# A Better Understanding of the Interaction Between Users and Items by Knowledge Graph Learning for Temporal Recommendation

Chunjing Xiao(✉), Cong Xie, Shuyan Cao, Yuxiang Zhang, Wei Fan, and Hongjun Heng

School of Computer Science and Technology, Civil Aviation University of China, Tianjin, China
chunjingxiao@163.com

**Abstract.** Recently the knowledge graph (KG) as extra auxiliary information is widely used to improve recommendation. Existing methods usually treat knowledge representation as characteristic information for addressing data sparsity and cold start issues. However, they ignore the implicit and explicit interaction between users and items, which may be gained by the relation extraction and knowledge reasoning, to lead to suboptimal performance. Thus, we believe that it is crucial to incorporate both relations and attributes of users and items into recommender system. That can better capture the extent that a user prefer to an item. In this paper, we propose a novel knowledge graph-based temporal recommendation (KGTR) model. Firstly, we design a lightweight KG on the basis of a single independent domains knowledge without extra supplement. We define three relationships to express interactions within/between users and items, including the interaction of a user browsing an item, the social relation of two users browsing one item, and the behavior of a user browsing items in the meantime. Different from previous knowledge translation-based recommendation methods, we embed interactions by adding them to the transformation from one entity to another in KG. Extensive experiments on real world dataset show that our KGTR outperforms several state-of-the-art recommendation methods.

**Keywords:** Knowledge graph · Implicit interaction · Explicit interaction · Temporal recommendation

## 1 Introduction

The various facts from different domains interlink with each other and store in a complex heterogeneous graph called knowledge Graph (KG). The entities, such as people, books, musics, movies, are treated as nodes in KG and the relations between entities are denoted as edges. Owing to the connection of

© Springer Nature Switzerland AG 2019
A. C. Nayak and A. Sharma (Eds.): PRICAI 2019, LNAI 11670, pp. 135–147, 2019.
https://doi.org/10.1007/978-3-030-29908-8_11

various information from different topic domains in KG, knowledge exploration can develop insights on problems. That are difficult to determine on a single domain data. Over the past years, KG has been widely adopted in many fields, including dialogue system, Web search, and recommendation system.

The rich information of KG has recently shown great potential to enhance accuracy and explainability of recommendation [17]. For example, Zhang et al. [19] extract items' semantic representations from structural content, textual content and visual content by considering the heterogeneity of both nodes and relationships in KG. Sun et al. [15] employ recurrent networks learning semantic representations of both entities and paths for characterizing user preferences to improve recommendation. In these cases, the semantic representations of user and item or user's preference have accurately been obtained. However, the potential of the KG may still fail to be exploited since they suffer from the following limitations: (1) relying on a large-scale knowledge graph and extra knowledge base to extract features by heavy feature engineering process. (2) only utilizing the semantic representations into recommender system while ignoring the implicit and explicit interaction between users and items. For instance, two users are likely to have interaction when they both connect one item.

To address the above issues, we propose a novel knowledge graph-based temporal recommendation (KGTR) model, which captures the joint effects of users and items interactions information. We design a lightweight KG by only utilizing the facts in one domain as the knowledge, meanwhile, extra auxiliary data is lack. Three categories relationships are defined to exploit user-item interaction, including user relationship, item relationship and rating activity. User social relationship implies that two different users browse one item simultaneously, and can be called as user relationship. Item relationship means that one user browses various items. They are considered as implicit interaction in recommendation. Rating activity expresses that a user has rated the item, which is regard as explicit interaction in recommendation. Then representations of users' and items' static feature are obtained by TransE [1] in the light of three kind of relationships separately. Meanwhile, embeddings of users' and items' various attributes are learned by the KR-EAR [11] on the basis of former static representations, which serve as explicit information of user and item.

Considering the important effect of temporal context, we hold that prevailing items at the previous moment, similar to users' preference changing, have affected in recommendation result. Therefore, different from traditional temporary recommendation, we aggregate long-term and short-term features of users and items in recommender process. The attributes features and static feature learned by above procedure constitute the user's long-term features. The item's long-term features are similar. The user's short-term features are learned by LSTM [9] with the user's interaction data in a short period of time (such as hourly, weekly). The item's short-term features are learned by attention machine [16] according to all users' behavior at the latest moment. The personalized recommendation process applies implicit and explicit interactions of users and items to long-term and short-term of users and items features.

