A Recommendation Algorithm Focusing on Time Bias via Neural Graph Collaborative Filtering

Simin Li, Yaping Dai, Kaoru Hirota, Wei Dai*

School of Automation, Beijing Institute of Technology, No. 5 Zhongguancun South Street, Haidian District, Beijing 100081, China
State Key Laboratory of Intelligent Control and Decision of Complex Systems, Beijing, 100081,China
*E-mail: wei.dai@riversecurity.com

Abstract. To reduce the error caused by time bias to the accuracy of recommendation algorithm, based on neural graph collaborative filtering (NGCF), a NGCF-Shallow tower recommendation algorithm is proposed. By inputting the users' and items' interaction social networks and time features, the recommended objects for a target user is obtained. Compared with other recommendation structures, this algorithm can make up for the influence of users' and items' time characteristics on users' selection. The effect of the algorithm is tested on the Gowalla dataset and the Hetrec2011-delicious-2k dataset.Compared with NGCF algorithm, the accuracy of the proposed algorithm is improved by about 1.5% on Gowalla dataset. In the experiment under the Hetrec2011-delicious-2k dataset, the accuracy rate can be increased up to 4.42%.

Keywords: Recommendation Algorithm, Time Bias, Neural Graph Collaborative Filtering

1. INTRODUCTION

With the advent of big data, various recommendation algorithms have emerged to help users find their most interesting content from massive data, as well as to help businesses to maximize their benefits. Among these algorithms, Sarwar et al. proposed a collaborative filtering algorithm , and they calculated the similarity between objects and then generated a recommendation list [1]; Lops et al, proposed a content-based recommender system that recommends products similar to the past preferences of the specified user [2];Based on the collaborative filtering algorithm, Zhao et al. added a multitask ranking module and proposed the Multitask Ranking System, which can adjust the weight of each task to make the recommended results fit the application scenario [3]; Wang et al. proposed a deep learning method using graph convolutional neural networks, and they calculated the relationship between users and items to obtain the recommendation list [4]. However, the computational complexity of the algorithms based on collaborative filtering is very high, which makes them difficult to apply to large data sets [5]. The algorithms based on deep learning mainly calculate the recommender list through user behavior, and pay little attention to the impact from users' and items' characteristics [6].

Therefore, considering the main factors that affect users' selection in practical application scenarios, a NGCF-Shallow tower algorithm is proposed. Compared with the collaborative filtering recommendation algorithms, the algorithm based on graph convolutional neural network can operate on a large amount of data. In most recommendation scenarios, users are usually more interested in other more active users and popular items, so the normal deep learning model usually forms a closed feedback loop. A shallow tower structure is designed to to make up for this deficiency. By fitting the impact of users' and items' time characteristics on user selection, the proposed NGCF-Shallow tower algorithm is able to improve the accuracy of the recommender estimation in time-sensitive application sites.

The proposed NGCF-shallow tower recommendation algorithm mainly includes two modules: recall module and rank module. The recall module uses the main structure of NGCF to learn the embedding representation of users and items through high-order connectivity relationships. Then joint the embedding of user(or item) in each layer together to form their final embeddings without time bias.

In rank module, a shallow tower structure is used to fit the time bias individually. The shallow tower uses the linear logistic regression to calculate the linearly distributed scalars for each user's and item's time feature. After adding the output of shallow tower one-to-one to the final embedding representations from recall module, the score for ranking can be obtained by using inner product. Finally, based on the score obtained in the rank module, a recommendation list is output according to the set number of recommended objects.

The effect of algorithm is verified on Gowalla and Hetrec2011-delicious-2k dataset. Gowalla dataset is provided by a location-based check-in social website. The Hetrec2011-delicious-2k dataset is provided by the delicious social bookmarking system. Compared with NGCF algorithm, the accuracy of the proposed algorithm is improved by 1.5% on average under Gowalla dataset. In the experiment on the Hetrec2011-delicious-2k dataset the accuracy rate can be increased up to 4.42%.

