A Hierarchical Knowledge-based Joint Learning Framework for Recommendation

Hongru Sun[1,2,a,*], Wancheng Ni[1,b] and Yiping Yang[1,c]

[1]Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
a. sunhongru2017@ia.ac.cn, b. wancheng.ni@ia.ac.cn, c. yiping.yang@ia.ac.cn
**corresponding author: Hongru Sun*

Keywords: User fine-grained preference, domain difference, hierarchical knowledge-based recommendation.

Abstract: Knowledge graphs (KGs), which provide rich semantic information, have proven to be effective in alleviating sparsity and cold start problems in recommender systems. Most existing KG-based methods concentrate on modeling relationship between users and items with constant item-attribute triplets, but neglect the particularity of the KG in the recommendation scenario, that is, the item-attribute triplets should have different importance. In this work, we propose a hierarchical knowledge-based recommendation model (HKRM), which exploits the item-attribute hierarchy and jointly learn knowledge representation with recommender system. By measure the proximity between items and attributes, we obtain two distributions: user preference and the candidate item's attributes importance. Then these two distributions are respectively transformed into the user representation and item representation, which are used in the recommender system in a duet matching way. To eliminate the domain difference, we propose a hypothesis based on distribution distance and a joint learning method to guide the learning of knowledge representation. We conduct extensive experiments on two datasets related to movies and books. The results demonstrate a significant improvement over the state-of-the-art baselines.

1. Introduction

To build an effective recommender system, a key factor is to accurately capture and understand user interest, which is very difficult to achieve without auxiliary information. To attain this goal, many researchers [1][2][3][4] have tried to incorporate item-side information into recommender system. Recently, knowledge graph (KG) stands out from all types of auxiliary data due to its fruitful background knowledge and connections. A typical KG is usually represented in the form of knowledge triple, which is denoted by (h,r,t), where h and t correspond to the head and tail entities respectively and r denotes the relation between them.

For recommendation, a type of important information in KG are the connections between an item and its attributes, which are represented by multiple constant triplets. But only utilizing constant knowledge connections in KG is not sufficient for recommender system. In the case of *Titanic* shown in Figure 1, it may be liked by users because it stars *Leonardo Dicaprio*, or because it is directed by *James Cameron*, or because it is a disaster movie, but not likely because it's in *English*. And for a user *u*, we assume that he has seen *Avatar*, the *Terminator*, and *True Lies*, and we can infer that user *u* may be a fan of *James Cameron*. The user will choose the movie according to the attribute of director and is less sensitive to other attributes, which could be regarded as the fine-grained preference of user *u*. When the candidate item is *Titanic*, we have more confidence that the user will like it because of the fine-grained preference we captured.

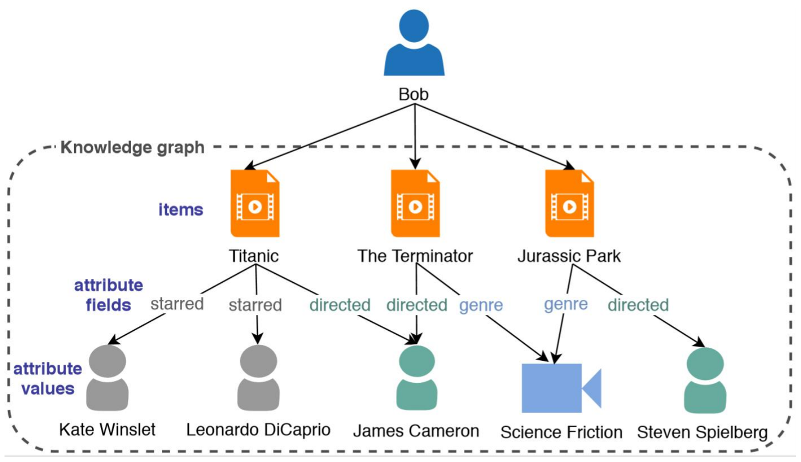


Figure 1: Example of hierarchical structure in KG for recommendation.

It is worth emphasizing that user fine-grained preference cannot be captured by considering only a single triple, instead, all relevant triplets of the user and item should be considered at the same time. We argue that all the triplets associated with the user constitute a hierarchical knowledge structure, as shown in Figure 1, whose smallest unit is an item-specific group of item-attribute triplets. This kind of knowledge structure establishes the connections between user-item-attribute, distinguishes different attribute fields, and can effectively represent users' fine-grained preferences.

Having the hierarchical knowledge structure, we need to design the knowledge representation model in the recommender system. The model needs to cover the following two aspects at the same time: (1) distinguishing between item entities and attribute entities in KG; (2) considering the varying degrees of importance of all attributes in item representation. However, most KG-based recommendation methods can not cover these two aspects. For example, Embedding-based methods [4][5][6] do not distinguish between items and attributes, simply leveraging the information from the connections in triplets, which might lead to deficiency in capturing fine-grained user preference. Path-based approaches [7][8][9][10] take into account the differences between items and attributes in KG but they only consider the paths between the user and the candidate item instead of all the attributes with different importance.

