High Order Semantic Relations-Based Temporal Recommendation Model by Collaborative Knowledge Graph Learning

Yongwei Qiao[1(✉)], Leilei Sun[2], and Chunjing Xiao[2]

[1] Engineering Technology Training Center, Civil Aviation University of China, Tianjin, China
qiaoyongwei76@126.com
[2] School of Computer Science and Technology, Civil Aviation University of China, Tianjin, China

Abstract. Knowledge graph (KG) as the source of side information has been proven to be useful to alleviate the data sparsity and cold start. Existing methods usually exploit the semantic relations between entities by learning structural or semantic paths information. However, they ignore the difficulty of information fusion and network alignment when constructing knowledge graph from different domains, and do not take temporal context into account. To address the limitations of existing methods, we propose a novel High-order semantic Relations-based Temporal Recommendation (HRTR), which captures the joint effects of high-order semantic relations in Collaborative Knowledge Graph (CKG) and temporal context. Firstly, it automatically extracts different order connectivities to represent semantic relations between entities from CKG. Then, we define a joint learning model to capture high-quality representations of users, items, and their attributes by employing TransE and recurrent neural network, which captures not only structural information, but also sequence information by encoding semantic paths, and to take their representations as the users'/items' long-term static features. Next, we respectively employ LSTM and attention machine to capture the users' and items' short-term dynamic preferences. At last, the long-short term features are seamlessly fused into recommender system. We conduct extensive experiments on real-world datasets and the evaluation results show that HRTR achieves significant superiority over several state-of-the-art baselines.

Keywords: Collaborative knowledge graph · High-order semantic relation · Structural information · Temporal recommendation

1 Introduction

Collaborative Filtering (CF) is the most popular recommendation strategy, which exploits users' historical interactions to infer their preferences. However,

X. Wang et al. (Eds.): APWeb-WAIM 2020, LNCS 12317, pp. 337–351, 2020.
https://doi.org/10.1007/978-3-030-60259-8_25

they usually suffer from the data sparsity and cold start problem. Various types of side information have been incorporated to address it, such as social networks [7], temporal context [11] and user/item attributes [14]. Knowledge graph (KG) as the source of auxiliary data has been widely adopted to enhance recommendation. It connects various entities and links from different topic domains as nodes and edges to develop insights on recommendation.

Some state-of-art methods utilizing KG are proposed to boost recommendation quality. Meta-path based methods extract paths between two entities to represent different semantic relations, which leverages the relations of item-item [19], user-user [1,10,21], and user-item [6,9]. They can generate effective recommendation by modeling the user preference based on the semantic relations. Because the extracted meta-paths rely on manually designed features based on domain knowledge, they are always incomplete to represent all semantic relations. KG embedding based methods [14,16,20] automatically learn the embeddings of entities to capture entity semantics and incorporate them to recommendation framework. But A major limitation of these KG embedding methods is less intuitive and effective to represent the connection semantic relations of entities. For example, Zhang et al. [20] extracted items' semantic representations from structural content, textual content and visual content by capturing entity semantics via TransR, but ignored the high-order semantic relations between paired entities for recommendation. Then, some methods try to seek a way which not only can capture the semantic relations of entities and paths, but also not rely on handcrafted features and domain knowledge. Sun et al. [12] employed recurrent neural network (RNN) to learn semantic representations of both entities and high order paths to improve recommendation.

Almost all above methods rely on knowledge graph which includes various information from different domains. However, information fusion and network alignment are also very difficult. To address the limitations of constructing knowledge graph, a solution is to design a lightweight Collaborative Knowledge Graph (CKG) by only utilizing the facts in one domain as knowledge. CKG that often includes the interaction behaviors of users on items and side information for items (e.g., item attributes and external knowledge) and users (e.g., age, zip code, and occupation). Wang et al. [17] proposed Knowledge Graph Attention Network (KGAT), which explicitly models high-order connectivities in CKG and recursively propagates the embedding from a node's neighbors to refine its representation. But it only considered the high-order relations between users and items. Wang Hongwei et al. [15] proposed RippleNet which extends a user's potential preference along links in the CKG. These methods model user's preference by utilizing the high-order semantic representations and relations into recommender system, while they don't consider temporal influence. Xiao et al. [18] proposed KGTR which captures the joint effects of interactions by defining three categories relationships in CKG and considers the effect of temporal context. It can obtain the first and second order semantic relations by TransE and the embeddings of user's and item's various attributes, however, can not learn the high-order semantic relations.

