Recurrent Knowledge Attention Network For Movie Recommendation

YiHua Cheng[1]
DaLian Polytechnic University
DaLian, LiaoNing, China
E-mail:340354813@qq.com

Na Liu[3]
DaLian Polytechnic University
DaLian, LiaoNing, China

Ying Lu *
DaLian Polytechnic University
DaLian, LiaoNing, China
*Corresponding author: Ying Lu(1970-), Professor

XiaoJun Tang[4]
DaLian Polytechnic University
DaLian, LiaoNing, China

***Abstract*—** A primary concern of the current recommendation system is how to provide personalized recommendation to users and improve the accuracy and user satisfaction. The Knowledge Graph(KG) provides a new way to improve the recommendation system. This paper proposes a movie recommendation model based on Recurrent Neural Network(RNN) and KG─RKAN, which uses the auxiliary information in the KG to look for the potential interests of users for personalized recommendations. In addition, in order to solve the problem of user's individual interests, an attention module was designed in RKAN,using different weights to converge user's interest; multiple sets of negative samples were used for comparison to balance model training; data collected from the real movie data set Movielens and IMDB was mapped into a new data set for testing. Experiments show that the model has significantly improved the recommendation accuracy, and can better explain the reasons behind recommendations.

Keywords-component;Knowledge Graph;Neural Ne-twork; Recommendation Algorithm;

I. INTRODUCTION

Traditional recommendation techniques, such as collaborative filtering algorithms, are often affected by the sparsity of user-item interactions and cold start issues, which is common in some online shopping scene with a huge number of items.The KG provides an effective way for the design of recommendation systems in a big data environment. As an emerging type of auxiliary data, it can effectively solve data sparsity and cold start problems, thus improving the accuracy, diversity, and interpretability of recommendation results.

Despite the above advantages, using of KG in recommendation systems is still quite challen1ging due to the high dimensions and heterogeneity of KG. A feasible method is to pre-process the KG through the Collaborative Knowledge Graph Embedding (CKE) algorithm[3], so that can lean the potential information of the project better with the help of the KG[2]. However, the CKE algorithm only considers the user-item scoring matrix information and ignores the information of the item itself, so it cannot make a recommendation by using the Semantic Path in the KG; another method is Recurrent Knowledge Graph Embedding (RKGE)[5] . The RKGE distinguishes the importance of different paths by using the pooling operation so that it can obtain the user's main preference information. However, the semantic information of all paths are not comprehensively considered, so it cannot obtain the user's true preference information.

Therefore, in order to make better recommendations for users, with the semantic information of all paths was comprehensively considered using KG, RNN[1] and Attention Mechanism[4], a more effective recommendation model─Recurrent Knowledge Attention Network (RKAN) was proposed.

II. RESEARCH STATUS OF RECOMMENDATION SYSTEM

Early methods of introducing KG into recommendation systems fall into two categories:feature-based recommendation algorithms represented by LIBFM[10]and meta-path-based recommendation algorithms represented by HeteRec[8]. Then came the graph-based recommendation algorithms represented by HeteRs[11] algorithm and GraphLF[7]algorithm. However, these graph-based recommendation algorithms only consider the topological structure of the graph, and do not consider the modeling of entities and entity relationships in the KG. In contrast, the current KG-based recommendation algorithms represented by CKE[3] and RKGE[5] can make the most of KG to help with recommendation.

III. RECURRENT KNOWLEDGE ATTENTION NETWORK

Given user-item interaction data, this paper seeks to use the auxiliary information in the KG which can help user-item to learn high-quality representations, and then use it to generate better recommendations. In order to achieve this goal, this paper proposes a Recurrent Knowledge Attention Network (RKAN).

Notations. Let U= $\{u_1,u_2,\ldots,u_m\}$, V = $\{v_1, v_2,\ldots, v_n\}$ and $\xi = \{e_1, e_2, \ldots, e_k\}$ denote the sets of user, item and entity, respectively. If r_{ij}= 1, means u_i prefers v_j and 0 otherwise.

