ISBN: 978-981-18-5852-9
WCSE 2022 Spring Event: 2022 9th International Conference on Industrial Engineering and Applications
doi: 10.18178/wcse.2022.04.034

Transformer Recommendation Algorithm with Fusion Knowledge Graph

Honghui Xie[1], Jun Yang[1+], Yi Liu[1] and Cong Huang[1]

[1] School of Software, Jiangxi Agricultural University, Nanchang 330045, China

Abstract. To address the problem that traditional knowledge graph recommendation algorithms only consider the message aggregation of neighbor nodes but not the sequence information of knowledge graph neighbor nodes, this paper proposes a Transformer recommendation algorithm that fuses knowledge graphs. Under the item knowledge graph space, the sequence information is constructed from the target item and its neighbors, and the sequence information is encoded using the Transformer model to capture the features of the item. Finally, MLP is used to integrate the obtained item feature vectors and user embedding vectors and send them to the prediction module to obtain user click-through rate prediction by vector inner product operation. Comparast experiments were performed on three publicly available datasets. On the MovieLens-20M dataset, compared with the optimal baseline KGCN, the AUC and F1 improve by 1.74% and 7.18%, respectively. On the Last.FM dataset, the AUC and F1 improve by 7.07% and 5.92%, respectively. On the Book-Crossing dataset, the AUC and F1 improve by 1.52% and 1.62%, respectively, compared to the optimal baseline. Thus, it is verified that the effectiveness of the algorithm is improved by considering the sequence information of the neighboring nodes of the knowledge graph.

Keywords: Knowledge Graph, Transformer, Recommendation Algorithm

1. Introduction

Currently, with the rapid development of Internet technology, users are overwhelmed by the large amount of information and it is difficult for them to find the desired information. To solve the problem of information overload, recommendation systems are important, which guide users to discover the products or services they are interested in from a large number of possible choices. Currently, the main recommendation algorithms are rule-based recommendation, collaborative filtering, content-based recommendation, and network-based recommendation.

Among the many recommendation algorithms, collaborative filtering [1] is the solution to most personalized recommendations. Collaborative filtering mines the behavioral preferences of users based on the interaction information between users and items to predict their interest preferences. The essence of collaborative filtering algorithm is to assume that historical behaviors that users have in the past tend to be repeated in the future. However, collaborative filtering suffers from the problem of data sparsity .And when there are insufficient features in the data, the recommendation effect is relatively poor. Therefore, alleviate this problem by adding auxiliary information such as social networks, attribute information of items, etc. Since knowledge graphs contain rich information of items, they are also applied in recommendation algorithms. There are three main types of recommendation algorithms for knowledge graphs: embedding-based methods, path-based methods and neighborhood-based methods. The embedding-based approach is to use the information embedding of the knowledge graph to obtain vector representations of users and items to expand the vectors of users and items in the knowledge recommendation algorithm. Wang proposed that DKN [2] uses KCNN to extract information from news headlines to construct knowledge graph and implemented embedding of knowledge graph using the TransH [3] method, and then takes the entity representation vector of the knowledge graph as input to the convolutional network to extract news features. The path-based approach is to use the path information of the knowledge graph. Yu et al [4] proposed HeteRec can implement meta-path-based recommendation in heterogeneous networks, namely obtaining the user's preference matrix based on the meta-path and then get implicit feedback for users and items with

+ Corresponding author..
E-mail address: ycjun515@163.com.

matrix decomposition. However, the path-based approach needs to set the path in advance, and the path has a great impact on the recommendation effect, so it is more difficult to achieve. The neighborhood-based recommendation is to use the structural information of the knowledge graph to extend the vector representation of users and objects in the recommendation system. Wang et al. proposed the Knowledge Graph Intent Network (KGIN) [5], which not only achieves higher-order information of the model by aggregating multi-order neighbor information, but also extracts useful information about user intent and encodes it into as a representation of users and projects.