Considering the limitations of existing solutions, we believe it is critical to develop a model that can effectively exploit high-order connections in CKG and take temporal information into account. To this end, we propose a novel High-order semantic Relations-based Temporal Recommendation (HRTR), which captures the joint effects of high-order semantic relations in CKG for recommendation. HRTR firstly mines semantic relations about some entities from different order connectivities. Then, it jointly learns high-quality representations of users, items, and their attributes to capture structural knowledge by employing TransE [2] and to explore sequence information by using recurrent neural network to encode semantic paths, which are regard as the users'/items' long-term static features. Next, by splitting the users' interactions with a time window, the users' short-term dynamic preferences are learned by LSTM [5]. The set of users who have recently interacted with an item is used to explore the items' short-term features by attention mechanism [13]. At last, the long-term and short-term preferences of users and items are integrated to recommend an item list to a user.

We summarize our main contributions as follows:

- We propose a joint learning model to capture high-quality representations of entities in a lightweight Collaborative Knowledge Graph, which not only can capture structural information, but also can explore sequence information by automatically encoding extracted semantic paths.
- We seamlessly fuse high-quality representations of entities and temporal context for recommendation, which effectively captures the users' and items' stable long-term and short-term dynamic preferences.
- We conduct experiments on real-world datasets, and the results show the significant superiority of HRTR over several state-of-the-art baselines.

2 Related Work

In this section, we review existing works on meta path based methods, KG embedding based methods, and semantic relation based methods, which are most related to our work.

2.1 Meta Path Based Methods

Meta path based methods capture the relations between two entities in KG by defining meta-paths, which are predefined by using handcrafted features based on domain knowledge. They generally infer a user preference by leveraging the different entity similarity of item-item [19], user-item [6,9], and user-user [1,10, 21]. HeteRec [19] learned the user preference on an item connected with his/her rated item via different meta paths. SemRec [10] captured semantic similarity among users by introducing the weighted meta path. Wang et al. [1] and Zheng et al. [21] respectively proposed matrix factorization model to regularize user similarity derived from meta path. SimMF [9] extended matrix factorization

based model by adding meta path based user-item similarity. They successfully model the user preference based on the semantic relations, but they heavily rely on manually designed features based on domain knowledge and can not completely represent all semantic relations between two entities.

2.2 KG Embedding Based Methods

KG embedding based methods first capture the entity embedding by exploiting the structural information of KG and incorporate the learned entity embeddings into a recommendation framework. CKE proposed by Zhang et al. [20] combined CF with item embeddings obtained via TransR [8]. DKN [16] combined the treated entity embeddings with CNN for news recommendation. SHINE [14] embed three types of networks by designing deep autoencoders for celebrity recommendations. But a major limitation of these KG embedding methods is less intuitive and effective to represent the semantic relations of entities.

2.3 Semantic Relation Based Methods

Another kind of methods effectively improves the performance of recommendation by mining the high-order semantic relations or integrating various other information and strategies to capture better representations for recommendation. Sun et al. [12] employed RNN to model different order semantics of paths to characterize user preferences. Wang et al. [17] proposed knowledge graph attention network (KGAT), which recursively propagates the embedding from a node's neighbors to refine its representation and discriminates the importance of neighbors by using an attention mechanism. Wang Hongwei et al. [15] proposed RippleNet which extends a user's potential preference along links in CKG. These methods model users' preferences by utilizing the high-order semantic representations and relations, while they do not consider temporal influence. Xiao et al. [18] proposed KGTR which captures the joint effects of interactions by defining three categories relationships and temporal context.

Different from these works, our proposed method not only can effectively exploit semantics of entities and high-order connectivities, but also take the long-short term preferences of users and items into account.

3 Our Proposed Model

Let $U = \{u_1, u_2, \cdots\}$ and $V = \{v_1, v_2, \cdots\}$ denote the sets of users and items, respectively. $M = \{M_{uv} | u \in U, v \in V\}$ is a sparse user-item interaction matrix that consists of users, items, and the interactions which include rating, browsing, clicking and so on. Meanwhile, there are various attributes of users and items, such as gender, age, occupation, which are significant auxiliary information for recommendation result. We aim to build temporal personalized recommendation model for a user based on the semantic embeddings of users, items and their attributes, and then recommend items to users.

The overview of our proposed HRTR is shown as Fig. 1, which consists of three parts: (1) learning high quality semantic representations of users, items and their attributes by TransE and RNN; (2) training long-short term preferences of users and items, in which the learned semantic representations are considered as the long-term features, the short-term features of users and items are captured by LSTM and attention machine based on the learned semantic embeddings and interactions, repectively; (3) predicting how likely a user interacts an item by integrating these learned long-short term preferences into a sigmoid based prediction model.