978-1-7281-8145-5/20/$31.00 ©2020 IEEE
DOI 10.1109/ICEDME50972.2020.00153

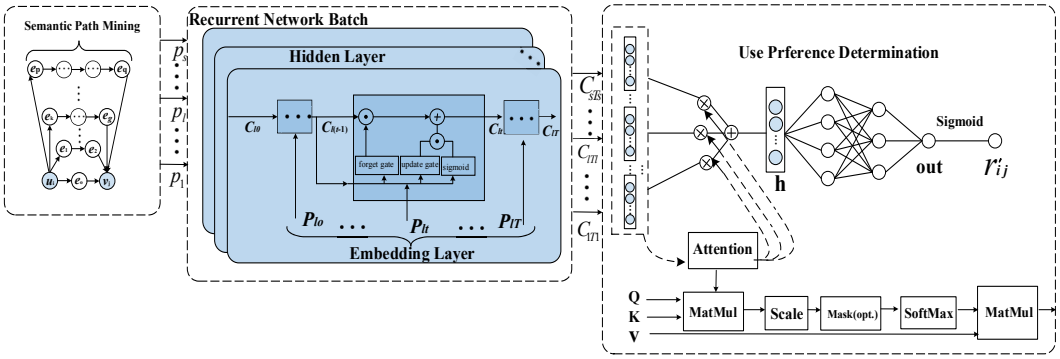

Figure 1. The overall framework of RKAN.

A Semantic Path

RKAN will first find out all the positive paths between the entity pairs from the KG. These different paths usually represent the user's different interest preferences.The path length of the entity pair (u_i, v_j) is dynamic. Path p_l of length T can be expressed as : $p_l = e_0 \rightarrow e_1 \rightarrow e_2 \rightarrow, ..., \rightarrow e_T$ with $e_0 = u_i$, $e_T = v_j$. The directed paths between entity pairs are regarded as sequences, and the elements of which are the entities in the path. The Long Short-Term Memory (LSTM) can model sequences of various lengths and learn the semantic representation of each entity and the representation of the entire path.

Because there are a large number of paths connecting entity pairs in the KG, in order to improve the model efficiency, this paper designs two strategies to help to choose the path：

(1) Paths with a length constraint, i.e. only paths with length less than the threshold.

(2) Considering that the goal of this article is to recommend items to users, so only the user-to-item path is considered. This article will also study how path length affects recommendation performance in the KG and verify our guesses in experiments. When RKAN finds qualified paths with different semantics, these paths will be processed by the recurrent network.

B Recurrent Network Batch

The S directed paths between the entity pair(u_i, v_j) as $p(e_i, e_j) = \{p_1, p_2, ..., p_s\}$ was sent to the corresponding recurrent neural network, and which consists of two network layers: the embedding layer and the hidden layer. Note that the number of paths S is dynamic, so the number of recurrent networks in a batch will change according to the number of paths.

Embedding Layer. For each entity e_t in the path, the embedding layer learns a distributed representation p_{lt}, This leads to the representation of the path as $\{p_{l0}, p_{l1}, ..., p_{lT}\}$, this new representation is then provided as input to the hidden layer.

Hidden Layer. The hidden layer contains a batch of LSTM structures. Each LSTM network encodes the sequence of the start entity to the end entity so it can get the semantic of the specified path, which was represented by C_{lT}. Update and forget gates are used to control the flow of information in the path. The hidden layer is modeled at time t as：

$$C_{lt} = \Gamma_u * C'_{lt} + \Gamma_f * C_{l(t-1)} \qquad (1)$$

Γ_u is the update gate, Γ_f is the forget gate, $C_{l(t-1)}$ is the hidden state at the previous moment, C'_{lt} is the current candidate hidden state. In addition, all RNNs in a batch share the same parameters, so as to avoid over fitting. Finally, we obtain the hidden representations of all paths, and then distinguish the importance of these paths through a method based on attention mechanism[4].