Another key problem of KG-based recommendation system is how to eliminate domain difference between knowledge and recommendation. KG-based methods usually represent the knowledge as a vector by using knowledge graph embedding (KGE) methods. Some KGE methods [11][12][13][14], assume that the embedding of tail $\boldsymbol{t}$ should be in the neighbourhood of h+r, and simply model the connections of knowledge triples in an energy function like $\| f_r(h)+r-f_r(t)\|$ [15]. These KGE methods concentrate on modeling the constant connection in a triple, while recommender systems need to consider the connections of all related triplets with different importance as aforementioned. However, most of existing KG-based recommender systems [3][5][6] directly use KGE methods model to learn the knowledge representation and then apply it in recommendation. These two-stage methods don't consider the domain difference, so that they can't ensure the learned knowledge vector suitable for recommendation. We argue that jointly training the knowledge representation and recommendation models, which integrate multiple related triples, can eliminate the domain difference.

To solve these problems, we propose the hierarchical knowledge-based recommendation model (HKRM), an effective method to model the item-attribute hierarchy and jointly learning knowledge representation learning with recommender system. In HKRM, we first represent user preference and candidate item at item-level. Then, we respectively calculate the attribute importance distribution of user interest and candidate item based on proximity measure inspired by DisMult model [16]. Furthermore, the attribute-level representations of user interest and candidate item are generated by the proximity. To jointly train the knowledge representation and recommendation model, we propose a hypothesis: If user *u* likes item *v*, the distribution of *u*'s preference for each attribute is closer to the distribution of *v*'s importance for each attribute, which leverages user's historical feedback (e.g. clicks, purchases, likes) to guide the learning of knowledge representation. Specifically, we design a distance loss associated with feedback on the aforementioned two distributions and jointly optimize it with the click objective function.

The contributions of this work are summarized as follows:

(1) We highlight the effectiveness of item-attribute hierarchy for capturing user fine-grained preference and design a hierarchical knowledge representation model for recommendation.

(2) We propose a joint training framework, which combines the hierarchical knowledge representation and recommendation model intrinsically, to eliminate the domain difference.

(3) Extensive experiments on two datasets show that our model outperforms state-of-the-art baselines, proving the effectiveness of our proposed method.

2. Method

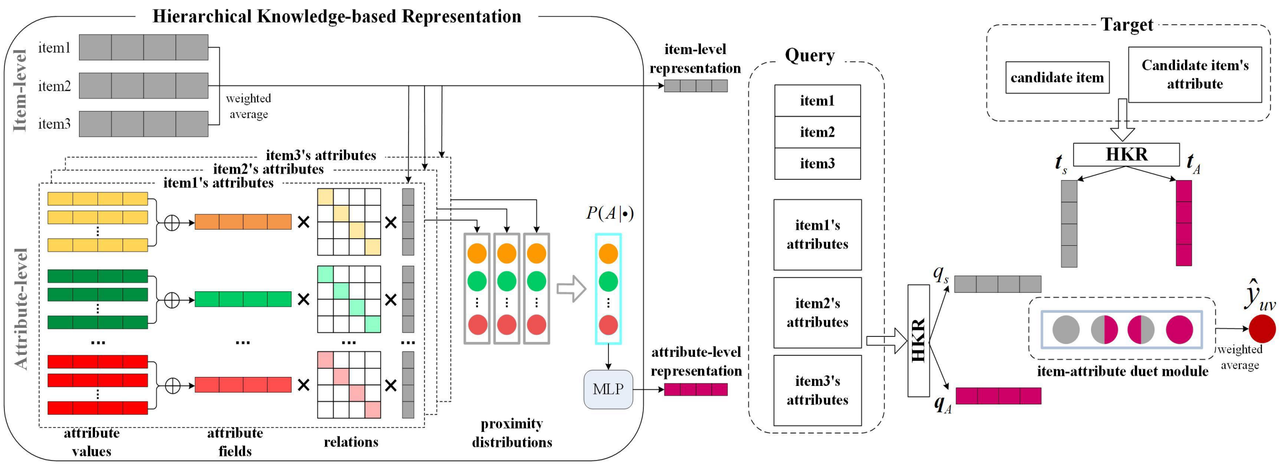


Figure 2: The over all framework of the HKRM.

In this section, we first formulate the problem, and then we present our proposed method in detail. As illustrated in Figure 2, taking a query as an input to HKR, we first obtain its item-level representation by calculating the weighted average of item embeddings. Then, the attribute-level representation of the query is obtained based on the proximity distribution. Similarly, we get both level representations of the target by HKR. By the query matching framework, the click probability of target item can be obtained.

