is how entities in link prediction interact with items in the recommendation system to achieve knowledge enhancement. Finally, we use a prediction layer to get the prediction results. Now, let us specifically introduce these four layers.

Table 2 The mapping between items in RS and entities in KG

Items in RS	Entities in KG
Stealing Beauty (1996)	http://dbpedia.org/resource/Stealing_Beauty
Suicide Kings (1997)	http://dbpedia.org/resource/Suicide_Kings
Romeo Is Bleeding (1993)	http://dbpedia.org/resource/Romeo_Is_Bleeding
Screwed (2000)	http://dbpedia.org/resource/Screwed_(2000_film)
Magnum Force (1973)	http://dbpedia.org/resource/Magnum_Force

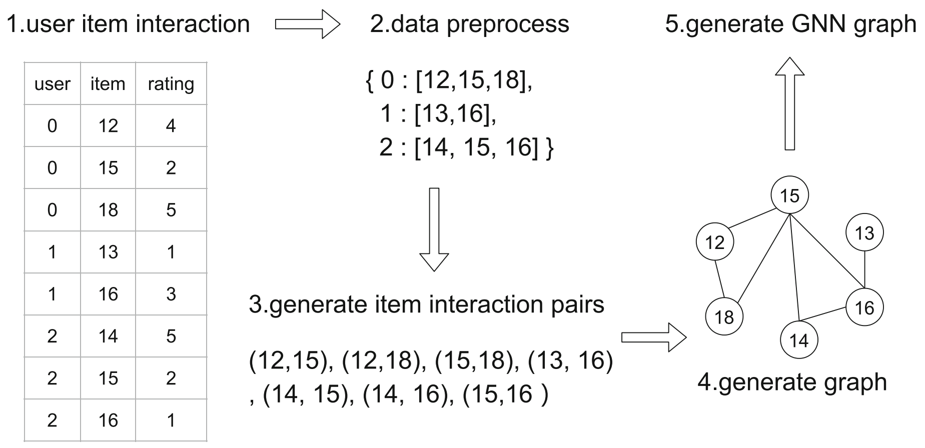


Fig. 3 A workflow example of employing GNN on recommendation systems

3.2.1 Input information layer

The input information layer contains three parts, i.e., Knowledge Graphs, User-Item Interaction Matrix and a set of item-entity alignments. For item-entity alignments, they are refined for LODRecSys [46, 47] by mapping items into DBPedia entities if there is an available mapping. As Table 2 shows, so that Stealing Beauty (1996) i.e., item can better utilize the information brought by http://dbpedia.org/resource/Stealing_Beauty, i.e., entity.

3.2.2 Embedding layer

Traditional embedding for item recommendation tasks is mainly devoted to modeling the relationship between users and items. However, semantic hierarchies implied in real-world applications are more and more prevalent, and many existing approaches fail to model this hierarchical information so that this information is not used effectively.

Inspired by word embedding [48], graphs can also be embedding well, which we call graph embedding [49]. In KG, knowledge graph embedding is good at preserving the knowledge graph structures while parameterizing entities and relations as vector representations. Based on this idea, in order to get the hierarchical structure of items in recommendation systems, we take all items in RS as a graph, a workflow example of how to generate items in RS as a graph is as Fig. 3 shows, then we use GCN on this item graph, so that we can preserve the knowledge graph structures as well as parameterizing items as vector representations i.e.,

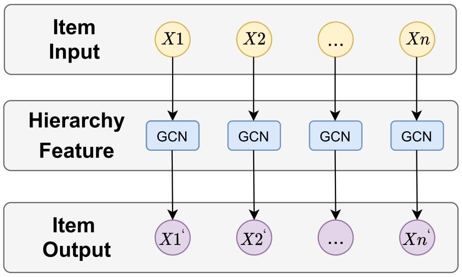
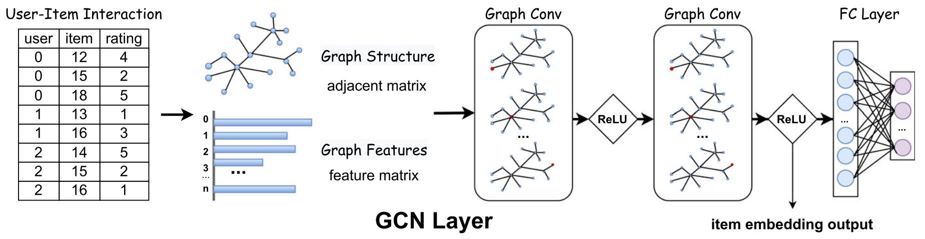


Fig. 4 The items embedding output by the GCN layer



user	item	rating
0	12	4
0	15	2
0	18	5
1	13	1
1	16	3
2	14	5
2	15	2
2	16	1

Fig. 5 GCN layer

hierarchical structure. Inspired by [50], our system can be applied to more functions such as node classification.

As shown in Fig. 4, given item graph G and feature matrix X, the problem of hierarchical structure item recommendation can be regarded as learning a mapping function f, as shown in Eq. (1).

$$[X_1^{'}, X_2^{'}, \dots, X_n^{'}] = f(G; (X_1, X_2, \dots, X_n)), \tag{1}$$

where n is the number of item. f is GCN function. Now, let us talk about GCN model.

- **GCN model**
  Input: User-Item Interaction Matrix.
  As shown in Fig. 5, user '0' has interaction with item '12', '15' and '18', so we can get an item graph on these three items. By analogy, we can get an overall item graph based on each user, as Fig. 3 shows.
  Output: Item Embedding.
  After two graph conv steps, we can output the item embedding processed by GCN layer. Notice that we do not get output after FC layer, structurally, the middle layer is used for capturing features, while the node features of the last layer are used for categorizing tasks.

Hierarchy feature modeling

We choose the GCN model to effectively learn spatial features from user-item interactions. As shown in Eq. (2), a GCN model which contains two layers can be expressed as follows [51]:

$$f(X, A) = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}Relu(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}XW_0)W_1), \tag{2}$$

where X stands for feature matrix, A represents adjacency matrix, $\tilde{A} = A + I$ and $\tilde{D}$ is a degree matrix. We use Eq. (3) to calculate $\tilde{D}$:

$$\tilde{D} = \sum \tilde{A}_{ij}. \tag{3}$$

Table 3 The input elements in HIREC model

Tag	Resource
u	User item interaction
i'	Processed by GCN layer
h or t	Knowledge graph

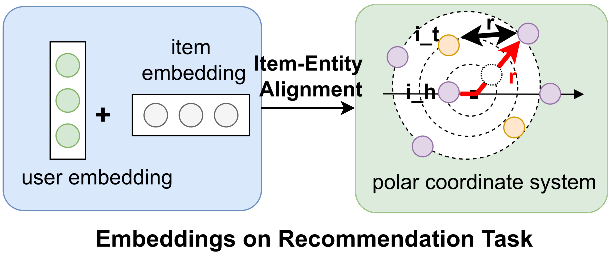


Fig. 6 Embeddings transfer from RS to KG

Each hidden layer in GCN corresponds to a feature matrix, and each row in the matrix is a characteristic representation of a node. GCN aggregates this information at each level using propagation rule f to form the characteristics of the next level. As a result, the features become more and more abstract in each successive layer.