C Attention-Based User Preference Determination

For the hidden representations of all paths learned through a recurrent network batch, an attention-based method was used to converge all hidden states,and then the importance of different paths was described through different weights, and finally obtains the user's true interest prefe-rence.Specifically, the hidden state tensors $C_{1T_1}, C_{2T_2} \cdots\cdots C_{sT_s}$ of each learned path were stitched together to form a multi-dimensional tensor D, and then the Attention mechanismwas used to calculate weight matrix. The three input matrices Q(Query),K(Key),V(Value) are D：

$$\text{Attention}(\text{Q}, \text{K}, \text{V}) = \text{softmax}\left(\frac{QK^T}{d_k^{-1}}\right)V \qquad (2)$$

The weight matrix describes how much the hidden state of each path contributes to the hidden state h of the final aggregation. So the result of the hidden state h of the final aggregation is：

$$h_j = \sum_{i=1}^{s} W_{ij} * D_{ij} \qquad (3)$$

h_j is the value of jth dimension of h. The fully connected layer is then used to further quantify the relationship (closeness) between u_i and v_j, that is,r'_{ij}, given by：

$$r'_{ij} = \text{f(h)} = \sigma(W_r \cdot h + b_r) \qquad (4)$$

W_r is the regression coefficient, b_r is the bias term. We use sigmoid function to control r'_{ij} to [0,1].

After the model training is completed, all paths between entity pairs can be encoded by RKAN to better represent u_i and v_j. According to[3], during the test, the closeness score bet-ween u_i and v_k is calculated by its inner product of the embedding vectors, i.e., $s(u_i, v_k) = < u_i, v_k >$. Finally, this article ranks items based on proximity scores and recommends the top Kitems with the highest scores to users.

D Model Optimization

Given the training data train, RKAN learns the involved parameters by minimizing the loss function:

$$J = \frac{1}{|train|}\sum_{r_{ij}\in \text{train}} BELoss(r_{ij}, r'_{ij}) \qquad (5)$$

Algorithm1:RKAN Optimization

- Built the graph G with python and map the auxiliary infomation into ID;
- Sample negative movies for each user to balance the model training;
- Extract paths for both positive and negative user-moive interaction;
- Feed both postive and negative path into the Recurrent Neural Network based on Epu(1);
- User preference determination based on attention based on Epu(2-4);
- Update parameters by back propagation a-lgorithm;

BCELoss represents the binary cross-entropy loss function, and r_{ij}, r'_{ij} represent the observed score and the estimated score, respectively. In this paper, the recommendation problem was solved as a binary classification problem [6]. To balance model training, each user's unrated items was randomly sampled to generate a series of negative samples. According to formula (5), The model is trained end-to-end, and then the parameters are updated by a back-propagation algorithm.

IV. Experiment And Analysis

A Experimental Setup

- Data Set

In order to test the validity of the model, a real data set was used for the research. The data set is constructed by combining Movielens 1M and the corresponding IMDB data set. Movielens 1M is a personalized movie rating data set, including 6040 users and 3706 movies, as well as user rating data for movies; the IMDB data set contains some auxiliary information about the movie, such as genre, director, actors, etc. The two datasets are linked to obtain the experimental data. (Dataset download addresses are: groplens.orgdatatsets/movielens; www.imdb.com).

According to[8], the data set was processed like this: if the user rates the movie, the feedback is set to 1, otherwise it is set to 0. The data was split in the order of time stamps, the previous 80% of the feedback data was used as training set and the latest 20% was used as test data.

- Comparison Method

In order to verify the effectiveness of the model, the RKAN model was compared with BPRMF[9]、LIBFM[10]、NCF[6]、HeteRs[11]、HeteRec[8]、GraphLF[7]、CKE[3]、RKGE[5] algorithm.

- Evaluation Metrics

We adopt Precision at N = {1,5,10} [7], i.e.,Pre@N,and the top-N Mean Reciprocal Rank (MRR), as evaluation metrics.