2.1. Problem Formulation

Let $\mathrm{U} = \{u_1, u_2, ...\}$ denote the user set and $\mathrm{V} = \{v_1, v_2, ...\}$ denote the item set. The user-item interaction data is represented by $\mathrm{Y} = \{y_{uv} \mid u \in \mathrm{U}, v \in \mathrm{V}\}$, where $y_{uv} = 1$ if there is an observed interaction (e.g., rate, click feedbacks) between user u and item v, otherwise $y_{uv} = 0$. We define the knowledge graph as $\mathrm{G} = \{(e_h, r, e_t) \mid e_h, e_t \in \mathrm{E}, r \in \mathrm{R}\}$, where E denotes the entity set, R denotes the relation set, and each triplet (e_h, r, e_t) indicates a fact that there is a relationship r from head entity e_h to tail entity e_t. To integrate the knowledge graph into the recommender system, the item set and the entity set are merged by string matchings. In other words, recommender system's item set V can be considered as a subset of knowledge graph's entity set E, so we have $\mathrm{V} \subseteq \mathrm{E}$ .Given a user-item pair, our goal is to estimate the probability (denoted as $\hat{y}_{uv}$) that user u will click item v which he has not seen before.

2.2. Hierarchical Knowledge-based Representation

As discussed in Section 1, hierarchical structure of KG can adequately represent user interest at different granularity (i.e., item-level and attribute-level). Therefore, based on the hierarchical knowledge, we propose an effective module, which can generate both item and attribute-level representations of user (query q) and candidate item (target t). The two representations will be further elaborated as follows.

2.2.1. Item-level Representation

The Item-level representation is directly obtained from item entities in the KG. Let $\mathbf{q}_s$ denote the item-level representation of query q. The item entities in the query arouse the interest of users to varying degrees. For instance, user may prefer movie *Titanic* to *Fantasia*. Therefore, for query q we should maintain a learnable weight vector $\boldsymbol{\alpha} \in \mathbb{R}^m$, where m denotes the number of entities in query q. Then the item-level representation for q is calculated by:

$$\mathbf{q}_s = \sum_i \alpha_i \mathbf{e}_i \qquad (1)$$

where $\mathbf{e}_i \in \mathbb{R}^d$ is the embedding vector of entity e_i in q, α_i is the corresponding weight. To keep $\sum_i \alpha_i = 1$, α_i is obtained by:

$$\alpha_i = \mathbf{softmax}\left(\frac{p_i}{\sqrt{|q|}}\right) \qquad (2)$$

where p_i is a learnable parameter that are randomly initialized with a uniform distribution, and $|q|$ is the size of a query. As for target, the item-level representation is expressed as $\mathbf{t}_s$, which is the embedding vector of the target.

2.2.2. Attribute-level Representation

In order to mine user preference at a finer granularity, we define attribute-level representation based on the attribute entities and relations in KG. Let $A = \{a_j \mid j = 1, ..., n\}$ denote the set of attribute fields. It is obvious that all the items in a dataset share the same A. For example, in the movie recommendation, items share the fields of *director*, *actor*, *writer*, *genre*, etc. Similarly, book has *writer*, *language*, *genre* fields and so on. However, the attribute value set of different items is item-specific and the quantity of it is very large. We use $\mathbf{h}_{jk}^{e} \in \mathbb{R}^{d}$ to denote the vector of the k-th attribute value in field a_j for item e. For representing user preference, the attribute value is too fine grained to be suitable. Therefore, we aggregate the attribute values by attribute field:

$$\mathbf{o}_{j}^{e} = \sum_{k} \mathbf{h}_{jk}^{e} \tag{3}$$

Furthermore, after merging the attribute values, the relation between item and attribute is symmetrical (one-to-one). Next, we model the proximity distribution so as to construct attribute-level representation of query and target.

2.2.3. Proximity Distribution

For each e_i in query q, the attribute field vector in a_j is $\mathbf{o}_{j}^{e_i}$. At first, we calculate the proximity between the query and every attribute field a_j in a semantic matching way:

$$p(a_j \mid q) = \sum_{i=1}^{m} \frac{exp(\mathbf{o}_{j}^{e_i T} \mathbf{R}_j \mathbf{q}_s)}{\sum_{j=1}^{n} exp(\mathbf{o}_{j}^{e_i T} \mathbf{R}_j \mathbf{q}_s)} \tag{4}$$

where $\mathbf{R}_j \in \mathbb{R}^{d \times d}$ is the representation of relation for j-th attribute field, and $\mathbf{q}_s$ is the item-level vector of query q. Similarly, $\hat{p}(a_j \mid t)$ measures the proximity between target t and j-th attribute's field. We use $\mathbf{o}_{j}^{t}$ to denote the attribute field vector in a_j, and the proximity can be calculated as:

$$\hat{p}(a_j \mid t) = \frac{exp(\mathbf{o}_{j}^{tT} \mathbf{R}_j \mathbf{t}_s)}{\sum_{j=1}^{n} exp(\mathbf{o}_{j}^{tT} \mathbf{R}_j \mathbf{t}_s)} \tag{5}$$

Then the proximity distribution of the target is constructed: $\hat{P}(A \mid t) = \left\{ \hat{p}(a_j \mid t) \mid j = 1, \ldots, n \right\}$. In the recommendation scenario, the relation between an item and an attribute should be considered as the importance of the attribute, not just the probability of the triple. So $\hat{P}(A \mid t)$ can be explained as the importance of each attribute field a_j to the candidate item t.