In order to obtain the item graph, we aggregate the interactive items of the same user according to the user-item interaction data, thus forming an item graph, as Fig. 3 shows. Then, a graph convolution network is used to capture the topological(hierarchical) structure of item network to obtain the spatial feature.

When it comes to the training process, we aim to minimize the error between the real item that user interacts with and the predicted one. The corresponding loss function is

$$Loss_{gcn} = \| Y - \hat{Y} \| + \lambda_{gcn} L_2, \tag{4}$$

where Y is the real item which user interacts with, and $\hat{Y}$ is the predicted one. To avoid the risk of overfitting, we added L_2 as L2 regularization, where λ_{gcn} is a hyperparameter.

- **HIREC Model**
  Input: As shown in Table 3, HIREC model takes three input, they are users who come from user-item interaction, items processed by GCN layer and entities come from knowledge graph, respectively.

- **Output**: Item and entity Embedding which contains hierarchical structure.
  Hierarchy Feature Modeling
  Here, we use HiRec(Hierarchy structure in Recommendation Systems), which is the extension of HAKE [9] as the embedding model for recommendation task, as shown in Fig. 6. To be more specific, we choose a polar coordinate system as a representational space as a polar coordinate system is good at describing the two relationships we want to model. To the best of our knowledge, we are the first to employ the hierarchy structure from RS to KG.

There are two key parts in our HiRec, the modulus part and the phase part, respectively. Different part models items in different categories.

The modulus part

In this part, our goal is to model the items at different levels of the hierarchy, for example, parent-child nodes in a tree structure.

Inspired the idea of RotatE [52], which regards the relation $\boldsymbol{r}$ as a transformation from $\boldsymbol{h}$ to $\boldsymbol{t}$, so that we can formulate the modulus part as Eq. (5) shows

$$\boldsymbol{i\_h}_m \circ \boldsymbol{r}_m = \boldsymbol{i\_t}_m, \tag{5}$$

where $\boldsymbol{i\_h}_m$, $\boldsymbol{r}_m$ and $\boldsymbol{i\_t}_m \in \mathbb{R}^k$, according by the setting of HAKE [9], we also make a assumption that $\boldsymbol{r}_m \in \mathbb{R}^k_+$.

The distance function for the modulus part is

$$d_{r,m}(\boldsymbol{i\_h}_m, \boldsymbol{i\_t}_m) = \| \boldsymbol{i\_h}_m \circ \boldsymbol{r}_m - \boldsymbol{i\_t}_m \|_2, \tag{6}$$

where $\boldsymbol{i\_h}$, $\boldsymbol{r}$ and $\boldsymbol{i\_t}$ denote the embedding representation of head, relation and tail in recommendation systems, and the index m represents modulus part.

The phase part

In this part, our goal is to model the items at the same level of the semantic hierarchy. For example, sibling nodes in a tree.

The phase part can be formulated as follows:

$$(\boldsymbol{i\_h}_p + \boldsymbol{r}_p) \mod 2\pi = \boldsymbol{i\_t}_p, \tag{7}$$

where $\boldsymbol{i\_h}_p, \boldsymbol{r}_p, \boldsymbol{i\_t}_p \in [0, 2\pi)^k$.

The distance function for the phase part is

$$d_{r,p}(\boldsymbol{i\_h}_p, \boldsymbol{i\_t}_p) = \| \sin((\boldsymbol{i\_h}_p + \boldsymbol{r}_p - \boldsymbol{i\_t}_p)/2) \|_1 . \tag{8}$$

To sum up, we integrate these two parts, so that in polar coordinate system, HiRec can model both the items of the same layer and the items of the superior and subordinate relationship, the HiRec can be formulated as follows:

$$\begin{cases} \boldsymbol{i\_h}_m \circ \boldsymbol{r}_m = \boldsymbol{i\_t}_m \\ (\boldsymbol{i\_h}_p + \boldsymbol{r}_p) \mod 2\pi = \boldsymbol{i\_t}_p, \end{cases} \tag{9}$$

where $\boldsymbol{i\_h}_m, \boldsymbol{i\_t}_m \in \mathbb{R}^k, \boldsymbol{r}_m \in \mathbb{R}^k_+, \boldsymbol{i\_h}_p, \boldsymbol{r}_p, \boldsymbol{h\_t}_p \in [0, 2\pi)^k$.

The distance function for HiRec is

$$d_r(\boldsymbol{i\_h}, \boldsymbol{i\_t}) = \lambda_1 d_{r,m}(\boldsymbol{i\_h}_m, \boldsymbol{i\_t}_m) + \lambda_2 d_{r,p}(\boldsymbol{i\_h}_p, \boldsymbol{i\_t}_p), \tag{10}$$

where λ_1, λ_2 are hyperparameters which learned from the training model. The score function for HiRec is

$$f_r(\boldsymbol{i\_h}, \boldsymbol{i\_t}) = d_r(\boldsymbol{i\_h}, \boldsymbol{i\_t}) = -(\lambda_1 d_{r,m}(\boldsymbol{i\_h}, \boldsymbol{i\_t}) + \lambda_2 d_{r,p}(\boldsymbol{i\_h}, \boldsymbol{i\_t})). \tag{11}$$

Time and space complexity analysis

The time complexity of Paguridae is $O(n^3)$, the modulus part's complexity is the same as [52], which is $O(n)$, for the phase part, we aim $d_{r,p}(\boldsymbol{i\_h}_p, \boldsymbol{i\_t}_p)$ to be zero, according to Taylor's expansion, $sinx = x - \frac{x^3}{6} + o(x^3)$, which complexity is $O(n^3)$, and the space complexity is O(n).

By integrating the modulus part and the phase part, HiRec can model semantic hierarchies of recommendation systems.

3.2.3 Jointly learning layer

Paguridae extends the conventional recommendation model, BPRMF [53], by incorporating abundant knowledge of entities contained in KG. As we know, KG can bring side information for the downstream application such as recommendation systems to avoid cold start and data sparsity problems.

To train the link prediction task, we choose negative sampling with self-adversarial training [52] as loss function:

$$\mathcal{L}_1 = -\log\sigma(\gamma - d_r(\boldsymbol{h}, \boldsymbol{t})) - \sum_{i=1}^{n} p(h_i^{'}, r, t_i^{'}) \log\sigma(d_r(\boldsymbol{h}_i^{'}, \boldsymbol{t}_i^{'}) - \gamma), \tag{12}$$

where γ is a fixed margin, and $(h_i^{'}, r, t_i^{'})$ is the ith negative sample.

To train the recommendation module, we use bprLoss [53]:

$$\mathcal{L}_2 = -\sum_{i \in RS} \sum_{i' \notin RS} \log\sigma(g(u, i) - g(u, i^{'})), \tag{13}$$

where (u, i) is positive sample and $(u, i^{'})$ is negative sample. $g(u, i)$ and $g(u, i^{'})$ represent the score of each (u, i) and $(u, i^{'})$, respectively. Notice the i in Eq. (13) is divided into two parts(i.e., the modulus part and the phase part) as the embedding layer described to model the hierarchical structure of recommendation systems, finally we use BPRMF to calculate $g(u, i)$ and $g(u, i^{'})$.