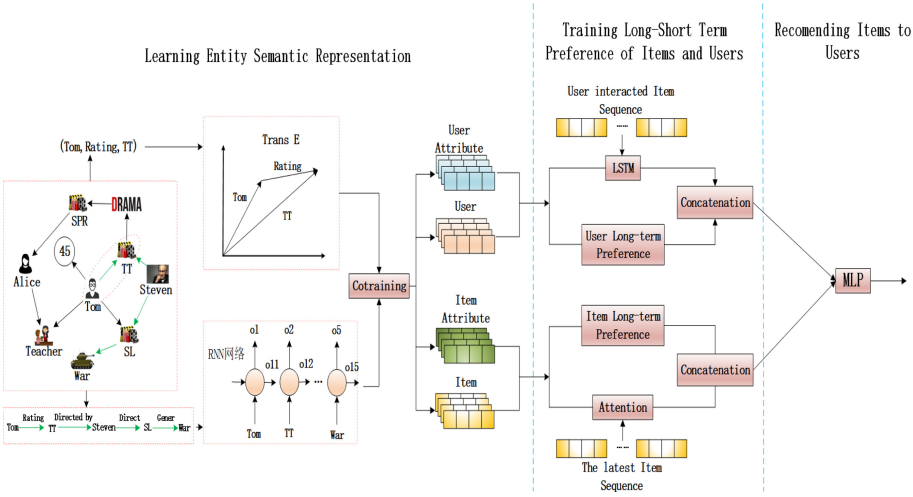

Fig. 1. The framework of high order semantic relations temporal recommendation

3.1 Different Order Semantic Relations Mining

Designing Collaborative Knowledge Graph. Given U, V, M as well as users'/items' attributes, user-item interaction graph and user/item attribute graph are defined, which is regarded as the formal construction of Collaborative Knowledge Graph (CKG).

As illustrated in Fig. 2, taking movie data as an example, the users and items are treated as entities. When there is an observed interaction between user u and item i (e.g., purchases, clicks, ratings), a link will be constructed between them. Here, user-item interaction graph G_1 is denoted as $G_1 = \{(u, m_{uv}, i)|u \in U, i \in V, m_{uv} \in R'\}$, and R' is the interaction sets. In addition to the interactions, users/items have different types of side information to profile them. The user/item attribute graph G_2 is defined to organize the side information in the form of directed graph. Formally, it is presented as $G_2 = \{(h', r', t')|h' \in$

$U \cup V, t' \in \Psi, r' \in \Omega\}$, where Ψ is the attribute values set, Ω is the attribute set and contain canonical relations and their inverse direction. (h', r', t') describes that there is a semantic relationship r' from h' to t'. For example, $(Tom, age, 45)$ states the fact that Toms age is 45. Then, Collaborative Knowledge Graph which encodes user interactions and the side information of users and items is defined as a unified graph $G = \{(h, r, t), h, t \in \varepsilon, r \in R\}$, where $\varepsilon = U \cup V \cup \Psi$, $R = R' \cup \Omega$.

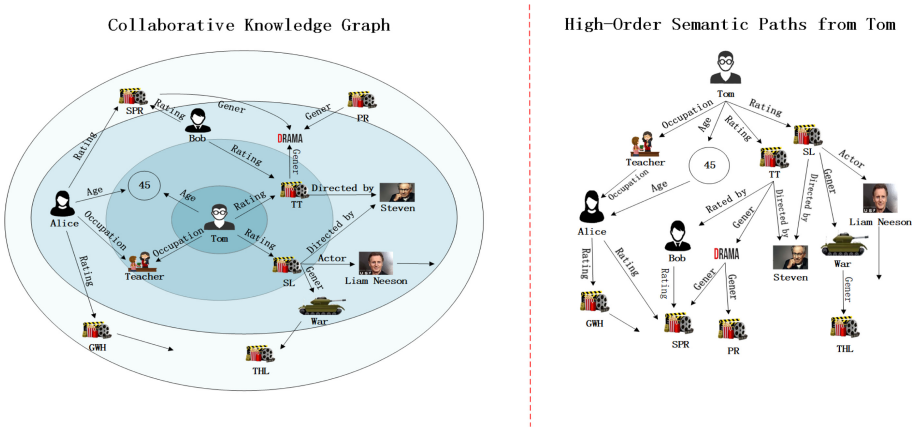

Fig. 2. Different order semantic relations mining on CKG

Different Order Semantic Relations Mining. The key to successful recommendation is to fully exploit the high-order relations in CKG, which represents the way to learn the embedding of entities by using the first-order, second-order or even higher-order semantic relations, respectively. Formally, we define the L-order relations between nodes as a multi-hop relation path: $e_0 \xrightarrow{r_1} e_1 \xrightarrow{r_2} \cdots \xrightarrow{r_L} e_L$, where $e_l \in \varepsilon$ and $r_l \in R$, (e_{l-1}, r_l, e_l) is the $l-th$ triplet, and L is the length of relation path. Then we can denote the semantic relation paths that reach any node from e_0 with different length l. As shown in Fig. 2, for an entity "Tom", we can exploit the first-order semantic relations, $Tom \xrightarrow{age} 45$, $Tom \xrightarrow{occupation} teacher$, $Tom \xrightarrow{rating} TheTerminal(TT)$ and $Tom \xrightarrow{rating} Schindler'sList(SL)$, which represents the attributes of Tom, and his rating activities, respectively. They can be easily extended to the second-order semantic relations, which contains more richer semantics. For example, $Tom \xrightarrow{age} 45 \xrightarrow{-age} Alice$, $Tom \xrightarrow{occupation} teacher \xrightarrow{-occupation} Alice$, $Tom \xrightarrow{rating} TT \xrightarrow{-rating} Bob$, $Tom \xrightarrow{rating} SL \xrightarrow{directedby} Steven$, which indicates semantic relations between Tom and Alice, Bob, Steven relying on common attributes, rating on one item TT, or the relationship on SL. However, to exploit such high-order relations, there are challenges: 1) the number of semantic paths