B RKAN Results

Table 1 summarizes the performance of all comparison methods on the real data set. "All Users" indicates that all users are considered in the test data; and "Cold Start" indicates that only users with less than 5 ratings are considered in the test data.

TABLE 1: PERFORMANCE OF ALL COMPARISON APPROACHES IN ALL EVALUATION INDICATORS.

	All Users				**Cold Start**			
	Pre@1	***Pre@5***	***Pre@10***	***MRR***	***Pre@1***	***Pre@5***	***Pre@10***	***MRR***
BPRMF	0.0409	0.0438	0.0441	0.1234	0.0171	0.0191	0.0205	0.0438
LIBFM	0.0459	0.0525	0.0456	0.1412	0.0330	0.0203	0.0273	0.0457
NCF	0.0450	0.0482	0.0485	0.1360	0.0188	0.0210	0.0225	0.0481
HeteRS	0.0689	0.0528	0.0475	0.1600	0.0405	0.0428	0.0370	0.1239
HeteRec	0.0764	0.0579	0.0488	0.1737	0.0573	0.0428	0.0402	0.1355
GraphLF	0.1069	0.0360	0.0581	0.1524	0.0677	0.0267	0.0422	0.1188
CKE	0.0954	0.0781	0.0682	0.2440	0.0687	0.0432	0.0372	0.1408
RKGE	0.1396	0.1092	0.0861	0.3056	0.0809	0.0481	0.0467	0.1521
RKAN	0.1595	0.1219	0.0951	0.3346	0.0905	0.0526	0.0510	0.1644
Improve	14.25%	11.63%	10.45%	9.49%	11.86%	9.36%	9.21%	8.09%

- Performance on All Users

Compared with other algorithms, the proposed model achieves the best performance. The average improvement of Precision is 12.11%, and the improvement of MRR is 9.49%.

- Performance on Cold Start

In "cold start", the model in this paper still outperforms other methods. The average improvement of Precision is 10.14%, and the improvement of MRR is 8.09%. It was shown that most methods are susceptible to cold starts because these methods learn user preferences based on the user's historical behavior, when, however, there's very little about the user's historical behavior information,the accuracy rate will have varying degrees of decline. But CKE, RKGE, and RKAN in this paper are always better than other methods, which shows that data sparseness and cold start in the

traditional recommendation system could be solved by embedding KG. And RKAN performance is better than RKGE and CKE, which shows that user's preference can be captured by RKAN in a more effective way.

- Effect of Path Length

Paths with different lengths were filtered from all paths between user-items,i.e.,L={3,5,7}, and then which were sent to the recurrent network for further processing. Figure 3 shows the results: as the path length increases, the accuracy rate gradually decreases. This fully shows that too long paths have more noise, and shorter paths have clearer semantics.

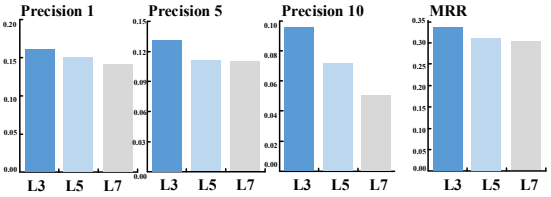

Figure 2.Influence of different path lengths of RKAN.

C *Example Test*

RKAN not only provides better recommendations, it is also interpretable.