We adopt an overall objective function to train Paguridae as follows:

$$\mathcal{L} = \lambda\mathcal{L}_1 + (1 - \lambda)\mathcal{L}_2, \tag{14}$$

where $\mathcal{L}_1$ and $\mathcal{L}_2$ represent the loss in link prediction task and loss in recommendation task, respectively. λ is a hyperparameter to weigh the performance of these two tasks.

In this layer, there are users and items embedding which integrates the abundant information from the KG, as well as entities and relationships that already exist in KG. They train together and interact with each other. The entity embedding can show the knowledge information in KG. Simultaneously, they can be fine-tuned by the connectivity with items during backpropagation.

3.2.4 Prediction layer

The final item recommendation for a user u is given according to the following ranking criterion [53]:

$$u : i_1 > i_2 > \cdots > i_n \rightarrow \boldsymbol{U}_u^T \boldsymbol{e}_{j1} > \boldsymbol{U}_u^T \boldsymbol{e}_{j2} > \cdots > \boldsymbol{U}_u^T \boldsymbol{e}_{jn}, \tag{15}$$

where i is item in recommendation systems, $\boldsymbol{U}_u$ is the embedding of user u, and $\boldsymbol{e}_{jn}$ is the embedding of item n.

The final link prediction result is based on the score function. We take all candidate entities to get the score function, and the higher, the order. For example, for a tuple $< h, r, ? >$, we adopt the link prediction method to calculate each candidate entity's score function f_1, f_2,..., f_n, as Eq. (16) shows:

$$< h, r, ? >: f_1 > f_2 > \cdots > f_n \rightarrow t_1, t_2, \ldots, t_n, \tag{16}$$

Table 4 Information of two datasets

Model	Attribute	MovieLens-1m	Last.FM
RS	Users	6040	1893
	Items	3,240	17,633
	Ratings	998,539	42,346
	Avg. ratings	165	22
KG	Entity	14,708	9367
	Relation	20	60
	Triple	434,189	1551

where n is the number of candidate entities. t_1, t_2,..., t_n represent the candidate entity for each scoring function f. Then, we will get the top n(i.e., 1,5,10) candidate entities as the link prediction result.

4 Evaluation and experiments

In this section, we evaluate our proposed framework on two real-world datasets: movie and music for recommendation scenarios. We find that when the average rating in the dataset is small, the accuracy of Paguridae is not obvious. But when the average rating is relatively large, Paguridae can be very advantageous. So we use two datasets(i.e., ML1m and Last.FM) in which their average.rating are quite suitable for our task.

The goal of this evaluation is to answer several questions:

- How does Paguridae perform compared with state-of-the-art item recommendation methods in recommendation system?
- How does Paguridae perform compared with state-of-the-art link prediction methods in knowledge graph?
- How much is the performance improved by adding the GCN layer?
- How does λ affect each task(i.e., item recommendation task in RS and link prediction task in KG)?

4.1 Datasets introduction

Following KTUP [43] and MKR [44], we use two publicly datasets in the movie and music domains: MovieLens-1m [47] and Last.FM [44] to evaluate the effectiveness of Paguridae.

We set the training, validation, and test ratio by 7:1:2 for each dataset. The relevant descriptions of these two datasets are shown in Table 4.

- MovieLens-1m. The MovieLens-1m data belong to the movie dataset. This dataset has large ratings and its average rating is also large which is suitable for item recommendation. In order to acquire the hierarchical structure in this dataset, we adopt a GCN model which contains two key parts. One is a 3240$\times$ 3240 adjacency matrix (3240 is the number of items), which describes the spatial structure between items. The other one is a feature matrix, which describes the attributes of items, and the hidden size is 200. After we got the spatial structure embedding of each item, we then combined RS with KG to a joint model Paguridae to get each result.

- Last.FM. The Last.FM data belong to the music dataset. Unlike the movie dataset, it has a large number of items in RS. In order to alleviate the experimental environment, for example, we filter out items with fewer than 10 [43] amounts which users have interacted with, and then operate similar to the movie dataset, notice the hidden size for Last.FM dataset is 1000, for the avg.rating is more minor than movie dataset so we need more hidden nodes to fit the data.

4.2 Competing methods

We will verify the efficiency of our model in two parts: Item Recommendation in Recommendation Systems and Link Prediction in Knowledge Graphs. In item recommendation task, the SOTA methods are Jointly Learning methods(Table 5) such as CoFM, MKR and KTUP. In link prediction task, the SOTA methods are Jointly Learning methods(Table 6) such as CoFM and KTUP.

4.2.1 Item recommendation in recommendation systems

We can divide these baselines into three types, traditional CF methods such as FM and BPRMF; KG-based methods such as CKE, CFKG, KGAT; jointly learning methods such as CoFM, MKR and KTUP.

- FM.
- BPRMF.

FM [54] and BPRMF [53] are two typically collaborative filtering models in recommendation system, they are the foundations of other baselines.

- CKE.

CKE [8] is a collaborative model which adopts link prediction method TransR to strengthen the effectiveness of item recommendation.

- CFKG.

CFKG [5] views the recommendation task as the link prediction task in the knowledge graph. The purchasing relationship between users and items is modeled, and then use TransE method for prediction.

- CoFM.

CoFM [7] makes an alignment of items and entities, and then jointly trains TransE method and FM method by their sharing parameters or regularization.

- MKR.

MKR [44] contains two parts, one is recommendation part and the other is KGE part. The recommendation part is used to learn the representation of each users and items, while the KGE part is used to learn the representation of items that have an association with entities in KG.

- KGAT.

KGAT [6] firstly proposed a CKG structure, which is a united graph constructed by users, items and entities. KGAT is divided into Embedding Layer and Attentive Embedding Propagation Layer.

Table 5 Performance with different methods on Item Recommendation Task

Type	Method	MovieLens-1m (@10, %)					Last.FM (@10, %)				
		Precision	Recall	F1	Hit	NDCG	Precision	Recall	F1	Hit	NDCG
Traditional CF methods	FM	29.28	11.92	13.81	81.06	59.48	3.31	7.41	4.35	25.88	16.09
	BPRMF	30.81	12.95	14.84	83.18	61.02	3.72	8.38	4.89	28.99	18.29
KG-based methods	CKE	38.67	16.65	18.94	88.36	67.05	3.83	8.42	4.98	31.07	**21.06**
	CFKG	29.45	12.49	14.23	82.24	58.97	4.09	8.95	5.33	31.46	21.5
	KGAT	–	–	–	–	–	**4.18**	5.68	4.81	–	–
Jointly learning methods	CoFM	32.08	13.02	15.12	83.3	58.69	3.03	6.6	3.95	24.73	15.79
	MKR	–	–	12.02	–	–	–	–	5.09	–	-
	KTUP	41.03	17.25	19.82	89.03	69.92	3.63	8.22	4.77	29.32	17.82
	Paguridae	**61.26**	**27.43**	**32.21**	**92.76**	**89.64**	**4.18**	**9.54**	**5.5**	**32.82**	20.66

Best values in this experimental results are shown in bold

Table 6 Performance with different methods on Link Prediction Task

Type	Method	MovieLens-1m (%)		Last.FM (%)	
		MR	Hit@10	MR	Hit@10
Translation-based methods	TransE	537	46.95	1461	54.29
	TransH	537	47.63	1463	54.29
	TransR	609	38.93	1678	50.9
KG-based methods	CFKG	523	41.56	1479	53.93
	CKE	585	34.37	1709	51.03
Jointly learning methods	CoFM	506	46.62	1437	54.58
	KTUP	527	48.9	1574	46.55
Sematic structure and relation modeling	Paguridae	**326**	**53**	**586**	**56.27**

Best values in this experimental results are shown in bold

- KTUP.