We summarize our main contributions as follows:

- We design a lightweight KG based on one topic domain without extra auxiliary data, and explore the nature of interaction under less information by relation extraction and knowledge reasoning.
- We learn implicit and explicit interaction of users and items by TransE according to the second-order proximity between the entities, which is determined by the shared neighborhood structures of the entities.
- KGTR considers freshness and popularity of items in recommendation, and learning the items' short-term feature by attention machine with all users behavior at previous moment.

## 2 Related Work

### 2.1 Knowledge Representation Learning

User/item clustering or matrix factorization techniques only represent single relation between the connectivity entities. Most existing methods have been designed to learn multi-relations from latent attributes [5,18]. Making use of multi-relational KG in recommender systems has been found to be effective in recent years.

TransRec [6] represented a user as a relation vector to capture the transition from the previous item to the next item in large sequences. A user's previous preference is important for predicting the next item in sequence recommendations, but the social relationships among users cannot be overlooked in context recommendations. TransTL [13] took both time and location into consideration with a translation-based model, which captured the joint effects of spatial and temporal information. Cao et al. [3] jointly learned the recommendation model and KG, which utilized the facts in KG to augment the user-item interaction. These models are not general for arbitrary recommendation scenarios, and ignore structures relationships among entities. Recent studies for KG focus on learning low-dimensional representations of entities and relations, and structural information of the graph is preserved. For completing knowledge graph and extracting relation from text, TransE learned a continuous vector space to preserve certain information of the graph, regarded relations as a translation between entities. TransE and its extensions TransH [21] and TransR [12] promoted prediction accuracy and computational efficiency by modeling multi-relational data. The most related work to ours is KR-EAR model, the method distinguished existing KG-relations into attributes and relations.

The entities embeddings were learned by building translation between entities according to relations, and attribute values embeddings were learned based on entity embeddings. We extend the KR-EAR to learn user's and item's representations. Three relations are defined according to the second-order proximity between the entities, which is determined by the shared neighborhood structures of the entities in KG.

### 2.2 Implicit and Explicit Interaction

Due to the significant impact for the quality of recommendations, many works have made great effort on gaining variety information. Cao et al. [2] described heterogeneity exists between users and between items, meanwhile detected the various coupling relationships to essentially disclose why a user liked an item. Methods [20] incorporated explicit and implicit couplings about users-items interactions and attributes' inter-coupled interactions. A classic work was the CoupledCF model [20], which integrated the explicit user-item couplings within/between user's and item's attributes and the implicit user-item couplings. The model were trained by deep learning. Different from these models, the explicit and implicit information of our model is more substantial by adding user's and item's interactions in various relationships.

User/item information has been increasingly involved into CF. NCF [7] can express and generalize matrix factorization by replacing the inner product with a neural architecture. NCF model may be supercharged with non-linearities, a multi-layer perceptron to learn the user-item interaction function. Wide&Deep [4] trained wide linear models by using cross-product feature transformations and deep neural networks to generalize recommendation. Unlike Wide&Deep model, we treat raw features as input by knowledge representation learning.

## 3 Our Proposed Model

In this section, we introduce our Model. Suppose there is a sparse user-item rating matrix that consists of users, items, and the rating. The rating is represented by numerical values from 1 to 5, where the higher value indicates the user has more interest in an item. Meanwhile, there are various attributes of users and items, such as gender and profession, which are important additional information for recommendation result. Given a dataset with user-item rating matrix and explicit attributes, we aim to build temporal personalized recommendation model for a user, and recommend a ranked list of items that are of interest to her/him accordingly.

As shown in Fig. 1, long-short term features of users and items are jointed in recommender process. The attributes features and static features capturing by knowledge representations learning are considered as the long-term features of users and items, that is explained in Sect. 3.1. The static features of items browsed by user previously are treated as input to LSTM. The fashionable items are interacted by users at the latest moment, and their attributes features and static features are served as input to attention machine. The implicit and explicit interactions of users and items are blended into long-short term of users and items features to recommendation.