increases dramatically with the order size, which will lead to more computation in training it, and 2) the different order relations are of different importance to recommendation, which requires the model to carefully define them.

Generally, shorter semantic paths indicate stronger relations, while longer ones may represent more semantic relations. To increase model efficiency, We only consider the semantic paths with the length less than a threshold and take the semantic relations started from an entity of user or item into account.

3.2 Semantic Relation Learning

We aim to parameterize entities and relations as vector representations to improve recommendation, which not only learns the structural information, but also the sequence information of semantic relations. Here we employ TransE [2], a widely used method, on CKG to capture this structural knowledge. Sequence information of semantic paths is exploited by adopting RNN.

Structural Embedding. To capture this structural information, TransE is used to learn it by optimizing the probability $P(h, r, t)$ of the relational triples (h, r, t), which exists in the graph. So the probability $P(h, r, t)$ is formalized as follows:

$$L_{SE} = P(h, r, t) = \sum_{(h,r,t^+) \in CKG} \sum_{(h,r,t^-) \in CKG^-} \sigma(g(h, r, t^+) - g(h, r, t^-)) \quad (1)$$

where $\sigma(x) = 1/(1 + exp(x))$ is sigmoid function. The CKG and the CKG^- are the positive and negative instances set, respectively. $g(\cdot)$ is the energy function which represents the correlation from h to t in the relation r. The score of $g(\cdot)$ is lower if the triplet is more likely to be true. Here, we define $g(h, r, t)$ as follow:

$$g(h, r, t) = ||e_h + e_r - e_t||_{L_1/L_2} + b_1 \quad (2)$$

where e_h, e_r, e_t are the embedding of h,r and t; b_1 is a bias constant. The relations of entities are modeled through the triples, which can inject the direct connections into embedding to increase the model representation ability.

Sequence Embedding. Structural embedding can capture entity semantics and semantic relations between entities, however, can not study the semantic relations of high-order paths. By regarding the entities in different high-order semantic paths as a sequence, we naturally think that recurrent neural networks are suitable for modeling different order semantic paths. This is mainly because that it has capability in modeling sequences with various lengths. To this end, we adopt RNN to learn the semantics of entities by encoding the semantic paths with different lengths, and then a pooling operation is used to get the final semantic representation.

Assume n paths of different lengths from an user u_i to any another entity e_j, i.e., $p_l = e_0 \xrightarrow{r_1} e_1 \xrightarrow{r_2} \cdots\cdots \xrightarrow{r_T} e_T$ with $e_0 = u_i$, the RNN learns a representation

h_{lt} for each entity e_t in p_l, which considers both the embeddings of entities in the path and the order of these entities. It encodes the sequence from the beginning entity of the path e_0 to the subsequent entity e_t. For entity e_t

$$O_{lt} = \delta(W \cdot O_{l(t-1)} + H \cdot h_{lt} + b_2) \tag{3}$$

where W is the linear transformation parameters for the previous step, H is for current step; b_2 is the bias term; δ is the sigmoid function. $O_{l(t-1)}$ is a learned hide state by encoding the subsequence from e_0 to e_{t-1}, O_{lt} is a learned hide state after learning the embedding of h_{lt} at step t. For n paths from a user entity u_i, their last representations are $O_{1T_1}, O_{2T_2} \cdots O_{nT_n}$, where T_n is the length of p_n. Based on this, we get the entity representation $O[u_i]$ by adding a max pooling or an average pooling operation towards all the n paths. Similarly, we can get the representation $O[v_j]$ of item v_j. So the objective function can be defined as:

$$L_{SP} = \sum_{(u_i, v_j) \in CKG^+} -\ln \delta(\hat{y}(u_i, v_j) - y(u_i, v_j)) \tag{4}$$

where the probability $\hat{y}(u_i, v_j) = \delta(O[u_i]^T O[v_j])$ is predicted by conducting inner product of user and item representations, CKG^+ is positive instances set, $\sigma(\cdot)$ is the sigmoid function.