2.2.4. Attribute-level Representation Generation

Now, we are ready to introduce the mechanism of the above proximity distributions for item recommendation. Preference distribution and attribute importance distribution are features of users and items at attribute level respectively. Here, we apply non-linear transformation to map $P(A \mid q)$ and $\hat{P}(A \mid t)$ into the same space. The corresponding attribute-level interest representation $\mathbf{q}_A$ for query q can be obtained by:

$$\mathbf{q}_A = \mathrm{MLP}(P(A \mid q)) \tag{6}$$

where $\mathbf{q}_A \in \mathbb{R}^{d}$, $\mathrm{MLP}(\cdot)$ is a multi-layer perceptron consisting of hidden layers with *tanh* as the activation function. One point should be noted that we add *softmax* normalization layer to ensure a probabilistic input. Similarly, the attribute-level representation $\mathbf{t}_A$ for target t is computed as:

$$\mathbf{t}_A = \mathrm{MLP}(\hat{P}(A\,|\,t)) \tag{7}$$

where $\mathbf{t}_A \in \mathbb{R}^d$. In our model, we share the weight of MLP for both $\mathbf{q}_A$ and $\mathbf{t}_A$.

2.3. Joint Learning Method

To eliminate the domain difference between knowledge and recommendation, we put forward a hypothesis that when query q is related to target t, $P(A\,|\,q)$ and $\hat{P}(A\,|\,t)$ should be close. Based on the distance between the two distributions, we design a knowledge representation loss for recommendation and jointly optimize it with the click objective function (see Model Training in Section 2.4). In this way, our proposed proximity distribution can guide the entity embeddings learning of items and attributes through implicit feedback, which ensures that the learned knowledge representation is beneficial to recommendation.

First, we illustrate the proximity distributions distance constraint mentioned in Section 2.2. To impose restriction on the distance of two proximity distributions, a straightforward way is to minimize the following objective function:

$$d(P(A\,|\,q),\hat{P}(A\,|\,t)) \tag{8}$$

where $d(\cdot,\cdot)$ is the distance between two distributions. Furthermore, we choose to minimize the KL-divergence of two probability distributions. When query q is relevant to target t, $\omega_{qt}=1$, otherwise $\omega_{qt}=0$. Replacing $d(\cdot,\cdot)$ with KL-divergence and omitting some constants, we have:

$$\begin{aligned}\mathrm{L}_a &= \omega_{qt} d(P(A\,|\,q),\hat{P}(A\,|\,t)) \\ &= -\omega_{qt}\sum_{a_j\in A} p(a_j\,|\,q)\log(\frac{1}{p(a_j\,|\,q)}) + p(a_j\,|\,q)\log(\frac{1}{\hat{p}(a_j\,|\,t)}) \\ &\rightarrow -\sum_{a_j\in A}\omega_{qt}\, p(a_j\,|\,q) log(\hat{p}(a_j\,|\,t))\end{aligned} \tag{9}$$

In this loss function, we add additional constraints to make the related knowledge representation more similar than the irrelevant one. We use this joint learning method to bridge the gap between knowledge and recommendation, and we will demonstrate the effectiveness of this loss in section.

2.4. Model Training

Inspired by Word-entity duet [17] in KG-based information matching, we design an effective way to cross match query and target at item and attribute level, named as item-attribute duet. The query and target representation at both item-level and attribute-level are combined to capture more matching patterns as following:

$$rel(q,t;\mathrm{G}) = \mathbf{q}_s\cdot\mathbf{t}_s + \mathbf{q}_s\cdot\mathbf{t}_A + \mathbf{q}_A\cdot\mathbf{t}_s + \mathbf{q}_A\cdot\mathbf{t}_A \tag{10}$$

The duet utilizes a four-way interaction: item-level ($\mathbf{q}_s\cdot\mathbf{t}_s$), attribute-level ($\mathbf{q}_A\cdot\mathbf{t}_A$) and item attribute crossing level ($\mathbf{q}_s\cdot\mathbf{t}_A$ and $\mathbf{q}_A\cdot\mathbf{t}_s$). Then the clicking probability of candidate item is:

$$\hat{y}_{uv} = \sigma(rel(q,t;\mathrm{G})) \tag{11}$$

We use the pointwise approach to learn the parameters of our model. In particular, the cross-entropy loss is adopted as the objective function, which is defined as follows:

$$\mathrm{L}_c = \sum_{(u,v)\in Y} -(y_{uv} log\hat{y}_{uv} + (1-y_{uv}) log(1-\hat{y}_{uv})) \qquad (12)$$

We conduct $L2$ regularization on the trainable parameters θ, which is omitted here for simplicity, to avoid overfitting. We elaborate the implementation details in the Section *3*. The final loss function of our model is defined as:

$$\mathrm{L} = \mathrm{L}_c + \lambda_1 \mathrm{L}_a + \lambda_2 (\|E\|_2^2 + \|R\|_2^2) \qquad (13)$$

where L_a is elaborated in section, E is the embedding matrix for all entities, and R is the embedding matrix of relation. The settings of λ_1 and λ_2 are hyper-parameters which will be illustrated in the Section 3.3.