Traditional recommendation algorithms usually map the original ID-like features into a low-dimensional space by embedding techniques and then input them into the MLP network, without considering the sequence information. Currently, in the Deep Interest Network (DIN) model, the user's diverse interest distribution is portrayed by introducing the Attention mechanism to calculate the correlation between the user's historical behavior sequence and the current Item. In recent years, it has become a trend to apply Transformer structures in recommendation algorithms based on the great results achieved by Transformer [6] in NLP, such as BERT [7]. For example, AutoInt [8], which is applied to feature combination of CTR prediction models, BST [8], which is used for behavioral sequence modeling, and PRM [9], which is a reordering model, have proved that the introduction of Transformer into the search recommendation domain can bring good results. Inspired by Transformer in recommendation algorithms, this paper combines Knowledge Graph and Transformer Network (KGNT) for extracting the hidden information behind the sequences in the project knowledge graph space, which can better represent the project.

The rest of the paper is organized as follows: Section 2 describes the model. Section 3 gives the experimental results and sensitive parameters. Section 4 summarizes the work of this paper.

2. Knowledge Graph Transformer Network

2.1. Problem Description

In the recommendation algorithm scenario, let $U=\{u_1,u_2,\cdots,u_n\}$ and $V=\{v_1,v_2,\cdots,v_n\}$ enote a set of users and a set of items, respectively, where n is the number of users and m is the number of items. Assuming $R\in\mathbb{R}^{p\times m}$ is the user-item interaction matrix Y. If a user views the item $y_{uv}=1$, otherwise $y_{uv}=0$ means the user has not viewed the item. In addition, there is a knowledge graph G, which consists of entity-relationship-entity triples < h,r,t >, where $h\in\mathcal{E},r\in\mathcal{R}$, and $t\in\mathcal{E}$ denote the triple head entity, relationship, and tail entity, respectively, $\mathcal{E}$ and R denote the set of entities and the set of relationships in the knowledge graph, respectively. For example, <Song of Ice and Fire, Author, George Martin>, indicates that George Martin wrote a book "Song of Ice and Fire".Given a user-item interaction matrix Y and a knowledge graph G, specifically learn the objective function : $\hat{y}_{uv}=F\left(u,v^u|\ \Theta,\mathrm{R},G\right)$, where $\hat{y}_{uv}$ denotes the probability of user-item interaction, the features of the user by the embedding vector $u\in R^f$ and the features of the item by the embedding vector $\mathrm{v^u}\in\mathrm{R^f}$, f is the length of the embedding vector, and Θ is the parameter of the F function.

2.2. KGNT Algorithm

The KGTN is used to obtain the higher-order structure information among entities in the knowledge graph. For item v,N(v) denotes the set of domain objects of items in the item knowledge graph, i.e., the circular node labeled 1 in Fig.1. Propagating this set of objects outward in the knowledge graph, a multilayer set of neighbors $S_u^k(k=1,2,\ldots,H)$ is formed. The graph structure of the knowledge graph is huge. And to ensure stable and efficient training of each batch, the size of the neighbors in the fixed model, i.e., k in the multilayer neighbor set, is a configurable parameter. The blue node in Fig.1 is the current node, and the circular node labeled 1 is a layer 1 neighbor. As the number of layers k increases, the number of neighbor sets becomes very large, and thus H will not be chosen very large. And when H is large, it brings more noise than valid information. Also, to reduce the number of sets, a fixed number of neighbors with fixed size are sampled.

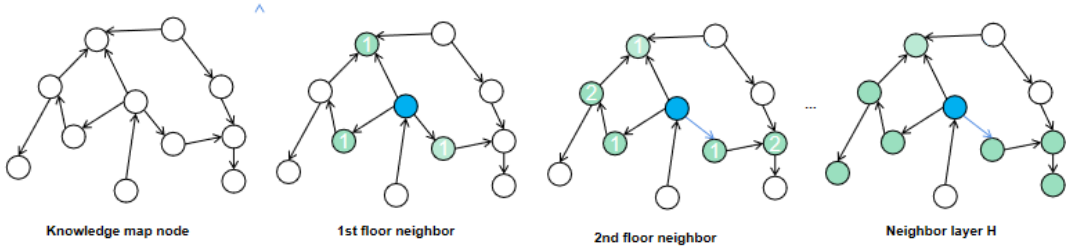

Fig. 1: multi layer neighbor sampling.

The KGTN model is adopted by graph to obtain the neighbors of the target node using the construction sequence of the knowledge graph neighbor nodes. The sequence of neighbor nodes is randomly arranged according to the neighbor sampling during the construction process. And KGNT mainly uses the Encoder part of Transformer to capture the correlation between the target item and the target neighbor sequence.The Transformer Encoder Layer model diagram is shown in Fig.2 below.