Finally, we have the objective function to jointly learn Eqs. (1) and (4), as follows:

$$L = L_{SE} + L_{SP} \tag{5}$$

We optimize L_{SE} and L_{SP} alternatively. Specifically, all representations for nodes are updated by randomly sampling a batch of instances h, r, t, t'; hereafter, we randomly sample some users or items and mine semantic paths starting from them, and update the representation for all nodes. Then we can get the embeddings of users, items and their attributes U_L, V_L, U_a, V_a, which are regard as the long term preferences of users and items for temporal recommendation.

3.3 Training Long-Short Term Preference of Users and Items

Users Long-Short Preference. A user preference is compose of the long-term and short-term preference. The long-term preference indicates the stable interest, which is represented by semantic presentations of the user's interacted items and their attributes. The short-term preference indicates a user's dynamic interest, which are learned by LSTM here. The size of time window t is the key issue when modeling the user dynamic preference. The more fine-grained interest changes can be captured by using smaller time window, but the training data is very sparse and the learning process is difficult. On the contrary, the larger time window will has sufficient training data, while the model is less adaptive for capturing dynamics changes of a user preference. To this end, we adopt the latest n items to model the user short term preference, which ensures the enough training data to train the user preference. Instead of inputting the user interacted history in form of items sequence into LSTM, the learned semantic

representations of the interacted items and their attributes are regarded as pretrain input of LSTM. This makes the training faster and more effective. Finally, the output of LSTM U_S is taken as the user short-term preference.

Items Long-Term Preference. Similar to the user preference, the item preferences are also made up of two parts. The learned semantic representations of items and their attributes are regarded as their long-term preferences. Their short-term features are determined by the popularity of them changing over time. We think that the most fashionable items currently have a greater contribution to user preference. Here, we adopt attention machine to capture the short-term characteristics of items because of its capability of keeping the contextual sequential information and exploiting the relationships between items. At last, the items recently viewed by all users are used as attention input. Similar to [13], the attention vector for items $(1, 2, \cdots I)$ are calculated by using Eq. (6) at each output time t.

$$V_s' = \sum(\delta(z^T tanh(W_c c_t + W_y y_i))y_i) \tag{6}$$

where z , W_c, W_y are learnable parameters, c_t is the training item at time t and y_i is i-th item in input sequence. $\delta(\cdot)$ is a sigmoid function. Lastly, c_t and V_s' are concatenated as the next input c_{t+1}. The final output V_s can be regarded as items' dynamic preferences.

3.4 Recommending Items to Users

Our task is to predict items which the user likely to prefer to when giving the long-short term preferences of users, items and their attributes. They can be concatenated into a single vector as the input of a standard multi-layer perceptron (MLP), as follow:

$$\begin{aligned} U_P &= U_L \| U_a \| U_S \\ V_P &= V_L \| V_a \| V_S \end{aligned} \tag{7}$$

where $\|$ is the concatenation operation. $\hat{y}_{uv}$ is used to represent the probability of the user u interact with the item v. It is represented by Eq. (8)

$$\hat{y}_{uv} = \sigma(h^T O_L) \tag{8}$$

where O_L is output of MLP. For any $l - th$ layer, O_l is defined as Eq. (9)

$$\emptyset_l = \phi_l(\lambda_l O_{l-1} + \vartheta_l) \tag{9}$$

where ϕ_l, λ_l, ϑ_l are the ReLU activation function, weight matrix and bias vector for the l-th layer's perceptron, respectively. O_{l-1} is the $l-1$-th layer's output of MLP. U_P and V_P are the input of input layer.

We treat y_{uv} as label, which represents the actual interaction. 1 means user u has interacted with item v, and 0 otherwise. Therefore, the likelihood function is defined as Eq. (10):

$$p(y, y^{-}|\Theta_f) = \prod_{(u,v)\in y} \hat{y}_{uv} \prod_{(u,v)\in y^{-}} (1 - \hat{y}_{uv}) \tag{10}$$

Taking the negative logarithm of the likelihood, we gain the objective function as Eq. (11):

$$\begin{aligned} L &= -\sum_{(u,v)\in y} \log \hat{y}_{uv} - \sum_{(u,v)\in y^{-}} \log(1 - \hat{y}_{uv}) \\ &= -\sum_{(u,v)\in y\cup y^{-}} y_{uv} \log \hat{y}_{uv} + (1 - y_{uv}) \log(1 - \hat{y}_{uv}) \end{aligned} \tag{11}$$

where $y^{-}$is the negative instances set, which is uniformly sampled from unobserved interactions with the sampling ratio related to the number of observed interactions. The output of each neuron is controlled in [0,1] by using sigmoid function. The learning will stop when their output is near to either 0 or 1.

We adopt adaptive gradient algorithm to optimize our model, which automatically adapts the step size to reduce the efforts in learning rate tuning. In the recommendation stage, candidate items are ranked in ascending order according to the prediction result, and we recommend the top ranked items to users.