3. Experiments

3.1. Datasets

We conduct our experiments on movie and book datasets equipped with sub-KGs from Microsoft Satori, released by [4]. Table 1 shows statistics about these two datasets. Since the connections in the KGs are sparse, we conduct a field-based filtering process on the KGs, ensuring that only qualified attribute fields exist. For MovieLens-1M, six qualified attribute fields including actor, genre, director, writer, language and country are kept. As for Book-Crossing, we obtain five qualified attribute fields including genre, date of first publication, publisher, author and series. Then we transform the two datasets into implicit feedback. For MovieLens-1M, the threshold of rating is 4 and for book-crossing, no threshold is set due to its sparsity. And we sample an unwatched set for each user, which has the same size with the rated ones.

Table 1: Basic statistics of the two datasets.

	MovieLens-1M	Book-Crossing
user	6035	17860
item	2445	14967
interactions	753772	139746
triplets	1241995	151500
item-entities	142350	66056
attribute-entities	39661	11948
attribute-fields	12	25

In our experiments, we first randomly divide each dataset into train, evaluation, and test set, and their proportions are respectively 6:2:2. Then we conducted two parts of evaluation, including CTR prediction and Top-*K* recommendation.

3.2. Baseline Methods

We choose the following state-of-the-art baselines to compare with our proposed method:

Ripplenet [4] treats user's historical interactions as a seed set and propagates user interest along relations in the KG. Then it generates the user vector and item vector, and use inner product to calculate the click probability. KPRN [18] extracts paths between the user-item pair from KG and utilizes LSTM to generate path representations for recommendation. NCF [19] explores neural network architectures for collaborative filtering, which combines MF and deep neural networks to learn user-item interaction function. NFM [20] introduces neural network to model higher-order feature interactions and combines it with linear FM. CKE [6] incorporates KG and other information (i.e., image and text) into recommendation. DKN [5] treats entity and word embeddings as multiple channels and combines them together in CNN for CTR prediction. FMG [10] is a state-of-the-art meta-path based method, which predefines various types of meta-graphs and uses Factorization Machine on each meta-graph similarity matrix to make recommendation.

3.3. Parameter Settings

For movie recommendation, we set the learning rate to 0.01 and the coefficients λ_1 and λ_2 in Equation are set to 1 and 10^{-7} respectively. Other hyperparameters of our proposed model are set as follows: the batch size is 1024, the embedding size of relation and entity is 50. For book recommendation, we set the learning rate and batch size to 0.005 and 256 respectively. The coefficient λ_2 is set to10^{-5}. Other hyperparameters are the same as those set for movie dataset. The $\mathrm{MLP}(\cdot)$ in Equation and is implemented with one hidden layer, which is activated by $tanh$. We use default Xavier initializer to initialize HKRM's parameters. The loss function is optimized by Adaptive Moment Estimation (Adam) algorithm iteratively.

3.4. Comparison with the State-of-the-art Methods

We compare our proposed method (named HKRM) with other state-of-the-art algorithms on both CTR prediction (in Table 2) and top-*K* recommendation (in Figure 3 and Figure 4).

Table 2: CTR prediction performance on MovieLens-1M and Book-Crossing datasets.

Model	MovieLens-1M		Book-Crossing	
	AUC	ACC	AUC	ACC
Ripplenet	0.921	0.841	0.729	0.662
KPRN	0.876	0.776	0.717	0.678
NCF	0.910	0.834	0.725	0.675
CKE	0.769	0.702	0.676	0.615
NFM	0.773	0.694	0.659	0.620
DKN	0.837	0.670	0.683	0.598
FMG	0.782	0.722	0.701	0.616
HKRM	**0.936**	**0.865**	**0.740**	**0.683**

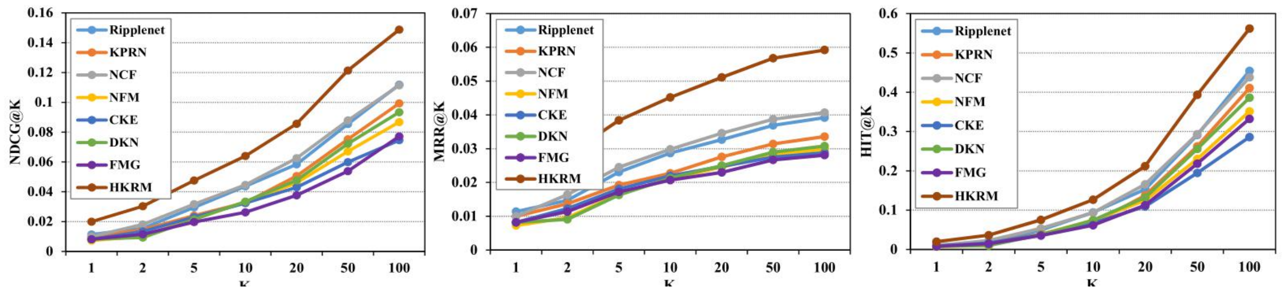


Figure 3: Top-K comparisons with other state-of-the-art method on MovieLens-1M dataset.