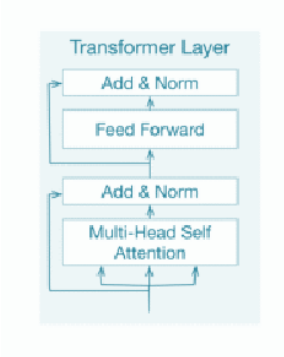

Fig. 2: Transformer Encoder

The Classic Transformer Encoder part consists mainly of six identical layers, each containing two parts: the Multi-Head Attention layer and the Feed Forward Full Connection layer, both of which contain Residual Connection residual connection and Add &Norm data normalization.

Self-attention layer: Using Multi-Head Attention is an implementation of self-attention:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V} \tag{1}$$

Where Q is the query, K and V are the keys and values respectively. In KGTN, Q,K and V are obtained from the vector linear mapping of target items and target vector neighbor sequences in the graph adoption. The results of the Self-attention layer are concatenated and linearly transformed to obtain the final Multi-Head Attention results.

$$\mathbf{S} = \mathbf{MH}(\mathbf{E}) = \text{Concat}(\text{head}_1, \text{head}_2, \cdots, \text{head}_h)\mathbf{W}^H \tag{2}$$

$$\text{head}_i = \text{Attention}\left(\mathbf{E}\mathbf{W}^Q, \mathbf{E}\mathbf{W}^K, \mathbf{E}\mathbf{W}^V\right) \tag{3}$$

Where $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{d\times Q}$ corresponds to the parameter matrix, E is the vector of each neighbor in the input neighbor sequence, and h is the number of heads of multi-headed attention.

In addition to the Multi-Head Attention section mentioned above, the Encoding section of the Transformer includes a Point-wise Feed-Forward Networks (FFN) section. The purpose of the FFN is to increase the nonlinearity of the model, as defined below:

$$\mathbf{F} = FFN \tag{4}$$

To prevent overfitting of the model, dropout and LeakyReLU are used in both the Self-attention layer and FFN, and the final Self-attention layer and FFN are as follows.

$$\mathbf{S}^{'} = \text{LayerNorm}\,(\mathbf{S} + \text{Dropout}\,(MH(\mathbf{S})) \tag{5}$$

$$\mathbf{F} = \text{LayerNorm}\,(\mathbf{S}^{'} + \text{Dropout}\,(\text{LeakyReLU}\,(\mathbf{S}^{'}\mathbf{W}^{(1)} + b^{(1)})\mathbf{W}^{(2)} + b^{(2)})) \tag{6}$$

Where $\mathbf{W}^{(1)}, b^{(1)}, \mathbf{W}^{(2)}, b^{(2)}$ are the learnable parameters and LayerNorm is the standard data normalizatio n layer.

The above structure is the complete Transformer Encoder Layer, and in order to be able to learn the hidden information behind the sequence more effectively, the model is stacked.

$$\mathbf{S}^{b} = SA\left(F^{(b-1)}\right) \tag{7}$$

$$\mathbf{F}^{b} = FFN\left(\mathbf{S}^{b}\right), \forall i \in 1,2,\cdots,n \tag{8}$$

b is the number of layers in the Transformer stack. In the experimental section for analysis b=1 is preferred.

2.3. Projections

After Transformer processing, the expression vector v, which is extracted to represent the item based on the sequence of item neighbors, and the user embedding vector u are obtained and fed into the objective function in 2.1 to calculate the user click item probability.

The loss function of the KGTN model is as follow:

$$\mathcal{L} = \sum_{u \in U}\left(\sum_{v:y_{uv}=1} \mathcal{J}\left(y_{uv}, \hat{y}_{uv}\right) - \sum_{i=1}^{T^{u}} \mathrm{E}_{v_i \sim P(v_i)} \mathcal{J}\left(y_{uv_i}, \hat{y}_{uv_i}\right)\right) + \lambda \| L\|_2^2 \tag{9}$$

Where $\mathcal{J}\left(\mathrm{y}_{\mathrm{uv}}, \hat{\mathrm{y}}_{\mathrm{uv}}\right)$ is the cross-entropy loss between the true and predicted values, P is the negative sample sampling. T^{u} denotes the number of samples sampled, and the last part is the L2 regular term.