4 Experiments

In this section, we perform experiments to evaluate HRTR. We first introduce experimental setup, including the datasets, baselines, evaluation metrics and parameter settings, and then present the experiment results against the related baselines.

4.1 Experimental Setup

Dataset Description. To demonstrate the effectiveness of HRTR, We conduct experiments on two public datasets. The one is MovieLens-1M[1] which consists of 6,040 users, 3,952 items and approximately 1M explicit ratings. Besides the user-item ratings, it also includes some auxiliary information about users and items, such as age, occupation, zip code, genre, title, director, etc. Ratings ranging from 1 to 5 are transformed into either 1 or 0, where 1 indicates a user have rated an item, otherwise 0. Another one is Yelp[2], which contains 4700000 review information, 156000 businesses and 110000 users. Here we consider businesses, for example movie theaters, as items. We set the threshold to 10, which represents that a user has at least 10 interactions.

For each user, his/her interactions are first sorted based on interactive time, and the latest one is regarded as the test positive instance and others are utilized as positive instances for training. Finally we randomly sample four negative instances for per positive one, and randomly sample 99 unrated items as the test negative instances.

[1] https://grouplens.org/datasets/movielens/.
[2] https://www.yelp.com/dataset/challenge.

Evaluation Metrics. Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) are used to evaluate the performance of a ranked list [4]. The HR intuitively measures whether the recommendation list includes the test item. The NDCG measures the ranking of the test item in top-K list. We calculate HR and NDCG for each test user and take the average score as the final results.

Baselines. To validate the effectiveness of our proposed HRTR, we compare it with the following state-of-the-art baselines

- NCF [3]: It uses a multi-layer perceptron replacing the inner product to learn the user-item interactions.
- MCR [6]: It is a meta path based model, which extracts qualified meta paths as similarity between a user and an item.
- CKE [20]: It is a collaborative KG embedding based method, which learns item latent representations by combining structural, textual and visual information in a unified framework.
- KGTR [18]: It is a semantic relation plus temporal method, which defines three relationships in CKG to express interactions for recommendation.

Parameter Settings. For structural embedding training, the embedding size is fixed to 100, hyper parameter b_1 is set to 7, and L_1 is taken as distance metric. For sequence embedding training, the threshold of the longest semantic path length is set to 6. A longer path hardly improves performance but brings heavier computational overhead. We implement HRTR in Python based on the Keras framework and employ mini-batch Adam to optimize it. For MovieLens dataset, 16 items are selected as the input of LSTM for one user to learn his/her short term preference. For Yelp, 8 items are selected to mine users' preference. We select items which interacted by all users in the latest hour as input of attention to learn the items' short term features.

We find out other optimal parameters for HRTR by experiment and take HR and NDCG as metrics to evaluate them. We apply a grid search to find out the best values for hyperparameters: the dimension of representation vector d is tuned in {50, 100, 150, 200}, the batch size s is searched in {128, 256, 512, 1024, 2048}. Due to space limitation and the same trend, only the results on MovieLens are shown in Fig. 3. From Fig. 3 we can see that HR@10 and NDCG@10 firstly increase and then decrease with the increase of d. The performance of HRTR is best when $d = 100$. As s increases, its performance increases rapidly and tends to be stable with different batch size and the best performance is obtained when bach size is set to 2048. So we set $s = 2048$ and $d = 100$ as the optimal parameters for MovieLens, while for Yelp $s = 2048$ and $d = 150$. The optimal parameters of baselines are set to their recommended values.

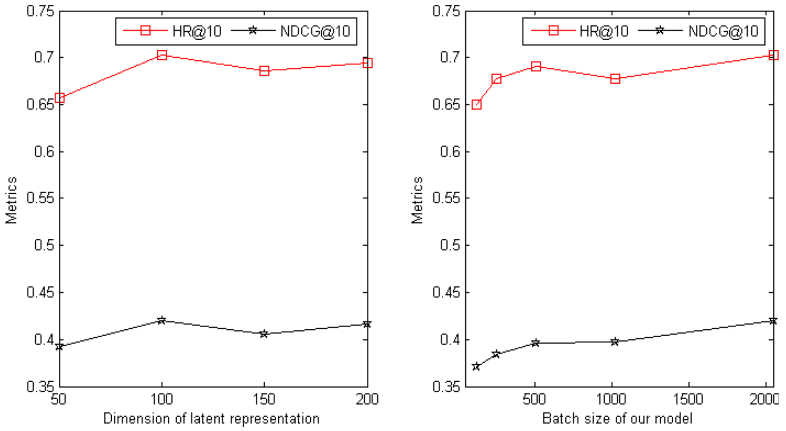

Fig. 3. The performance of our HRTR with different dimensions and batch sizes

4.2 Results and Analysis

Results Analysis of HRTR. Here, we report the performance of our HRTR for top@k recommendation on MovieLens, where k is tuned in {5, 10, 15, 20, 25, 30}. Firstly, the batch size is set to 2048 and the dimension is tuned in {50, 100, 150, 200}, the results are shown in Fig. 4. Some interesting observations can be noted from Fig. 4. With increasing of k, HR@k and NDCG@k are improved rapidly and tend to be stable. In general, HR@k and NDCG@k get better results when $d = 100$, while the difference is very slight. The result is consistent with the analysis in parameter settings. That shows it is not sensitive to vector dimension.