Section 2 introduces the structure of the proposed recommendation algorithm NGCF-Shallow tower. The implementation and experimental verification of the algorithm are described in section 3.

2. NGCF-SHALLOW TOWER ALGORITHM'S STRUCTURE

In this part, the overall structure design of the algorithm is introduced and the basic theory of the two modules are elaborated.

2.1. The Algorithm's Structure

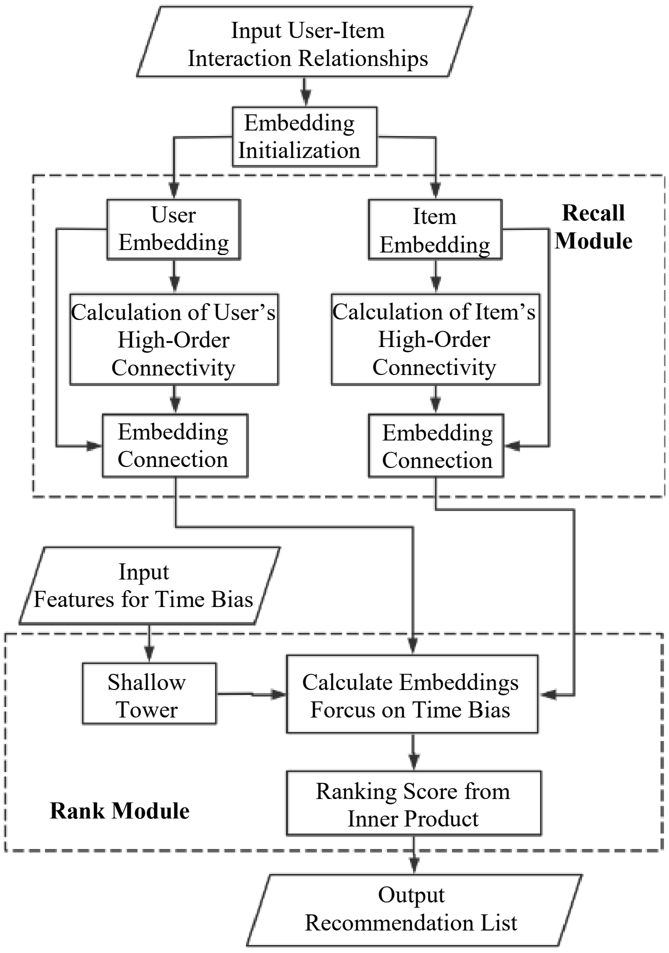
Input User-Item Interaction Relationships
Embedding Initialization
User Embedding
Item Embedding
Recall Module
Calculation of User's High-Order Connectivity
Calculation of Item's High-Order Connectivity
Embedding Connection
Embedding Connection
Input Features for Time Bias
Shallow Tower
Calculate Embeddings Forcus on Time Bias
Ranking Score from Inner Product
Rank Module
Output Recommendation List

Fig. 1. The overall workflow of the algorithm

The overall workflow of the algorithm is designed according to the target function as shown in Fig. 1. The user-item interaction relationships is input to obtain the initial embedding for each user and item first. Then put the initial embeddings to the recall model. The recall module uses NGCF structure to learn the final embedding representation of users and items through high-order connectivity. Input the features of time bias to the shallow tower to fit the bias with linearly distributed scalars.Add the scalars from shallow tower to the final embedding representation from NGCF, then use inner product to get the score for ranking. A recommendation list is output according to the number of recommended objects.

2.2. Recall Module

The recall module is used to learn the embedding vector of users and items with the input of user-item information and context information. Embedding process is to map an ID class feature or discrete feature into an n-dimensional vector [7]. In the training process, the embedding is trained as a variable like the weight of model. [8]

In the proposed algorithm, the recall module uses embedding training part in NGCF structure.The full structure of NGCF is shown in Fig.2.