### 3.1 Knowledge Representation Learning for Interaction

We design a lightweight KG with information of the dataset. As shown in Fig. 2(a), the users and items are treated as entities in KG. When the user has

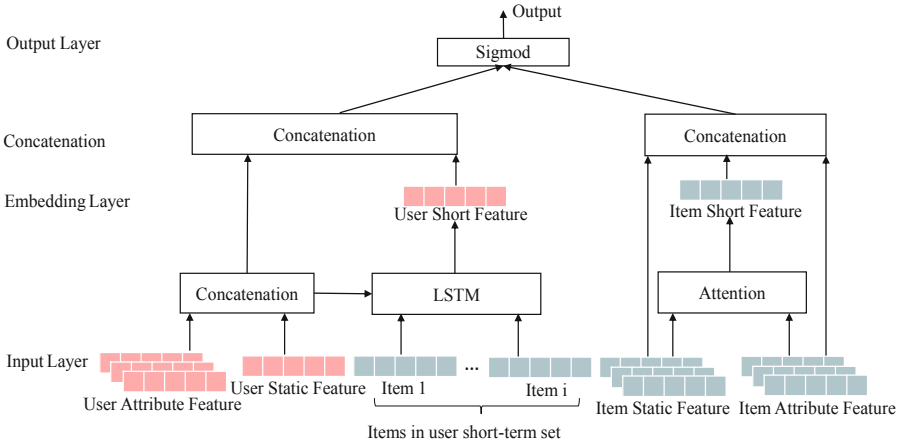


**Fig. 1.** The framework of knowledge graph-based temporal recommendation model

rated the item, there is a edge between the user and the item. The attributes of users and items are linked with corresponding entities. According to the extracted neighborhood structures of the entities in KG, we express first-order and second-order proximity [10] as three relationship definitions for learning static features.

We define the attribute triple additionally, for the purpose of learning attributes features based on former relationships representations. Our objective is to learn embeddings of users, items, and attributes preserving the structures information and semantic relations. The static feature belongs to the implicit interactions within/between user and item. The attribute feature is part of the explicit information.

**Definition 1.** *Rating Activity. As shown in up of Fig. 2(b), a rating activity is a triple* $(u, r, v)$*, which means user* $u$ *give a rating to item* $v$*.*

**Definition 2.** *Users Relationship. As shown in left of Fig. 2(b), a triple* $(u_i, v, u_j)$ *represent users relationship, which implies both user* $u_i$ *and* $u_j$ *give ratings to item* $v$*.*

**Definition 3.** *Items Relationship. As shown in right of Fig. 2(b), Item relationship is a triple* $(v_i, u, v_j)$*, which shows user* $u$ *give rating to item* $v_i$ *and* $v_j$*.*

**Definition 4.** *Attribute Triple. An attribute triple of user or item is a triple* $(u/v, a, e)$*, which indicates the attribute* $a$ *of user* $u$ *or item* $v$ *with values* $e$*, such as* $(u_1$*, gender, female*$)$ *illustrates the gender of user* $u_1$ *is female.*

We aim to embed users and items to capture the implicit and explicit correlations between them. We usually optimize the probability $\mathrm{P}(u, r, v)$, $\mathrm{P}(u_i, v, u_j)$ and $\mathrm{P}(v_i, u, v_j)$ for learning from relational triples. In this paper, we adopt TransE

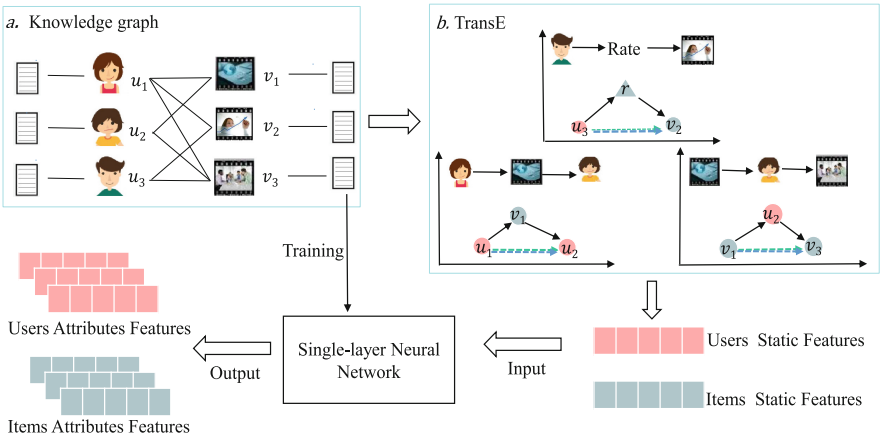


**Fig. 2.** Knowledge representation learning for implicit and explicit interaction

to encode relational triples. So the probability $\mathrm{P}(u, r, v)$ is formalized as follows:

$$P(u,r,v) = \sum_{(u,r,v^+)\in KG} \sum_{(u,r,v^-)\in KG^-} \sigma(g(u,r,v^+) - g(u,r,v^-)) \tag{1}$$

where $\sigma(x) = 1/(1 + exp(x))$ is sigmoid function, $g(\cdot)$ is the energy function which indicates the correlation of rating r and entity pair $(u, v)$. The $KG$ and the $KG^-$ are the positive and negative instances set, respectively. $KG^-$ contains incorrect triplets constructed by replacing tail entity in a valid triplet randomly. The probabilities of $\mathrm{P}(u_i, v, u_j)$ and $\mathrm{P}(v_i, u, v_j)$ are similar. Here, we can follow TransE to define the function $g(u, r, v)$ as Eq. 2:

$$g(u,r,v) = ||u + r - v||_{L_1/L_2} + b_1 \tag{2}$$

where $b_1$ is a bias constant. A classification model is used for capturing the correlations between entities and their attributes. Hence, we consider the probability $\mathrm{P}(u, a, e)$ for each triple $(u, a, e)$ and $\mathrm{P}(v, a, e)$ for each triple $(v, a, e)$, and formalize the probability $\mathrm{P}(u, a, e)$ for example, it is formalized as follows:

$$P(u,a,e) = \sum_{(u,a,e^+)\in KG} \sum_{(u,a,e^-)\in KG^-} \sigma(h(u,a,e^+) - h(u,a,e^-)) \tag{3}$$

where $h(\cdot)$ is the scoring function for each attribute value of a given entity. The function $h(\cdot)$ is described in Eq. 4. We first transform entity embedding into the attribute space by a single-layer neural network. For training attribute embedding, we calculate the semantic distance between the transformed embedding and it, as shown in Eq. 4:

$$h(u,a,e) = ||f(uW_a + b_a) - e_{ae}||_{L_1/L_2} + b_2 \tag{4}$$

where $f(\cdot)$ is a nonlinear function such as $tanh$, $W_a$ is the parameters by learning, $e_{ae}$ is the embedding of attribute value $a$ and $b_2$ is a bias constant.

The final embeddings of users' and items' static features propagating information between triples are recorded as $U_r, V_r$, respectively. And the attribute embeddings of users' and items' are written as $U_a, V_a$, respectively. The static features based on knowledge representation learning remain relatively stable over time, while a user's preference is affected by current prevalence. The freshness and temporal dynamics of the items are more likely to improve recommendation. Therefore, we extend our model by including user-item temporal information. Different from existing models, the item's short-term features are also discussed.

### 3.2 Temporal Recommendation

**Users Preference.** The users' short-term features are learned by recurrent neural networks (RNN). Instead of modeling the user history sequence using RNN which is difficult to calculate, our model combines users' static features and attribute features as pre-train input. This can make neural network training faster and more effective. The key issue of dynamic preferences is to choose the granularity of each input time spot $t$. Using smaller time spans can capture more fine-grained interest changes, but the feature space is very sparse and learning process is difficult [14]. Having large time spans may lead to sufficient content at each time spot, but makes the model less adaptive for capturing users' dynamics change. Unlike the previous model, we order 16 items for one user according to the latest browsing records. That can sure the enough context in behavior sequence to train user preference. To this end, we propose leveraging LSTM in capturing sequential patterns, and use it to model user's recent interaction trail. The output of LSTM $U_S$ is took as the users' short-term features.