3. Experiment

3.1. Data set description

This article uses three publicly available datasets, MovieLens-20M, Book-Crossing, Last.FM. The MovieLens-20M dataset is from the MovieLens website and contains 20 million user movie ratings (1-5). The Book-Crossing dataset is from the Book-Crossing community and contains one million data score (0-10). The Last.FM dataset is from the FM online music system and contains ratings information from 2,000 users. The three datasets are user display feedback, and the dataset is preprocessed to convert the display feedback into implicit feedback. The mark of 1 indicates a positive user rating for the item and the mark of 0 indicates a negative user rating for the item. The positive rating threshold for the MovieLens-20M dataset is 4. Book-Crossing and Last.FM do not have a threshold, as these two datasets are relatively small. The knowledge graph processing of the corresponding datasets was obtained from the KGCN [10] public preprocessed knowledge graph. The basic statistics of the three datasets are shown in Table 1.

Table 1: The basic statistics of three data sets

number	MovieLens-20M	Book-Crossing	Last.FM
users	138159	19676	1872
items	16954	20003	3846
interactions	13501622	173576	42346
entities	102569	25787	9366
relation	32	18	60
KG triples	499474	60787	15518

3.2. Evaluation index and parameter setting

In the process of model training and prediction, 50% cross validation is adopted to ensure that the algorithm is more fair. Dividing the data into five equal parts，for each test, randomly select one as the verification set, and the other four groups are the training sets. Each group of data is tested, and the average

value of the five test results is taken as the experimental result. Two indicators are used to measure the performance of each algorithm, namely AUC (area under curve) and F1, which reflect the performance of the algorithm in CTR (click through rate) prediction. The larger the value of the three indicators are, the better the performance of the algorithm is.

This article is tested on MovieLens-20M, Book-Crossing and Last.FM datasets. The superparameter settings of the model in this paper are shown in Table 2.

Table 2: The setting of super parameters

Super parameter	MovieLens-20M	Book-Crossing	Last.FM
K	4	8	8
d	32	64	16
H	2	1	1
λ	10-7	2×10-5	10-4
β	2×10-2	2×10-4	5×10-4
Batch size	65536	256	128

Where K is the number of neighbors, d is the size of the embedded vector. H is the depth of the receptive field. λ Represents the regularization parameter. β Indicates the training learning rate. Batch size is the training batch size.

3.3. Algorithm performance comparison

In order to verify the performance of KGNT, the proposed algorithm is compared with the classical recommendation algorithms SVD, LibFM and the knowledge graph-based recommendation algorithms LibFM+TransE, PER, CKE, RippleNet, KGCN, and the obtained experimental results are analyzed.

SVD: Matrix Factor Decomposition recommendation algorithm is a classical collaborative filtering algorithm that uses the interaction information of users and items for modeling.

LibFM [11][12]: A decomposer implementing stochastic gradient descent and selectable least squares as well as using Bayesian inference as features.

LibFM+TransE: Extends LibFM by attaching the entity representation learned by TransE to each user item pair.

PER: Treats knowledge graphs as heterogeneous information networks and extracts meta-path-based features to represent the connections between users and items.

CKE [13]:This model is based on the joint training of the knowledge graph in collaboration with a knowledge embedding based recommendation algorithm.

RippleNet[14][15]:The key idea of this approach is preference propagation, which continuously and automatically discovers the potential hierarchical interests of users by means of preference propagation in the knowledge graph, incorporating the knowledge graph approach into the recommender system.

KGCN[16]:A model that fuses knowledge graph features with graph convolutional neural networks to combine neighbor information with bias in the computation of a given item representation.

Comparing the performance of various recommendation algorithms, Table 3 shows the AUC and F1 of click-through rate prediction in three dataset recommendation methods, MovieLens-20M, Book-Crossing, and Last.FM[17].