As shown in Fig. 5, we also tested the top@k item recommendations, when vector dimension is fixed to 100 while batch size is searched in {512, 1024, 2048}. We can observe that HR@k and NDCG@k increase when k varies from 5 to 30. HR@k and NDCG@k all get the better performance when batch size becomes larger and it is obvious for NDCG@k. Due to the same trends, the results on Yelp are not described in detail.

Comparing HRTR with Baselines. Table 1 summarizes the performance of all methods on two datasets and the best performance is boldfaced. MCR without considering temporal information and CKE without employing semantic paths achieve poor performance compared with other methods. This confirm that semantic path and temporal context are useful to provide better recommendation. MCR highly outperforms all models if the ranking list is short. That is mainly because MCR exploits the entity relation in the KG by introducing meta paths relying on domain knowledge, which has its superiority when requiring shorter recommendation list. The performance of KGTR is closest to HRTR.

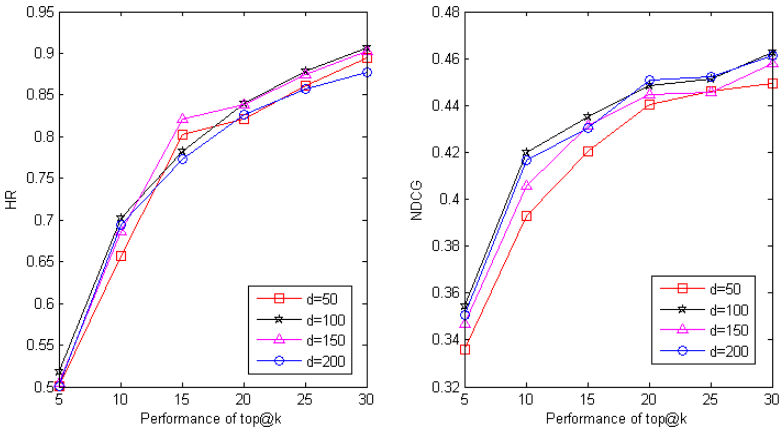

Fig. 4. HR@K and NDCG@K results with different dimensions

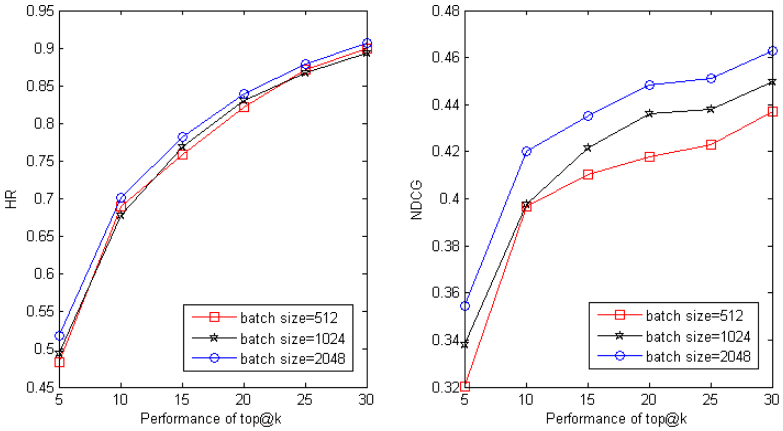

Fig. 5. HR@K and NDCG@K results with different batch size

The reason is that they all consider semantic relation with temporal context. While HRTR is still superior to KGTR. The mainly reason is that HRTR can capture high order semantic relation, KGTR can not do it. For sparser Yelp dataset, HRTR achieves significantly better performance than other methods, that shows that HRTR is more suitable for sparser data.

Table 1. Performance of all comparison methods across all the evaluation metrics.