**Items Preference.** The items' popularity are changing over time, and the features of most fashionable items currently have a greater impact on user preference. Here, we apply attention to obtain items' short-term characteristics. Attention can keep the contextual sequential information and capture the relationships between elements in the sequence. The items viewed by all users in the latest hour are considered as the items sequence. That is matched with items of the whole training datas $C$ to refine representation. The input of attention consists of items' attribute features and static features. The output is a weighted sum of the items, where the weight matrix $T^t$ is determined by similarity. Similar to [16], the attention vector are calculated at each output time $t$ over the input items $(1, \ldots, I)$ with Eq. 5.

$$\begin{aligned} T_i^t &= z^T tanh(W_c c_t + W_y y_i) \\ S_i^t &= softmax(T_i^t) \\ V_s' &= \sum S_i^t y_i \end{aligned} \tag{5}$$

The vector $z$ and matrices $W_c, W_y$ are learnable parameters, $c_t$ is the train item at time t and $y_i$ is $i$-th item of input sequence. The $i$-th item of vector $T^t \in R^I$

indicates the similarity between $y_i$ and the training datas. The attention weight matrix $S^t$ is created by normalizing similarity scores with softmax. Lastly, we concatenate $c_t$ with $V_s'$, which is regarded as $c_{t+1}$ at the next time step. Here, the final attentive output $V_s$ can be viewed as item's short-term feature.

### 3.3 Model Learning

**Objective Function.** Our task is to predict the item which the user will interact at next time, according to given long-short term preferences of users and items. A straightforward solution is to combine the outputs of their characteristics. So $U_{s,a,r}$ are the concatenation of $U_s, U_a, U_r$, and $V_{s,a,r}$ are the concatenation of $V_s, V_a, V_r$. Similar with the NCF, hidden layers are added on the concatenated vector by using a standard multi-layer perceptron (MLP) to learn the long-short term features. Specifically, the model can be formulated as

$$\begin{aligned} q_1 &= \Phi_1(U_{s,a,r}, V_{s,a,r}) = \begin{bmatrix} U_{s,a,r} \\ V_{s,a,r} \end{bmatrix} \\ \Phi_2(q_1) &= \alpha_2(w_2^T q_1 + b_2) \\ &\cdots \\ \Phi_l(q_{l-1}) &= \alpha_l(w_l^T q_{l-1} + b_l) \\ \hat{y}_{uv} &= \sigma(h^T \Phi_l(q_{l-1})) \end{aligned} \tag{6}$$

where $w_x, b_x$ and $\alpha_x$ denote the weight matrix, bias vector, and ReLU activation function for the $x$-th layer's perceptron, respectively. $\hat{y}_{uv}$ indicates whether the user $u$ is likely to interact with the item $v$.

Considering implicit feedback of interaction, we treat the value of $y_{uv}$ as a label. 1 means user $u$ has browsed item $v$, and 0 otherwise. The prediction score $\hat{y}_{uv}$ represents how likely $u$ interacts with $v$. We limit the output $\hat{y}_{uv}$ in the range of [0,1], thus, the output is achieved by using a probabilistic function as the activation function. Finally, we define the likelihood function as

$$p(y, y^-|\Theta_f) = \prod_{(u,v)\in y} \hat{y}_{uv} \prod_{(u,v)\in y^-} (1 - \hat{y}_{uv}) \tag{7}$$

Taking the negative logarithm of the likelihood, we gain the objective function to minimize for KGTR in Eq. 8.

$$\begin{aligned} L &= -\sum_{(u,v)\in y} \log \hat{y}_{uv} - \sum_{(u,v)\in y^-} \log(1 - \hat{y}_{uv}) \\ &= -\sum_{(u,v)\in y\cup y^-} y_{uv} \log \hat{y}_{uv} + (1 - y_{uv}) \log(1 - \hat{y}_{uv}) \end{aligned} \tag{8}$$

For the negative instances $y^-$, we uniformly sample them from unobserved interactions in each iteration and control the sampling ratio about the number of observed interactions. The sigmoid function restricts each neuron to be in [0,1], where neurons stop learning when their output is near either 0 or 1.

We optimize the proposed approach with adaptive gradient algorithm which could adapt the step size automatically. Hence it reduces the efforts in learning rate tuning. In the recommendation stage, candidate items are ranked in ascending order based on the recommendation score computed by Eq. 8, and the top ranked items are recommended to users.