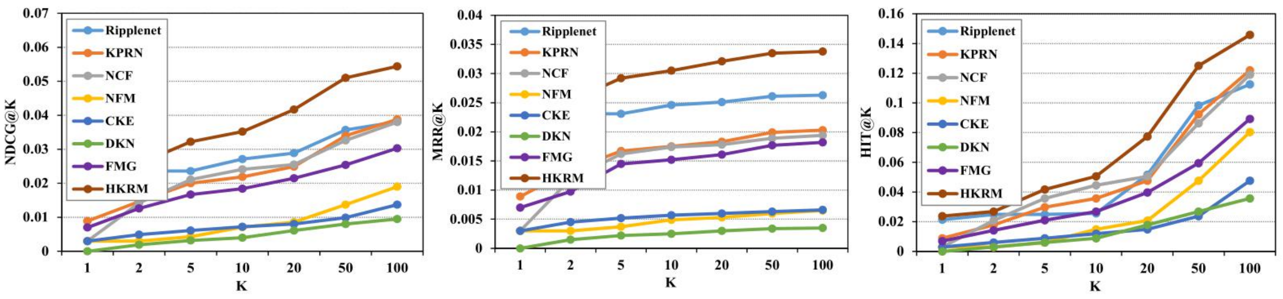


Figure 4: Top-K comparisons with other state-of-the-art method on Book-Crossing dataset.

From the results, we can see that HKRM shows a significant increase comparing with other baselines on both datasets. In HKRM, we model hierarchical knowledge, which are of great importance to capture fine-grained user preference, then an effective matching framework is proposed. Therefore, our model achieves the best performance.

Besides, we also have the following findings: Ripplenet achieves better performance than other baseline models, demonstrating that it can well integrates the structural information of KG into recommender system. However, its performance improvement is not obvious enough, especially comparing with NCF on the book dataset. This is probably because Ripplenet does not utilize hierarchical information in KG, which is proved to be effective by our model. KPRN performs worse than Ripplenet and NCF, which is probably because path-based methods rely heavily on the selection of paths. CKE gives poor performance in both datasets. It is because we only have structural knowledge available, without text and visual information. NFM achieves a comparable performance to CKE on both datasets, demonstrating that FM-based methods cannot utilize structure information well. DKN performs unsatisfactorily because movie and book names both contain only one entity, which is too short to be suitable for the CNN model. FMG also gives poor performance in both datasets. This demonstrates that meta-path based methods, which rely heavily on the predefined meta-path patterns, is not very good at mining user preferences.

3.5. Study of Our Proposed Model

In this section, we conduct a detailed analysis of our model to prove and explain the effectiveness of three components we proposed, including item-attribute hierarchy and joint learning method. Due to space limitations, we only report the AUC results of click prediction on both datasets.

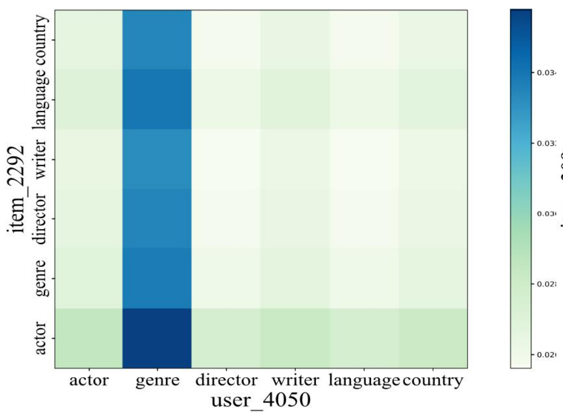
Table 3: Ablation Analysis (AUC) on MovieLens-1M and Book-Crossing datasets.