Table 3: The Results for AUC and F1 in CTR prediction

Model	MovieLens-20M		Book-Crossing		Last.FM	
	AUC	F1	AUC	F1	AUC	F1
SVD	0.936	919	0.672	0.635	0.769	0.696
LibFM	0.959	0.906	0.691	0.618	0.778	0.710
LibFM+TransE	0.966	0.917	0.698	0.622	0777	0.709

CKE	0.924	0.871	0.677	0.611	0.774	0.673
PER	0.832	0.788	0.617	0.562	0.633	0.596
RippleNet	0.968	0.912	0.715	0.650	0.780	0.702
KGCN	0.977	0.932	0.721	0.675	0.749	0.754
KGTN	**0.994**	**0.999**	**0.732**	**0.686**	**0.802**	**0.786**

From the results in Table 3, we can see that on the three datasets of MovieLens-20M, Book-Crossing, and Last.FM, compared with the traditional recommendation algorithms SVD, LibFM, CKE, PER, LibFM+TransE, as well as RippleNet, which is the preferred propagation method in the knowledge graph and the aggregated neighbor-only method KGCN, the algorithms presented in this paper have all shown an improved performance. On the MovieLens-20M dataset, AUC and F1 improved by 1.74% and 7.18%, respectively, compared to the optimal baseline model KGCN. On the dataset Book-Crossing, AUC and F1 improved by 1.52% and 1.62%. And on the dataset Last.FM, AUC and F1 improved by 7.07% and 5.92%.

From the statistics of Table 1 dataset, Book-Crossing and Last.FM dataset are relatively sparse. In Table 3 dataset inside for both book and music recommendation effect has improved, indicating that KGNT can solve the sparse scenario. From the algorithms compared in Table 3, PER has the worst effect because it is difficult to define the best meta-path in the recommendation scenario; RippleNet and KGCN have better performance compared to other baseline models, and both of them utilize the neighbor information of the knowledge graph, indicating that the neighbor information of the knowledge graph has a greater impact on the recommendation effect.

3.4. Sensitivity analysis of parameters

The number of stacks on the captive model has an impact on the KGNT recommendation algorithm for the encoding part of the transformer. Experimenting with the size of the series of stacking numbers b set in Table 4 in MovieLens-20M, Book-Crossing and Last.FM respectively, it can be seen that the model performs best when the value of b = 1. When b is too high, the performance of the model suffers. b is too large, the model has too many parameters and introduces a lot of noise into the model.

Table 4: Effect of parameter D on AUC values

b	1	2	3	4
MovieLens-20M	**0.993**	0.986	0.980	0.950
Book-Crossing	**0.732**	0.724	0.718	0.712
Last.FM	**0.798**	0.790	0.781	0.778

When aggregating neighbours, the number of sampled neighbours has an impact on the recommendation algorithm in this paper. As shown in Table 5 below, the performance of the model is best when the value K = 4 or 8. When the value of K is too low and too high, the performance of the model suffers. A value of K that is too large is easily misled by noise and will introduce more noise, and a value of K that is too small does not aggregate more effective domain information.

Table 5: Effect of K parameters on AUC values

K	2	4	8	16	32	64
MovieLens-20M	0.992	**0.993**	0.991	0.990	0.985	0.988
Book-Crossing	0.723	0.730	**0.734**	0.731	0.728	0.729
Last.FM	0.792	0.796	**0.799**	0.798	0.794	0.793

Tabale 6 indicates the effect of aggregation depth on the performance of the algorithm. It can be seen that the algorithm in this paper is more sensitive to the aggregation depth, and a serious model run-down occurs when H = 3 or 4, because a larger depth introduces a large amount of noise. In the knowledge graph, as the number of hops increases, the relevance of neighboring information to entity items becomes weaker and the effective information becomes less. According to the experimental results, H=1 or 2 is sufficient for real scenarios.

Table 6: Effect of H parameters on AUC values

H	1	2	3	4
MovieLens-20M	0.984	**0.994**	0.834	0.623
Book-Crossing	**0.732**	0.716	0.638	0.553
Last.FM	**0.800**	0.736	0.559	0.546

Table 7 represents the effect of vector embedding dimensionality on the performance of the algorithm. Initially increasing the dimensionality of the vectors increases the performance of the algorithm, and too large d decreases the performance of the algorithm instead. The best results are obtained when d = 32 or 64, because a suitable d encodes more valid information without introducing more noise and also alleviates the overfitting phenomenon of the algorithm, and also reduces the training time of the algorithm. Too large d leads to overfitting and reduces the performance of the algorithm.