KTUP [43] adopts the framework of multi-tasking learning, which can carry out recommendation task and knowledge graph completion task simultaneously. However, it does not take into account the semantic structure of knowledge.

4.2.2 Link prediction in knowledge graphs

We divided these methods into three types, they are Translation-based methods such as TransE, TransH and TransR; KG-based methods such as CFKG and CKE; Jointly Learning methods such as CoFM and KTUP.

- Translation model.

TransE, TransH, TransR, etc., are translation-based methods. These models view relationships as a transition between head-to-tail entities, and they are good at modeling patterns such as inverted relationships.

- CKE.

CKE [8] is a collaborative model which adopts link prediction method TransR to strengthen the effectiveness of item recommendation.

- CFKG.

CFKG [5] views the recommendation task as the link prediction task in the knowledge graph. The purchasing relationship between users and items is modeled, and then use TransE method for prediction.

- CoFM.

CoFM [7] makes an alignment of items and entities, and then jointly trains TransE method and FM method by their sharing parameters or regularization.

- KTUP.

KTUP [43] adopts the framework of multi-tasking learning, which can carry out recommendation task and knowledge graph completion task simultaneously.

4.3 Performance comparison

4.3.1 Effectiveness of the item recommendation task

For this part, we aim to evaluate our models and the state-of-the-art methods on the item recommendation task based on two datasets. The overall performance on item recommendation task is as Table 5 shows.

- Metrics.

We evaluate the item recommendation performance with Hit, NDCG, Precision, Recall, and F1-Score metrics.

Precision: Precision is the probability of correctly predicting a positive sample in a sample with a positive forecast.

$$Precision = \frac{TP}{TP + FP}. \tag{17}$$

Recall: In RS, recall can be viewed as the ratio of the item found to the total item that user has interacted with.

$$Recall = \frac{TP}{TP + FN}. \tag{18}$$

F1-Score: It is the trade-off between recall and precision.

$$F1 = \frac{2 * Recall * Precision}{Recall + Precision}. \tag{19}$$

Hit: If the positive item appears in the recommended N items, the Hit would be 1, otherwise, it would be 0. In Top-K recommendation, HR is a commonly used index to measure recall rate.

$$Hit\ ratio@N = \frac{NumberOfHits@N}{GT}. \tag{20}$$

where GT is the set of all the tests.

NDCG: DNCG is a standard index in sorting algorithms, which is called Normalized Discounted Cumulative Gain. We use NDCG to measure the ranking performance of recommended items.

$$NDCG@N = \frac{DCG@N}{IDCG}. \tag{21}$$

where IDCG represents the list of the best recommendation results returned by a user in the recommendation system, and the most relevant results are placed first.

- Parameter setting.

For Item Recommendation task, we try to tune the key parameters and find that when we set -l2_lambda as 1e-5, negative_samples as 1, batch_size as 1,024, embedding_size as 100, and learning_rate as 0.005, the performance of item recommendation task could reach the optima.

- Findings.

We test the performances of FM, BPRMF, CKE, CFKG, CoFM, KTUP, MKR, KGAT and Paguridae in this paper on two datasets, which results are illustrated in Figs. 7, 8and 9, in which Fig. 7shows an overall improvement of our Paguridae compared with other state-of-the-art methods. For our experiment, each result is the best one of multiple runs, and we

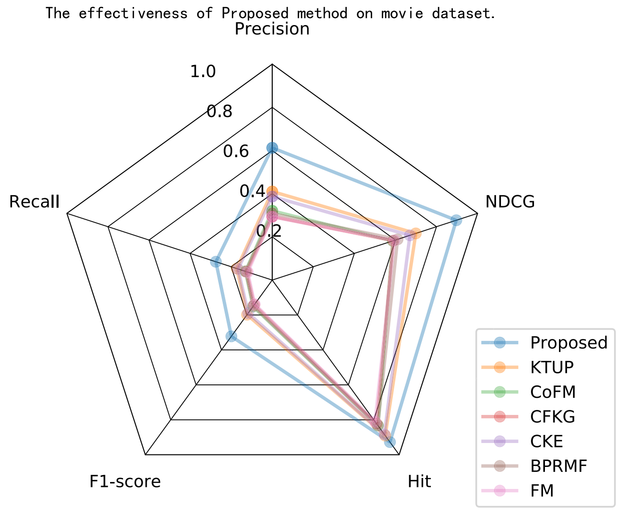


Fig. 7 The effectiveness of proposed model on movie dataset

use Adagrad as the gradient descent function in item recommendation task. As can be seen from our experimental results, the performance and accuracy of recommendation systems without knowledge graph such as FM are much worse than those with knowledge graph such as Paguridae. Traditional methods may face the cold start and data sparsity problems. We can see that our proposed Paguridae has improved all metrics on dataset ml1m, especially it improves F1-score by 62.51% and precision by 49.31% compared to state-of-the-art method in recommendation systems. Because Paguridae obtain the better hierarchical structure than other methods that focus on preference induction. When complex relational reasoning occurs, such as exploring the transformation of relationships, the knowledge-based methods may be more dominant. Paguridae adopt two key parts to acquire the hierarchical structure, one is GCN model and the other is HIREC model, which is illustrated specifically in 3.2.2. As a conclusion, Paguridae performs better compared with state-of-the-art item recommendation methods.

4.3.2 Effectiveness of the link prediction task

For this part, we aim to evaluate our models as well as the baseline methods on the link prediction task based on two datasets. The overall performance on link prediction task is as Table 6 shows.

- **Metrics.**

We evaluate the link prediction performance with MeanRank and Hit@10 metrics.

MeanRank: First, for each testing triple, taking the predictive tail entity(t) as an example, we replace the $\langle h, r, t\rangle$ with each entity in the knowledge graph, and then calculate the score using the function f(h,t) so that we can get a series of scores, which are then arranged in ascending order, the average of these rankings is the MeanRank, and the smaller, the better.

Hit@10: Rank f functions as described above, then check to see if each testing triple $\langle h \ or \ t\rangle$ is in the top ten in the sequence. If it is, count +1, the final top ten number/total number is Hit@10.