## 4 Experiments

In this section, we evaluate our proposed framework for movie recommendation scenarios. We test our methods against the related baselines for recommendations items to users. The experimental results demonstrate that our method better than many competitive baselines.

### 4.1 Experimental Settings

**Dataset Description.** We used MovieLens-1M[1] dataset in our experiments. The dataset consists of one million ratings from 6,040 users and 3,952 items, user auxiliary information (Gender, Age, Occupation and Zip code) and some item attributes (Genres, Title and release dates). We transformed the original rating matrix scaled from $R \in \{1, 2, ..., 5\}$ into a binarized preference matrix $R \in \{0, 1\}$. Each rating was expressed as either 0 or 1, where 1 indicates an interaction between a user and an item, otherwise 0. Then we sampled four negative instances per positive instance.

For each user, we sorted the user-item interactions by the time stamps at first. Then we took her/his latest interaction as the test positive instance and utilized the remaining data for training positive instance. Finally we randomly sampled 99 items that are not interacted by the user as the test negative instance and randomly sampled four negative instances for per positive instance.

**Evaluation Metrics.** Similar to [4], we ranked the test item among the 100 items and used Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) to evaluate the performance of a ranked list [8]. The HR intuitively measures whether the test item is included in top-K list. The NDCG measures the position of the hit on top-K list. The higher NDCG scores show that the test item hits at top ranks. We calculated both metrics for each test user and reported the average score.

**Baseline Methods.** We evaluated our framework from in three versions based on the different input.

- KGTR_user: for every user, we used the sequence of items recently watched by user as the input vector of LSTM and the sequence of items recently viewed by all users as the input of attention. The input vector of LSTM was learned by translation representation learning and the input vector of attention was one-hot encoding of items;

[1] https://grouplens.org/datasets/movielens/.

- KGTR_item: different from KGTR_user, the input vector of LSTM is one-hot encoding of items and the input vector of attention was learned by translation representation learning.
- KGTR_NCF:different from KGTR_user and KGTR_item, both the input vector of LSTM and attention were learned by translation representation learning.

The following relevant and representative state-of-the-art methods were used as the baselines to evaluate our methods.

- NCF [7]: It presents a neural architecture replacing the inner product and proposes to leverage a multi-layer perceptron to learn the user-item interaction function.
- CoupledCF [20]: This model proposes a neural user-item coupling learning for collaborative filtering, which jointly learns explicit and implicit couplings within/between users and items.
- Wide&Deep [4]: It combines memorization and generalization for recommendation, which involves feature engineering (such as cross-product features) of the input to the wide network.

**Parameter Settings.** The configurations of TransE are $k = 100, b_1 = 7, b_2 = -2$, and taking $L_1$ as distance metric. The KGTR model was implemented in Python based on the Keras framework. We selected 16 items recently watched by every user as the input vector of LSTM, and the items viewed by the users in the latest hour were selected as input to the attention.

To determine hyper-parameters of KGTR, we randomly sampled one interaction for each user as the validation data and tuned hyper-parameters on it. All KGTR models were learnt by optimizing the log loss of Eq. 8. We used mini-batch Adam as the optimizer for our model. We initialized the embedding matrix with a random normal distribution (the mean and standard deviation are 0 and 0.01 respectively). All biases are initialized with zero. We tested all combinations of the batch size (S = {128, 256, 512, 1024}) and the learning rate (R = {0.0001, 0.0005, 0.001, 0.005}) that S = 256 and R = 0.001 was the best combination. From Fig. 3, we could see that HR@10 and NDCG@10 increased firstly. When S = 256 and R = 0.001, the performance of KGTR was best, and then HR@10 and NDCG@10 decreased or stabilized with the increase of batch size and learning rate. So we set S = 256 and R = 0.001 as the optimal parameters.

### 4.2 Results and Analysis

First, the performances of the KGTR and baselines were shown in Table 1 for TOP@10 recommendation. From Table 1, A number of interesting observations could be noted. Our method KGTR_item and CoupleCF were superior to NCF and Wide&Deep in both HR@10 and NDCG@10. The reason is that the user preference and item popularity contributed to improve the recommendation performance. Both the KGTR_item and CoupleCF integrated item popularity, user-item interaction and implicit user-item interaction to gain the best performance.