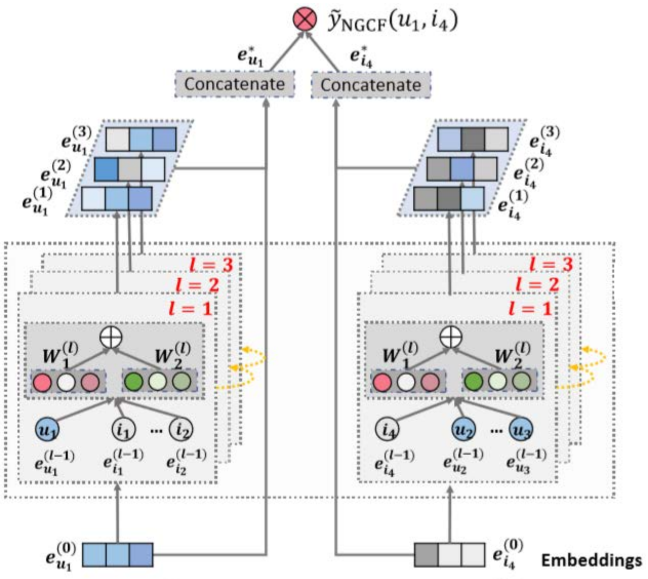

Fig. 2. Illustration of NGCF model architecture(the arrowed lines present the flow of information).The representations of user u_1(left) and item i_4(right) are refined with multiple embedding propagation layers, whose outputs are concatenated to make the final prediciton.[4]

Firstly, initialize the state for user embeddings e_u and item embeddings e_i as shown in Eq. (1), according to the input users' and items' information. It can be initialized randomly or pretrain according to known characteristics. The initialization result can be seen as building a parameter matrix as an embedding look up table.

$$E = [e_{u1}, \cdots, e_{uN}, e_{i1}, \cdots, e_{iM}]$$
$$e_u, e_i \in R^d, E \in R^{(N+M)\times d} \tag{1}$$

Then aggregate interactive users and items to update user(or item) embedding according to user-item interaction relationship data. For interactive user u and item i, define the embedding propagation formula as shown in Eq.(2). Among the formula, N_u and N_i are the set of the first-hop neighbors of user u and item i, "⊗" is the multiplication between elements, W_1 and W_2 are trainable paramenters. $1/\sqrt{|N_i||N_u|}$ reflects how much the historical item contributes the user preference, and considering the messages being propagated should decay with the path length, $1/\sqrt{|N_i||N_u|}$ can be interpreted as a discount factor.

$$m_{u\leftarrow i} = \frac{1}{\sqrt{|N_i||N_u|}}(W_1 e_u + W_2(e_u \otimes e_i)) \tag{2}$$
$$W_1, W_2 \in R^{\hat{d}\times d}, m_{u\leftarrow i} \in R^{\hat{d}}$$

Similarly, define the embedding propagation formula between interactive users and users as shown in Eq.(3).

$$m_{u\leftarrow u} = W_1 e_u \tag{3}$$

Taking user u as an example, an aggregation function is defined as shown in Eq.(4). The formula is used to aggregate messages propagated from u's neighborhood to refine u's embedding representation.

$$e_u^{(1)} = Leaky\,\mathrm{ReLU}(\mathrm{m}_{u\leftarrow u} + \sum_{i\in N_u} \mathrm{m}_{u\leftarrow i}) \tag{4}$$

where $e_u^{(1)}$ denotes the representation of user u obtained after the first embedding propagation layer. The activation function of LeakyReLU allows messages to encode both positive and small negative signals. According to high-order connectivity, a user or item can receive information from its farther neighbors in each layers [9]. Therefore, substitute Eq.(2) and Eq.(3) into Eq.(4), the aggregation function of a certain layer l is as shown in Eq.(5).

$$e_u^{(l)} = Leaky\,\mathrm{ReLU}(W_1 e_u^{(l-1)} + \sum_{i\in N_u} \frac{1}{\sqrt{|N_i N_u|}} (\mathrm{W}_1 \mathrm{e}_u^{(l-1)} + \mathrm{W}_2(\mathrm{e}_u^{(l-1)} \otimes \mathrm{e}_i^{(l-1)}))) \tag{5}$$

Analogously, the representation $e_i^{(l)}$ is obtained for item i by propagating information from its connected users.