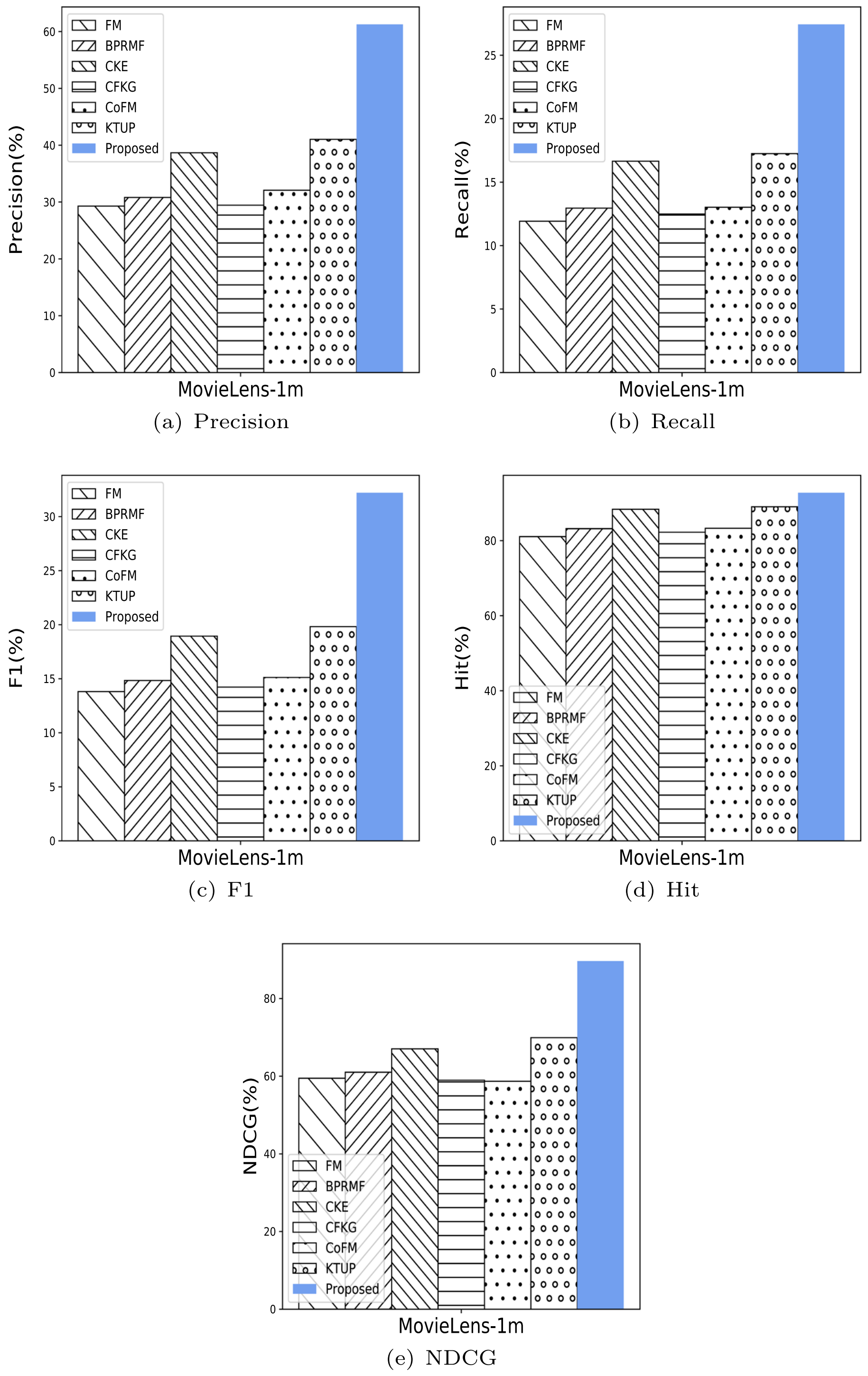


Fig. 8 The item recommendation performances on mL1m dataset

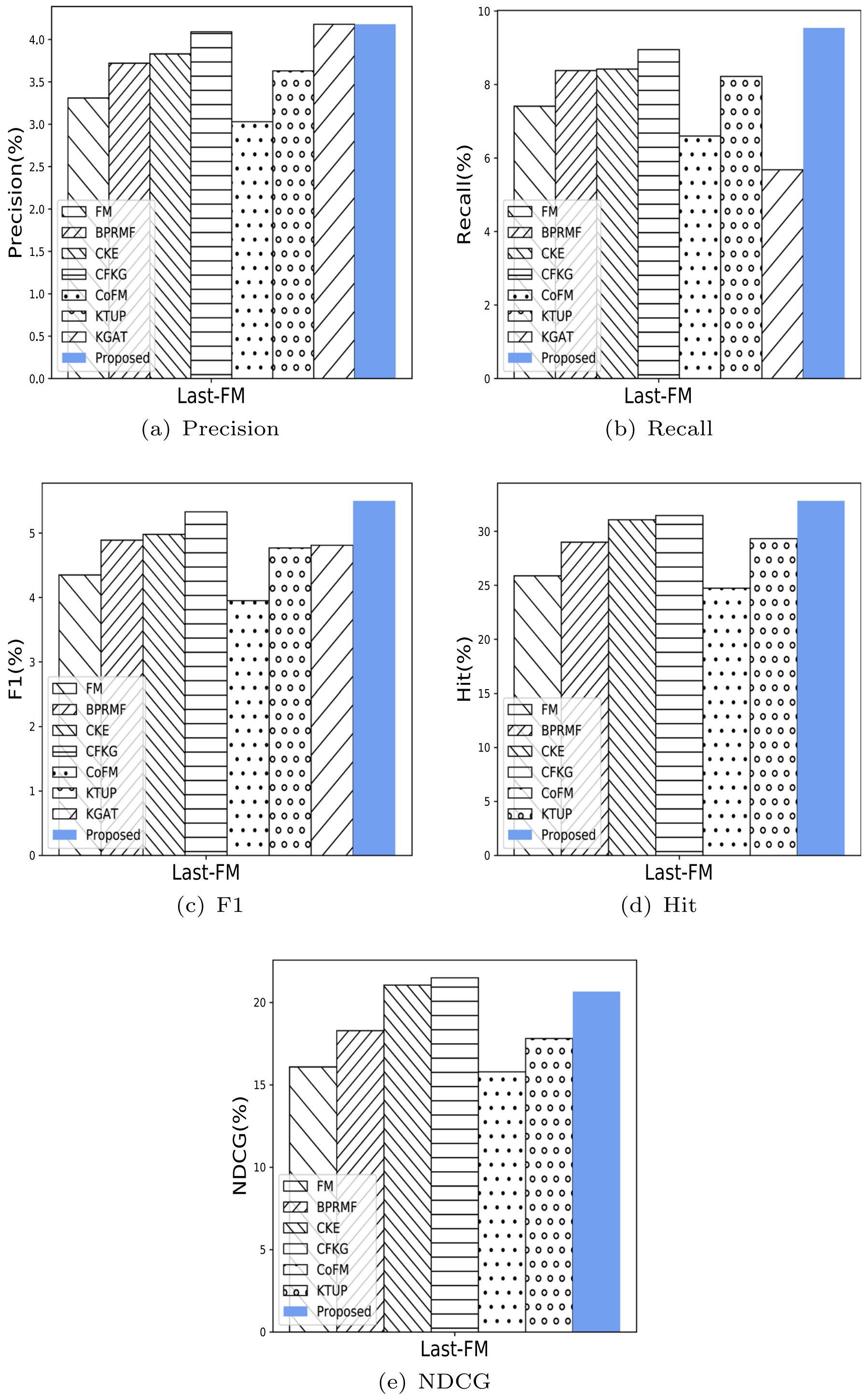


Fig. 9 The item recommendation performances on music dataset

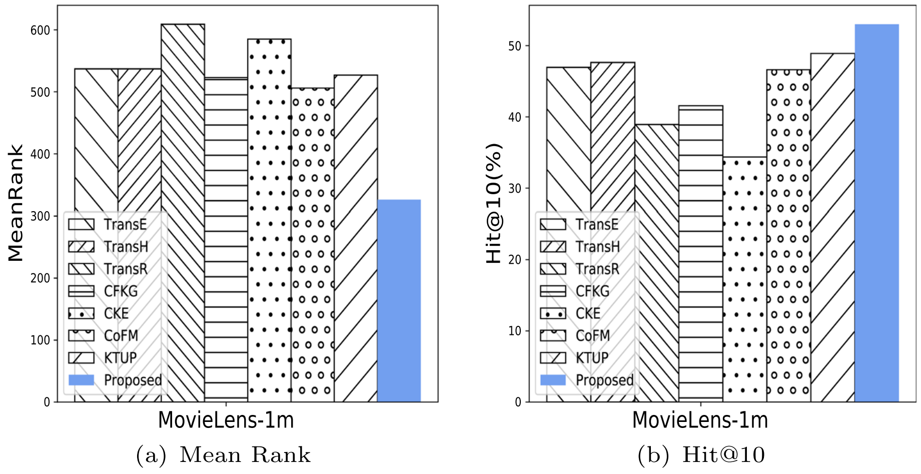


Fig. 10 The link prediction performances on mL1m dataset

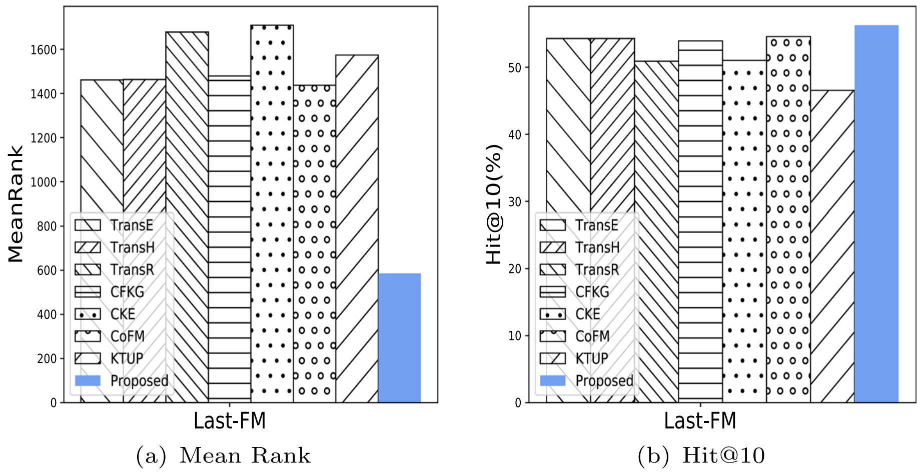


Fig. 11 The link prediction performances on music dataset

- **Parameter Setting.**

For link prediction task, we try to tune the key parameters and find that when we set l2_lambda as 0, batch_size as 256, embedding_size as 1000, and learning_rate as 0.001, the performance of link prediction task could reach the optima.

- **Findings.**

We test the performances of Translation models(i.e., TransE, TransH, TransR), CFKG, CKE, CoFM, KTUP and Paguridae in this paper on two datasets, which results are illustrated in Figs. 10and 11. For our experiment, each result is the best one