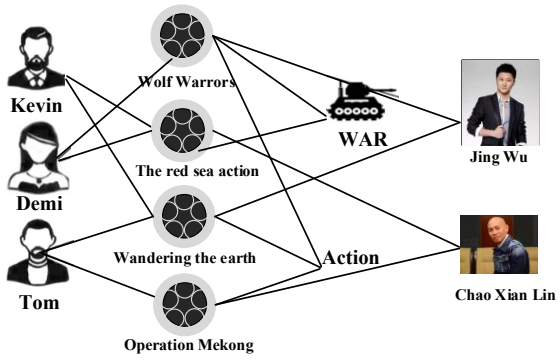

Figure3. A KG in the movie field.

a)Kevin ⟶ *The RedSea Action* ⟶ *War Movies* ⟶ *WolfWarriors II*

b)Kevin ⟶ *Wandering The Earth* ⟶ *Jing Wu* ⟶ *WolfWarriors II*

To prove this, take a KG-based movie recommendation system as an example: each user's rating item and the correctly recommended item were first mapped into KG, and then whether there is a semantic relationship between the two was checked.The semantics described in path (a) above is that Kevin may be interested in the same movie type; the semantics described by path (b) is that Kevin likes movies starring the same actor. Correctly recommended movies(Wolf WarriorsII) are connected to Kevin's rated movies by genre (War Movie), actor (Wu Jing). This shows that RKAN can infer that Kevin likes War Movies or actor Wu Jing, thereby providing users with interpretable recommendations.

V. CONCLUSION

The recommendation system is of great signif-icance for screening effective information and im-proving the efficiency of information acquisition. This paper proposes a model based on KG embe-dding, RKAN, which combines a RNN, KG, and an Attention Network for high-quality recommend-ation. It is widely verified on real data sets that RKAN outperforms existing models. However, this model still needs to be improved, for example, t-he accuracy rate still has a huge room for impro-vement. This also proposes new ideas for future s-tudy.

ACKNOWLEDGEMENT

This work was supported by National Natural Science Foundation of China(NO.61402069),Natio-nal key research and development plan(NO.217Y-FC0821003),Natural Science Foundation of Liaoni-ng Province(NO.20180550395).

REFERENCES

[1] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcu-oglu, and Pavel Kuksa. 2011. Natural Language Processing (almost) from Scratch. Journal of Machine Learning Research (JMLR) 12,Aug(2011),2493–2537.

[2] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo.2018.DKN:Deep Knowledge-Aware Network for News Recommendation. In Proceedings of the2018 World Wide Web Conference on World Wide Web.1835–1844.

[3] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge base Embedding for Recommender Systems.In Proceedings of the 22nd ACM SIGKDD International Conference on Knowle-dge Discovery and Data Mining (KDD).ACM,353–362.

[4] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan -Salakhudinov, RichZemel, and YoshuaBengio.2015. Show, Attendand Tell:Neural Image Caption Generation with Visual Attention.In International Conference on Machine Learning (ICML).2048–2057.

[5] ZhuSun , JieYang , JieZhang , Alessandro Bozzon , Long-Kai Huang, and ChiXu. 2018. Recurrent knowledge graph embedding for effective recommendation.In RecSys.297–305.

[6] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng -Chua. 2017. Neural Collaborative Filtering. In Proceedings of the 26th International Conference on World Wide Web (WWW).International World Wide W-eb Conferences Steering Committee,173–182.

[7] Rose Catherine and WilliamCohen.2016. Personalized Recommendations using K-nowledge Graphs:A Probabilistic Logic Programming Approach.In Proceedings of the 10th ACM Conference on Recommender Systems (RecSys).ACM,325–332.

[8] XiaoYu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khan-delwal , Brandon Norick , and Jiawei Han.2014.Personalized Entity Recommendation: A Heterogeneous Information Network Approach. In Proceedings of the 7th ACM International Conferenceon WebSearch and Data Mining(WSDM).ACM,283–292.

[9] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR:Bayesian Personalized Ranking from Implicit Feedback.In Proc-eedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI).AUAIPress, 452–461.

[10] Steffen Rendle. 2010. Factorization Machines.In Data Mining (ICDM), 2010 IEEE 10th International Conference on.IEEE,995–1000.

[11] Tuan-Anh Nguyen Pham, Xutao Li, Gao Cong, and Zhenjie Zhang. 2016. A General Recommendation Model for Heterogeneous Networks. IEEE Transactions on Knowledge and Data Engineering (TKDE) 28,12(2016),3140–3153.