Model	MovieLens-1M	Book-Crossing
HKRM	0.9355	0.7394
HKRM-Attr	0.8838	0.7090
HKRM-Item	0.9242	0.7143
HKRM-woKL	0.9302	0.7262

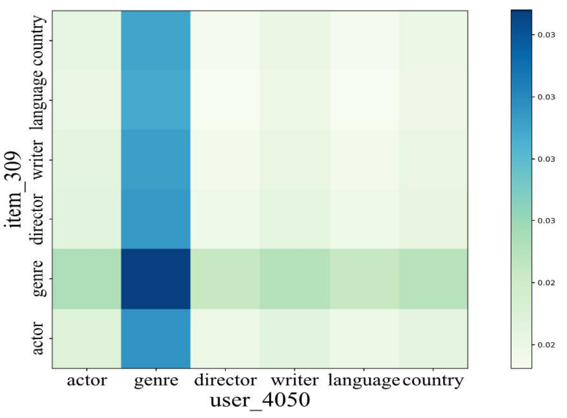
First of all, we prove that the item's attributes are helpful to the performance improvement of the recommender system. We use only item-level and attribute-level representation to conduct recommendation, respectively are HKRM-Item and HKRM-Attr in Table 3. The experimental results illustrate that only leveraging the attribute-level without direct user-item interaction can also obtain acceptable recommendation performance. This can explain that item's attribute can capture the user preference and supplement the connection information between items. The results also show that traditional methods based on user-item interaction are sensitive to data sparsity and leveraging the item-attribute hierarchical knowledge can greatly improve the accuracy.

Then, we investigate the joint learning method of knowledge representation and recommendation models. In HKRM-woKL, we remove the proximity distributions distance constrain (i.e., L_a), which is the bridge between knowledge representation and recommender system and measured by KL-Divergence. The comparison between HKRM and HKRM-woKL on both datasets clearly proves the effectiveness of the joint method we proposed. The main reason is that, without the proximity distributions distance constraint, the knowledge representation learning process only concentrates on modeling the connection between entities, regardless of the guidance of implicit feedback in recommender system. Thus, the learned embedding vectors that make up the item-level and attribute-level representations are not enough efficient to recommender system.

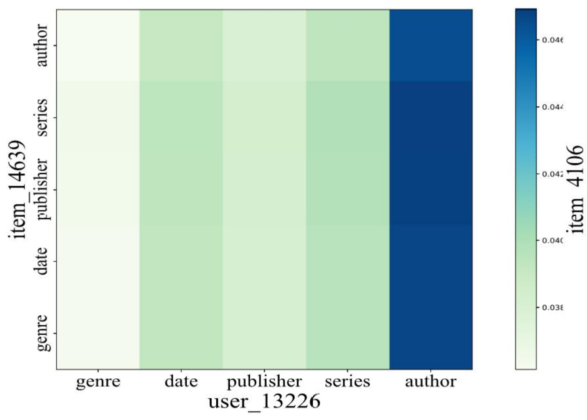
Finally, we conduct a case study to reveal the inner mechanism of hierarchical knowledge and distribution constraint on two datasets. Each recommendation scenario is associated with specific attributes, which also determine the boundaries of the KG. We first randomly choose a user with a clicked item (positive case) and an unclicked item (negative case) in the test set. Then, the corresponding user preference distribution and the item-attributes importance distribution are obtained from trained HKRM in Table 3. Figure 5 shows the heatmaps of attribute distribution correlation of user-item pairs. Note that the user-item pairs are selected from test set, so the distributions of them are not constrained by training based on our hypothesis.



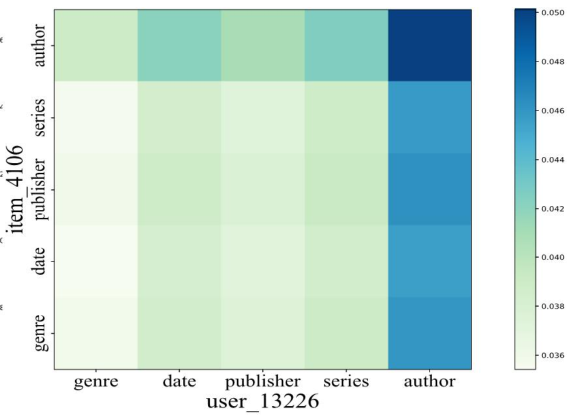
(a) A negative case in ML-1M



(b) A positive case in ML-1M



(c) A negative case in BC



(d) A positive case in BC

Figure 5: Visualization of user preference distribution and item-attributes importance distribution.

We consider two comparisons among the heatmaps:

(a) vs. (b): This comparison indicates that the user 4050 in dataset ML-1M prefers to select movies by genre, furthermore, the most important attribute of item 2292 and 309 is the actor and the genre respectively. Obviously, item 309 caters more to the user's taste than item 2292. Meanwhile, the predicting click probability of (a) and (b) is 0.01 and 0.97 respectively, which indicates that HKRM is confident in its prediction. This is a good illustration that the distributed constraint we designed can guide the knowledge representation learning, what's more, our model can accurately capture users preferences and important attributes of items.

(c) vs. (d): This comparison shows that user 13266 in dataset BC tends to choose books by author, meanwhile, item 4106 is most related to the attribute of the author. In addition, HKRM predicts a 90% probability that the user will click item 4106. As for the negative item 14639, HKRM does not highlight any important attributes of it and predicts a 29% chance of clicking. That's probably because it was clicked too few times by other users (0 time in the training set and 1 time in the test set).

4. Related Work

Several recent efforts on knowledge-graph-aware recommendation can be roughly classified into path-based methods and embedding-based methods.