of multiple runs, and we use Adam as the gradient descent function in link prediction task. We can see that our proposed Paguridae has improved all metrics on dataset ml1m and Last.FM. This is due to Paguridae is a joint learning model which can benefit both tasks in item recommendation and link prediction, we can see that translation-based models are the worst as they train link

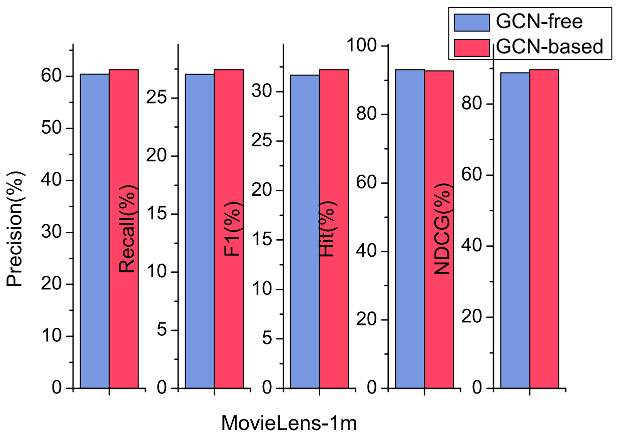
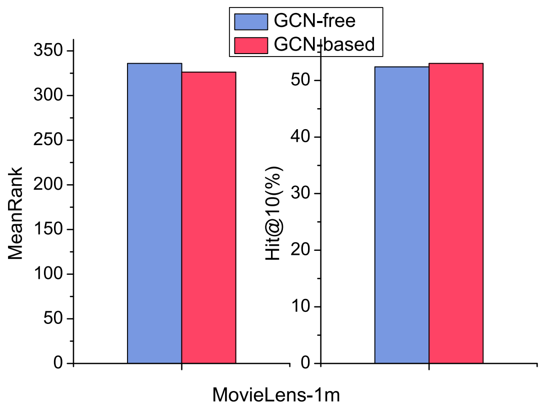


Fig. 12 The effectiveness of GCN model on item recommendation task

Fig. 13 The effectiveness of GCN model on link prediction task

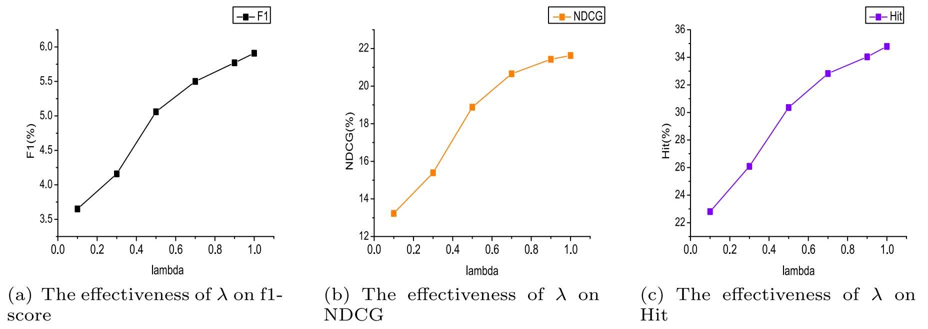


prediction task only, the Paguridae performs best among these methods as it considers the incompleteness as well as the hierarchical structure between entities than other methods. As a conclusion, Paguridae performs better compared with state-of-the-art knowledge graph completion methods.