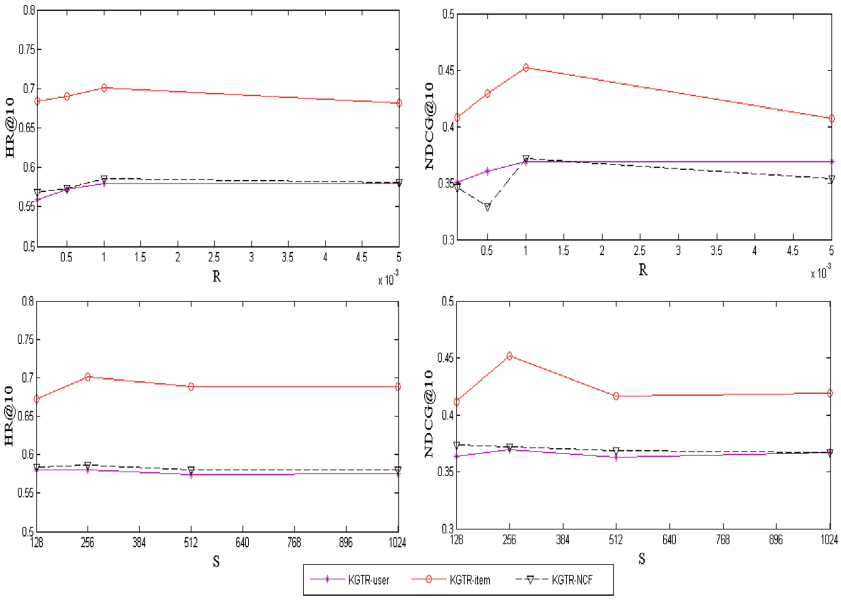


**Fig. 3.** Performance of KGTR models w.r.t the learning rate (bach size = 256) and the bach size s(learning rate = 0.001)

The results confirmed that the interactions between users and items were useful. Beside, KGTR_item is slightly worse than CoupleCF. That may because the CoupleCF used the rating values from the users and the items, our method only used the rating relationship between the users and the items.

Second, we also tested the top@K item recommendations in Fig. 4. As previously introduced, KGTR models were customized to three versions: KGTR_user, KGTR_item and KGTR_NCF. KGTR_item was compared with KGTR_user, KGTR_NCF and all the baselines. Figure 4 shows the performance of the top@K recommendation, where $K$ ranges from 1 to 10. As shown in Fig. 4, all the baselines and KGTR_item highly outperformed KGTR_user and KGTR_NCF. That were mainly because KGTR_user and KGTR_NCF were personalized recommendation method via learning individual user's preference. When learning user

**Table 1.** HR@10 and NDCG@10 for Top-10 item recommendation

| | *#HR@10* | *#NDCG@10* |
|---|---|---|
| NCF | 0.6947 | 0.4149 |
| CoupledCF | 0.7310 | 0.4819 |
| Wide&Deep | 0.6864 | 0.4082 |
| KGTR_user | 0.5801 | 0.3691 |
| KGTR_item | 0.7012 | 0.4518 |
| KGTR_NCF | 0.5861 | 0.3719 |

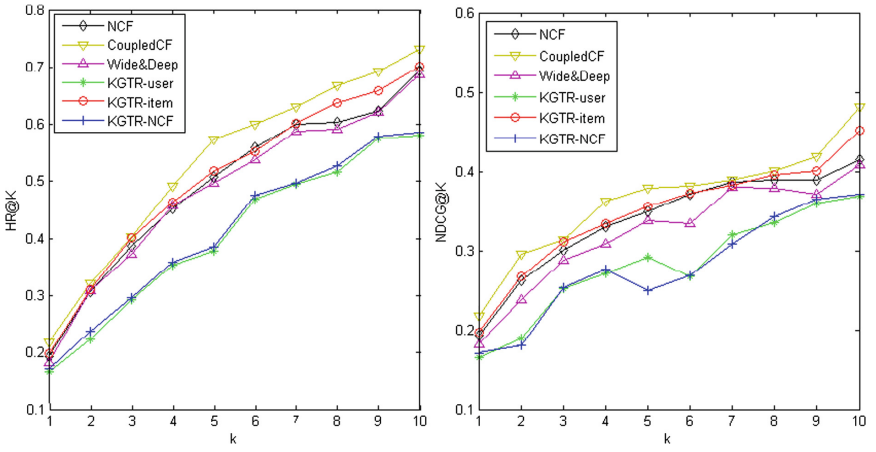


**Fig. 4.** HR@K and NDCG@K results comparison between our framework and related baselines

temporary preferences by LSTM, we selected 16 items viewed recently by every user, and did not divide the time period according to the traditional time spots (minutes, hours or days). Therefore, recommendation performances in KGTR_user and KGTR_NCF were not improved by adding user preferences.