Path-based methods [7][8][9][10] use meta-paths to describe the semantic relationship between users and items. Meta-paths explore the various patterns of paths from the user to the candidate item, which has many attributes, such as *user-movie-director-movie* and *user-book-author-book* in Heterogeneous Information Network (HIN). However, the performance of these methods relies too heavily on the predefined meta-paths, which require domain knowledge and are hard to cover all connectivity patterns. To address this issue, Wang [18] and Zhu [21] integrate all users into KG and use RNN-based model to represent some qualified paths from a user to the candidate item in KG. Despite performance improvements, these methods are not good at capturing user fine-grained preference because of the insufficient representations of item and attribute after path selection, which is not optimized for the recommendation objective.

Embedding-based methods use entity embeddings to represent user and item leveraging knowledge graph embedding (KGE) algorithms. For example, Ripplenet [4] imitates user preference propagating through the connection between triplets in KG, and represents user with the combination of all the tails with regularization of KGE, regardless of the distinction between item and attribute. DKN [5] treats entity embedding and word embedding, which are learned from KGE and word2vec respectively, as multiple channels and combines them together in CNN for news CTR prediction. CKE [6] simply generates the representation of users and items by combining its latent factor from MF and corresponding entity embeddings from TransR [14]. Huang [3] designs a model for sequential recommendation which regards items and attributes as a key-value structure, but it only uses the learned embeddings pretrained by transE [11], instead of modeling item-attributes hierarchy. These embedding-based-methods are less effective to represent user interest and eliminate domain difference.

Inspired by embedding-based methods, HKRM leverages the historical interaction records and the item connection information in KG to represent users and items. But note that we design a more suitable knowledge representation model for recommender system than these methods. The major difference between our work and the existing works is that we provide a new knowledge embedding way and consider the domain differences with the recommender system.

5. Conclusions

In this paper, we propose HKRM to model the item-attribute hierarchy in KG, which contains effective knowledge for recommendation, and combine the knowledge representation learning with recommender system based on our hypothesis. We utilize item-attribute duet way to catch more relevance pattern between users and candidate items, and design a query framework to explore user interest. Our method can capture fine-grained user preference and important attributes of items, which are very effective in improving the performance of recommendation. In addition, our method considers the domain difference between knowledge and recommendation, which is a key problem of KG-based recommendation model, and uses implicit feedback to guide the learning of knowledge representation. Finally, extensive experiments show that our model outperforms other state-of-the-art models on two real-word datasets.

References

[1] Qingyao Ai, Vahid Azizi, Chen Xu, and Yongfeng Zhang. Learning heterogeneous knowledge base embeddings for explainable recommendation. Algorithms, 11(9), 2018.

[2] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 1531–1540. ACM, 2018.

[3] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. Improving sequential recommendation with knowledge-enhanced memory networks. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pages 505–514, 2018.

[4] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pages 417–426, 2018.

[5] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. Dkn: Deep knowledge-aware network for news recommendation. In Proceedings of the 27th International Conference on World Wide Web, pages 1835–1844, 2018.

[6] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Weiying Ma. Collaborative knowledge base embedding for recommender systems. In Proceedings of the 22nd ACM SIGKDD international Confere 2016.

[7] Li Gao, Hong Yang, Jia Wu, Chuan Zhou, Weixue Lu, and Yue Hu. Recommendation with multi-source heterogeneous information. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, pages 3378–3384, 2018.

[8] Yizhou Sun and Jiawei Han. Mining heterogeneous information networks: a structural analysis approach. Acm SigKDD Explorations Newsletter, 14(2):20–28, 2013.

[9] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. Personalized entity recommendation: A heterogeneous information network approach. In Proceedings of the 7th ACM international conference on Web search and data mining, pages 283–292, 2014.

[10] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. Meta-graph based recommendation fusion over heterogeneous information networks. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 635–644, 2017.

[11] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Advances in neural information processing systems, pages 2787–2795, 2013.

[12] Guoliang Ji, Shizhu He, Liheng Xu, Liu Kang, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing, 2015.

[13] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu.Learning entity and relation embeddings for knowledge graph completion. In AAAI, volume 15, pages 2181–2187, 2015.

[14] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In AAAI, volume 14, pages 1112–1119, 2014.

[15] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. IEEE Transactions on Knowledge and Data Engineering, 29(12):2724–2743, 2017.

[16] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575, 2014.

[17] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. Word-entity duet representations for document ranking. In Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval, pages 763–772, 2017.

[18] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In Thirty-Third AAAI Conference on Artificial Intelligence, 2019.

[19] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In Proceedings of the 26th International Conference on World Wide Web, pages 173–182, 2017.

[20] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, pages 355–364, 2017.

[21] Zhu Sun, Jie Yang, Jie Zhang, Alessandro Bozzon, Long-Kai Huang, and Chi Xu. Recurrent knowledge graph embedding for effective recommendation. In Proceedings of the 12th ACM Conference on Recommender Systems, pages 297–305, 2018.