4.3.3 Effectiveness of the GCN layer

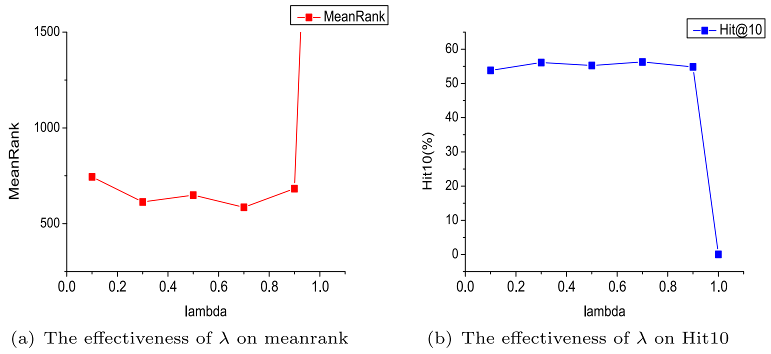
In order to verify the influence of GCN module on the Paguridae model, we further conducted an ablation experiment.

- **Metrics.** The metrics are the same as mentioned above.
- **Findings.** We test the performances of GCN-free model and proposed method Paguridae(GCN-based) model in this paper on ml1m dataset, which results are illustrated in Figs. 12 and 13. We can see that our proposed Paguridae model improves F1 by 1.74% in Fig. 12, which illustrates the effectiveness of the GCN model, the reason is that GCN can better capture the topological structure(i.e., hierarchical structure) between items in RS, so that the GCN model can make Paguridae convergence faster, loss lower



(a) The effectiveness of λ on f1-score (b) The effectiveness of λ on NDCG (c) The effectiveness of λ on Hit

Fig. 14 The influence of λ on item recommendation task. (To observe the effect more directly, we set a scale for the y axis)



(a) The effectiveness of λ on meanrank (b) The effectiveness of λ on Hit10

Fig. 15 The influence of λ on link prediction task

and accuracy higher. Based on this, it is better to systematically analyze the data complexities and characteristics in RS which can improve the effect of item recommendation. Our proposed Paguridae improves MeanRank by 2.98% than GCN-free model in Fig. 13, this is because our Paguridae is a joint learning model, we can improve the link prediction task as well as item recommendation task.

4.3.4 Influence of the λ for each of the two tasks

In order to verify the impact of λ on the two parts of the task, we conducted experiments on music dataset for different λ, as shown in Figs. 14and 15.

- The influence of λ on item recommendation task.

As shown in Fig. 14, with the increment of λ, item recommendation in the RS shows an excellent upward trend. However, as our framework Paguridae is a joint training model, will the linking prediction task of knowledge graph introduced as side information also perform well? How to balance the λ between these two tasks? We also conducted an experiment on the influence of λ on link prediction task.

- The influence of λ on link prediction task.

As shown in Fig. 15, the MeanRank performs best when the λ is 0.7, with the increment of λ, not only MeanRank but also Hit@10 performed the worst compared with other values.

Therefore, to achieve a balance between the two tasks, we settled on a value of 0.7 for λ. As for our further work, we plan to choose λ automatically for different tasks or datasets other than manual tuning parameters.

5 Conclusion

In this work, we proposed a new framework Paguridae, which can train item recommendation task and link prediction task together. To model the semantic hierarchies in real-world datasets, we first adopt GCN model on item graphs, which performs better based on our experiments, then we use the polar coordinate system as the representation space, which can combine with our model so as to achieve the goal that the completion of KG can benefit from the improved modeling of user-item interactions. The experiment results show that our proposed Paguridae effectively outperforms existing state-of-the-art methods on two datasets for both the recommendation task and link prediction task.

For future work, we will incorporate other KGE methods as the implementation of KGE module in Paguridae as users in RS may have a unique structure. We also plan to choose λ automatically for different tasks or datasets other than manual tuning parameters.

References

1. Jamali M, Ester M (2010) A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the fourth ACM conference on recommender systems, pp 135–142
2. Feng F, He X, Wang X, Luo C, Liu Y, Chua T-S (2019) Temporal relational ranking for stock prediction. ACM Trans Inf Syst 37(2):1–30
3. Cheng Z, Ding Y, Zhu L, Kankanhalli M (2018) Aspect-aware latent factor model: rating prediction with ratings and reviews. In: Proceedings of the 2018 World Wide Web conference, pp 639–648
4. Catherine R, Cohen W (2016) Personalized recommendations using knowledge graphs: A probabilistic logic programming approach. In: Proceedings of the 10th ACM conference on recommender systems, pp 325–332
5. Zhang Y, Ai Q, Chen X, Wang P (2018) Learning over knowledge-base embeddings for recommendation. arXiv preprint arXiv:1803.06540
6. Wang X, He X, Cao Y, Liu M, Chua T-S (2019) Kgat: knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining, pp 950–958
7. Piao G, Breslin JG (2018) Transfer learning for item recommendations and knowledge graph completion in item related domains via a co-factorization model. In: European semantic web conference, pp 496–511. Springer
8. Zhang F, Yuan NJ, Lian D, Xie X, Ma W-Y (2016) Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD International conference on knowledge discovery and data mining, pp 353–362
9. Zhang Z, Cai J, Zhang Y, Wang J (2020) Learning hierarchy-aware knowledge graph embeddings for link prediction. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 3065–3072
10. Rossi A, Barbosa D, Firmani D, Matinata A, Merialdo P (2021) Knowledge graph embedding for link prediction: a comparative analysis. ACM Trans Knowl Discovery Data 15(2):1–49
11. Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems, pp 2787–2795
12. Wang Z, Zhang J, Feng J, Chen Z (2014) Knowledge graph embedding by translating on hyperplanes. In: Twenty-eighth AAAI conference on artificial intelligence
13. Lin Y, Liu Z, Sun M, Liu Y, Zhu X (2015) Learning entity and relation embeddings for knowledge graph completion. In: Twenty-ninth AAAI conference on artificial intelligence
14. Yang S, Tian J, Zhang H, Yan J, He H, Jin Y (2019) Transms: knowledge graph embedding for complex relations by multidirectional semantics. In: IJCAI, pp 1935–1942