Datesets	Methods	HR@k			NDCG@k		
		5	10	20	5	10	20
MovieLens-1M	NCF	0.5339	0.6812	0.8204	0.3501	0.4102	0.4403
	MCR	**0.5764**	0.5873	0.6079	**0.4354**	0.4001	0.3809
	CKE	0.4251	0.5736	0.7345	0.1585	0.1698	0.1999
	KGTR	0.5402	0.6978	0.8303	0.3603	**0.4232**	0.4501
	HRTR	0.5102	**0.6989**	**0.8312**	0.3412	0.4120	**0.4508**
Yelp	NCF	0.1123	0.1360	0.2011	0.1650	0.1123	0.0543
	MCR	**0.1202**	0.1143	0.1156	**0.1815**	0.1828	0.1798
	CKE	0.0931	0.1098	0.1172	0.1212	0.1131	0.0821
	KGTR	0.1102	0.1179	0.1206	0.1528	0.1698	0.1658
	HRTR	0.1090	**0.1379**	**0.2152**	0.1811	**0.1903**	**0.1822**

5 Conclusions

In this paper, we proposed a High-order semantic Relations-based Temporal Recommendation model (HRTR) that explores the joint effects of different semantic relations in CKG and temporal context. HRTR overcame the limitations of existing KG-aware methods by jointly learning different order semantic relations between entities, which not only captures structural information, but also explores sequence information in CKG. HRTR respectively exploited the users' and items' long and short term features, which could capture their stable and temporal dynamic preferences. Extensive experiments were conducted on real datasets and the experimental results demonstrated the significant superiority of HRTR over state-of-the-art baselines. In future, we plan to design an effectively unified model to simultaneously explore the structural and sequence information to improve the performance.

Acknowledgement. This work was partially supported by grants from the National Natural Science Foundation of China (No. U1533104; U1933114), the Fundamental Research Funds for the Central Universities (No. ZXH2012P009) and Civil Aviation Science and Technology Project (No. MHRD20130220)

References

1. Wang, Y., Xia, Y., Tang, S., Wu, F., Zhuang, Y.: Flickr group recommendation with auxiliary information in heterogeneous information networks. Multimedia Syst. **23**(6), 703–712 (2016). https://doi.org/10.1007/s00530-015-0502-5
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems, pp. 2787–2795 (2013)
3. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of WWW, pp. 173–182. WWW (2017)

4. He, X., Tao, C., Kan, M.Y., Xiao, C.: Trirank: review-aware explainable recommendation by modeling aspects (2015)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
6. Hu, B., Shi, C., Zhao, W.X., Yu, P.S.: 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, pp. 1531–1540 (2018)
7. Jamali, M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the 2010 ACM Conference on Recommender Systems RecSys 2010, Barcelona, Spain, 26–30 September 2010 (2010)
8. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of AAAI (2015)
9. Shi, C., Liu, J., Zhuang, F., Yu, P.S., Wu, B.: Integrating heterogeneous information via flexible regularization framework for recommendation. Knowl. Inf. Syst. **49**(3), 835–859 (2016). https://doi.org/10.1007/s10115-016-0925-0
10. Shi, C., Zhang, Z., Luo, P., Yu, P.S., Yue, Y., Wu, B.: Semantic path based personalized recommendation on weighted heterogeneous information networks. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (2015)
11. Sun, Y., Yuan, N.J., Xie, X., Mcdonald, K., Zhang, R.: Collaborative intent prediction with real-time contextual data. **35**(4), 30 (2017)
12. Sun, Z., Yang, J., Zhang, J., Bozzon, A., Huang, L.K., Xu, C.: Recurrent knowledge graph embedding for effective recommendation. In: Proceedings of the 12th ACM Conference on Recommender Systems, pp. 297–305. ACM (2018)
13. Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., Hinton, G.: Grammar as a foreign language. Eprint Arxiv, pp. 2773–2781 (2015)
14. Wang, H., Zhang, F., Hou, M., Xie, X., Guo, M., Liu, Q.: Shine: signed heterogeneous information network embedding for sentiment link prediction (2018)
15. Wang, H., et al.: Ripplenet: propagating user preferences on the knowledge graph for recommender systems. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 417–426 (2018)
16. Wang, H., Zhang, F., Xie, X., Guo, M.: DKN: deep knowledge-aware network for news recommendation (2018)
17. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: KGAT: knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 950–958 (2019)
18. Xiao, C., Xie, C., Cao, S., Zhang, Y., Fan, W., Heng, H.: A better understanding of the interaction between users and items by knowledge graph learning for temporal recommendation. In: Nayak, A.C., Sharma, A. (eds.) PRICAI 2019. LNCS (LNAI), vol. 11670, pp. 135–147. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29908-8_11
19. Xiao, Y., Xiang, R., Sun, Y., Sturt, B., Han, J.: Recommendation in heterogeneous information networks with implicit user feedback. In: Proceedings of the 7th ACM Conference on Recommender Systems (2013)
20. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.Y.: Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD, pp. 353–362. ACM (2016)
21. Zheng, J., Liu, J., Shi, C., Zhuang, F., Li, J., Wu, B.: Recommendation in heterogeneous information network via dual similarity regularization. Int. J. Data Sci. Analytics **3**(1), 35–48 (2016). https://doi.org/10.1007/s41060-016-0031-0