Table 7: Effect of d parameters on AUC values

d	4	8	10	32	64	128
MovieLens-20M	0.984	0.990	0.993	**0.995**	0.993	0.990
Book-Crossing	0.722	0.726	0.729	0.730	**0.732**	0.731
Last.FM	0.790	0.792	0.795	0.796	**0.801**	0.799

4. Conclusion

In this paper, we propose a Transformer recommendation algorithm that fuses knowledge graphs. Under the item knowledge graph space, the target item and its neighbors are constructed as sequence information. And the sequence information is encoded using the Transformer model to capture the features of the item. KGNT is able to cope with multiple datasets, discover hidden features in the recommender system, and learn the structural and semantic information of the knowledge graph. Through extensive experiments, KGNT has consistently outperformed state-of-the-art baseline models for movie, book, and music recommendations. At present, only the sequence information of neighbor vectors under the knowledge graph space is considered, and user intention is not added. In the future, historical behavior sequences from users will be added to the recommender system to mine the hidden features of users and improve the accuracy of recommendations.

5. Acknowledgements

This work was supported by the This work was supported by the Natural Science Foundation of Jiangxi Province - Surface Project (20212BAB205009) and the Science and Technology Foundation of Jiangxi Provincial Education Department (GJJ13266, GJJ180374, GJJ170303).

6. References

[1] SARWAR B M，KARYPIS G，KONSTAN J A，et al.Item based collaborative filtering recommendation algorithms[C]// Proceedings of the 10th International Conference on World Wide Web，2001：285-295.

[2] Wang H, Zhang F, Xie X, et al. DKN: Deep knowledge-aware network for news recommendation[C]//Proceedings of the 2018 world wide web conference. 2018: 1835-1844.

[3] Lin Y, Liu Z, Sun M, et al. Learning entity and relation embeddings for knowledge graph completion[C]//Twenty-ninth AAAI conference on artificial intelligence. 2015.

[4] Yu X, Ren X, Sun Y, et al. Personalized entity recommendation: A heterogeneous information network approach[C]//Proceedings of the 7th ACM international conference on Web search and data mining. 2014: 283-292.

[5] Wang X, Huang T, Wang D, et al. Learning Intents behind Interactions with Knowledge Graph for Recommendation[C]//Proceedings of the Web Conference 2021. 2021: 878-887.

[6] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in neural information processing systems. 2017: 5998-6008.

[7] Song W, Shi C, Xiao Z, et al. Autoint: Automatic feature interaction learning via self-attentive neural networks[C]//Proceedings of the 28th ACM International Conference on Information and Knowledge Management. 2019: 1161-1170.M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

[8] Chen Q, Zhao H, Li W, et al. Behavior sequence transformer for e-commerce recommendation in Alibaba[C]//Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data. 2019: 1-4.

[9] Pei C, Zhang Y, Zhang Y, et al. Personalized re-ranking for recommendation[C]//Proceedings of the 13th ACM Conference on Recommender Systems. 2019: 3-11.

[10] Wang H, Zhang F, Zhang M, et al. Knowledge Graph Convolutional Networks for Recommender Systems with Label Smoothness Regularization[J]. 2019.

[11] Xiao J, Ye H, He X, et al. Attentional factorization machines: Learning the weight of feature interactions via attention networks[J]. arXiv preprint arXiv:1708.04617, 2017

[12] STEFFEN R. Factorization machines with libFM[J]. ACM Transactions on Intelligent Systems and Technology (TIST),3, 3 (2012), 57.

[13] Takahashi R. Geometric quantization of coupled K\" ahler-Einstein metrics[J]. arXiv preprint arXiv:1904.12812, 2019.

[14] WANG H, ZHANG F, ZHANG M, et al. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems[C]// In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, August 4–8, 2019. New York: ACM, 2019:968–977.

[15] WANG X, WANG D, XU C,et al. Explainable reasoning over knowledge graphs for recommendation[C]//In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI,January 27-February 01, 2019.PaloAlto, CA, USA:Assoc Advancement Artificial Intelligence, 2019:5329–5336.

[16] DEFFERRARD M,BRESSON X,VANDERGHEYNST P.Convolutional neural networks on graphs with fast localized spectral filtering[C]//Advances in Neural Information Processing Systems,2016:3844-3852.

[17] DERR T,MA Y,TANG J.Signed graph convolutional networks[C]//2018 IEEE International Conference on Data Mining(ICDM),2018:929-934.