15. Cui Z, Liu S, Pan L, He Q (2020) Translating embedding with local connection for knowledge graph completion. In: Proceedings of the 19th international conference on autonomous agents and multiagent systems, pp 1825–1827
16. Nickel M, Tresp V, Kriegel H-P (2012) Factorizing yago: scalable machine learning for linked data. In: Proceedings of the 21st international conference on world wide web, pp 271–280
17. Yang B, Yih W-t, He X, Gao J, Deng L (2014) Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575
18. Nickel M, Rosasco L, Poggio T (2016) Holographic embeddings of knowledge graphs. In: Thirtieth Aaai conference on artificial intelligence
19. Plate TA (1995) Holographic reduced representations. IEEE Trans Neural Netw 6(3):623–641
20. Trouillon T, Welbl J, Riedel S, Gaussier É, Bouchard G (2016) Complex embeddings for simple link prediction. In: International conference on machine learning (ICML)
21. Balažević I, Allen C, Hospedales TM (2019) Hypernetwork knowledge graph embeddings. In: International conference on artificial neural networks, Springer, pp 553–565
22. Balažević I, Allen C, Hospedales TM (2019) Tucker: tensor factorization for knowledge graph completion. arXiv preprint arXiv:1901.09590
23. Tran HN, Takasu A (2020) Multi-partition embedding interaction with block term format for knowledge graph completion. arXiv preprint arXiv:2006.16365
24. Dettmers T, Minervini P, Stenetorp P, Riedel S (2018) Convolutional 2d knowledge graph embeddings. In: Thirty-second AAAI conference on artificial intelligence
25. Nguyen DQ, Nguyen TD, Nguyen DQ, Phung D (2017) A novel embedding model for knowledge base completion based on convolutional neural network. arXiv preprint arXiv:1712.02121
26. Guo L, Sun Z, Hu W (2019) Learning to exploit long-term relational dependencies in knowledge graphs. arXiv preprint arXiv:1905.04914
27. Kim Y (2014) Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882
28. Mikolov T, Karafiát M, Burget L, Černockỳ J, Khudanpur S (2010) Recurrent neural network based language model. In: Eleventh annual conference of the international speech communication association
29. Li Z, Liu H, Zhang Z, Liu T, Xiong NN (2021) Learning knowledge graph embedding with heterogeneous relation attention networks. IEEE Trans Neural Netw Learn Syst
30. Vashishth S, Sanyal S, Nitin V, Talukdar P (2019) Composition-based multi-relational graph convolutional networks. arXiv preprint arXiv:1911.03082
31. Cai L, Yan B, Mai G, Janowicz K, Zhu R (2019) Transgcn: Coupling transformation assumptions with graph convolutional networks for link prediction. In: Proceedings of the 10th international conference on knowledge capture, pp 131–138
32. Tian A, Zhang C, Rang M, Yang X, Zhan Z (2020) Ra-gcn: relational aggregation graph convolutional network for knowledge graph completion. In: Proceedings of the 2020 12th international conference on machine learning and computing, pp 580–586
33. Zeb A, Haq AU, Zhang D, Chen J, Gong Z (2021) KGEL: a novel end-to-end embedding learning framework for knowledge graph completion. Expert Syst Appl 167:114164
34. Wang Y, Zhang H (2021) Harp: a novel hierarchical attention model for relation prediction. ACM Trans Knowl Discovery Data 15(2):1–22
35. Schafer JB, Frankowski D, Herlocker J, Sen S (2007) Collaborative filtering recommender systems. In: The adaptive web: methods and strategies of web personalization, pp 291–324
36. Beidas RS, Kendall PC (2010) Training therapists in evidence-based practice: a critical review of studies from a systems-contextual perspective. Clin Psychol: Sci Pract 17(1):1–30
37. Piao G, Breslin JG (2017) Factorization machines leveraging lightweight linked open data-enabled features for top-n recommendations. In: International conference on web information systems engineering. Springer, pp 420–434
38. Fu Z, Xian Y, Gao R, Zhao J, Huang Q, Ge Y, Xu S, Geng S, Shah C, Zhang Y et al (2020) Fairness-aware explainable recommendation over knowledge graphs. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, pp 69–78
39. Geng S, Fu Z, Tan J, Ge Y, De Melo G, Zhang Y (2022) Path language modeling over knowledge graphsfor explainable recommendation. In: Proceedings of the ACM web conference 2022, pp 946–955
40. Wang Z, Lin G, Tan H, Chen Q, Liu X (2020) 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, pp 219–228
41. Liu X, Song R, Wang Y, Xu H (2022) A multi-granular aggregation-enhanced knowledge graph representation for recommendation. Information 13(5):229
42. Sakurai K, Togo R, Ogawa T, Haseyama M (2022) Deep reinforcement learning-based music recommendation with knowledge graph using acoustic features. ITE Trans Media Technol Appl 10(1):8–17

43. Cao Y, Wang X, He X, Hu Z, Chua T-S (2019) Unifying knowledge graph learning and recommendation: towards a better understanding of user preferences. In: The World Wide Web Conference, pp 151–161
44. Wang H, Zhang F, Zhao M, Li W, Xie X, Guo M (2019) Multi-task feature learning for knowledge graph enhanced recommendation. In: WWW, San Francisco, pp 2000–2010
45. Xin X, He X, Zhang Y, Zhang Y, Jose J (2019) Relational collaborative filtering: modeling multiple item relations for recommendation. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, pp 125–134
46. Heitmann B (2012) An open framework for multi-source, cross-domain personalisation with semantic interest graphs. In: Proceedings of the sixth ACM conference on recommender systems, pp 313–316
47. Noia TD, Ostuni VC, Tomeo P, Sciascio ED (2016) Sprank: semantic path-based ranking for top-n recommendations using linked open data. ACM Trans Intell Syst Technol 8(1):1–34
48. Levy O, Goldberg Y (2014) Neural word embedding as implicit matrix factorization. Adv Neural Inf Process Syst 27:2177–2185
49. Goyal P, Ferrara E (2018) Graph embedding techniques, applications, and performance: a survey. Knowl-Based Syst 151:78–94
50. Wang Y, Duan Z, Liao B, Wu F, Zhuang Y (2019) Heterogeneous attributed network embedding with graph convolutional networks. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 10061–10062
51. Zhao L, Song Y, Zhang C, Liu Y, Wang P, Lin T, Deng M, Li H (2019) T-gcn: a temporal graph convolutional network for traffic prediction. IEEE Trans Intell Transp Syst 21(9):3848–3858
52. Sun Z, Deng Z-H, Nie J-Y, Tang J (2019) Rotate: knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197
53. Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2012) BPR: Bayesian personalized ranking from implicit feedback. arXiv preprint arXiv:1205.2618
54. Rendle S (2010) Factorization machines. In: 2010 IEEE international conference on data mining, pp 995–1000. IEEE

Haizhou Du received his Ph.D. degree in computer science and technology from Tongji University, Shanghai, China. His research interests include Machine Learning, Data Management, Distributed System.

Yue Tang received the B.Eng. degree in Shanxi University of Finance and Economics, Taiyuan, China, in 2018, and the M.Eng. degree in computer technology, Shanghai University of Electronic Power, Shanghai, China, in 2022. Her research interests include Link prediction in knowledge graph and recommendation systems.

Zebang Cheng received the B.Eng. degree in computer technology from Shanghai University of Electric Power, Shanghai, China, in 2021. He is pursuing the Master degree in Shanghai University of Electronic Power. His research interests include data mining, data analysis.