## 5 Conclusions

In this paper, we proposed a knowledge graph-based temporal recommendation (KGTR) model to explore implicit/explicit and latest interactions between users and items. Firstly, we designed a lightweight KG based on one domain knowledge without extra information. We defined three categories relationships by TransE, according to the implicit/explicit interactions of users' and items'. That could capture global structural dependencies in the historic behavior and united information between triples. Taking the different impact of long-short term interest into account, our model was trained by deep learning. Specially, the popular features of items were jointed in the course of learning dynamic preferences. The experimental results showed significant improvement has obtained over state-of-the-art baselines on large dataset. In future, we will study how to incorporate short-term users' preferences and the ratings for better recommendation.

**Acknowledgement.** This work was supported by the Fundamental Research Funds for the Central Universities (No. ZXH2012P009).

## References

1. 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)

2. Cao, L.: Non-IID recommender systems: a review and framework of recommendation paradigm shifting. Engineering **2**(2), 212–224 (2016)
3. Cao, Y., Wang, X., He, X., Chua, T.S., et al.: Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. arXiv preprint arXiv:1902.06236 (2019)
4. Cheng, H.T., et al.: Wide & deep learning for recommender systems (2016)
5. Han, X., Shi, C., Wang, S., Philip, S.Y., Song, L.: Aspect-level deep collaborative filtering via heterogeneous information networks. In: IJCAI, pp. 3393–3399 (2018)
6. He, R., Kang, W.C., McAuley, J.: Translation-based recommendation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 161–169. ACM (2017)
7. 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)
8. He, X., Tao, C., Kan, M.Y., Xiao, C.: Trirank: review-aware explainable recommendation by modeling aspects (2015)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
10. Jian, T., Meng, Q., Wang, M., Ming, Z., Yan, J., Mei, Q.: Line: large-scale information network embedding. In: 24th International Conference on World Wide Web, WWW 2015 (2015)
11. Lin, Y., Liu, Z., Sun, M.: Knowledge representation learning with entities, attributes and relations. Ethnicity **1**, 41–52 (2016)
12. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of AAAI (2015)
13. Qian, T.Y., Liu, B., Hong, L., You, Z.N.: Time and location aware points of interest recommendation in location-based social networks. J. Comput. Sci. Technol. **33**(6), 1219–1230 (2018)
14. Song, Y., Elkahky, A.M., He, X.: Multi-rate deep learning for temporal recommendation. In: Proceedings of ACM SIGIR, pp. 909–912. ACM (2016)
15. 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)
16. Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., Hinton, G.: Grammar as a foreign language. Eprint Arxiv, pp. 2773–2781 (2015)
17. Wang, H., Zhang, F., Xie, X., Guo, M.: DKN: deep knowledge-aware network for news recommendation. In: Proceedings of WWW, pp. 1835–1844. WWW (2018)
18. Wang, X., Peng, Z., Wang, S., Yu, P.S., Fu, W., Hong, X.: Cross-domain recommendation for cold-start users via neighborhood based feature mapping. In: Pei, J., Manolopoulos, Y., Sadiq, S., Li, J. (eds.) DASFAA 2018. LNCS, vol. 10827, pp. 158–165. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91452-7_11
19. 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)
20. Zhang, Q., Cao, L., Zhu, C., Li, Z., Sun, J.: CoupledCF: learning explicit and implicit user-item couplings in recommendation for deep collaborative filtering. In: IJCAI, pp. 3662–3668 (2018)
21. Zhen, W., Zhang, J., Feng, J., Zheng, C.: Knowledge graph embedding by translating on hyperplanes. In: Proceedings of 28th AAAI (2014)