Splice the embedding of user of item in each layer obtained by Eq.(5), the final embedding is achieved as shown in Eq. (6)

$$e_u^{'} = [e_u^1, e_u^2, \cdots, e_u^l] \tag{6}$$
$$e_i^{'} = [e_i^1, e_i^2, \cdots, e_i^l]$$

2.3. Rank Module

One challenge in recommendation algorithm, similar to the general search ranking problem, is to achieve both memorization and generalization. Memorization can be loosely defined as learning the frequent co-occurrence of items or features and exploiting the correlation available in the historical data. Generalization, is based on transitivity of correlation and explores new feature combinations that have never or rarely occurred in the past [10]. In NGCF algorithm, the score of user u and item i is simply the inner product of their final embeddings.However, the deep neural network model used in recall module has generalization capabilities, but in the case of sparse data, there is also a situation that it is too "generalized" and sometimes recommend irrelevant items.

In the proposed algorithm, the final embeddings is input into the rank module to take the influence of time bias into consideration. The existing logistic regression model has strong memoryization capabilities. Therefore, to combine the benefits of memorization and generalization for the recommendation algorithm, the rank module uses the linear logistic regression as shallow tower structure to joint wide linear models and deep neural networks.

The proposed algorithm uses the activity time stamp of users and items as their time feature. According to Fig.2, the users' and items' final embeddings and their time feature are input into the rank module. The shallow tower learns the users' and items' time feature and output their linearly distributed scalars.Then the scalars is added to the final embeddings one-to-one based on tensor broadcasting principle.After that, the embeddings forcusing on time bias are obtained.The score used for ranking of user u and item i is shown in Eq.(7):

$$\hat{y}(u,i) = e_u^* \times (e_i^*)^T \tag{7}$$

where e_u^* means the users' embeddings with time bias and e_i^* means the items'.

The parameters are updated by loss function shown in Eq.(8):

$$\mathrm{Loss}= \sum_{(\mathrm{u,i,j})\in \mathrm{O}} -In\sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda \,\|\, \theta \,\|_2^2 \tag{8}$$

where (u,i,j) ∈ O means user u and item i have interactions, while user u and item j have no interaction. σ represents the sigmoid function. θ denotes all trainable model parameters,and λ controls the L2 regularization strength to prevent overfitting.

After the training, for specified users, the items in dataset are ranked according to their scores obtained from Eq.(7). A recommender list is finally output in the light of the set number of recommended objects.Compared with the users' real choice, the accuracy of the proposed NGCF-Shallow tower algorithm can be calculated.

3.EXPERIMENTAL VERIFICATION AND RESULT ANALYSIS

The NGCF model used in recall module performs best when it has three layers[4]. That means only three hops neighbors' embeddings of users and items is involved in the high-order connectivity calculation to refine the embedding representation.

There are two datasets: Gowalla set and Hetrec2011-delicious-2k set are used for experimental verification in this algorithm.

Gowalla set is a location-based social networking website where users share their locations by checking-in. The friendship network is undirected and was collected using their public API, and consists of 196,591 nodes and 950,327 edges. A total of 6,442,890 check-ins of these users are colected over the period of Feb. 2009 - Oct. 2010.

The Hetrec2011-delicious-2k dataset contains social networking, bookmarking, and tagging information from a set of 2K users from delicious social bookmarking system.The set consists of 71,090 nodes and 104,830 interactions and their timestamp.

3.1. Experiment under Gowalla Gataset

In order to prove the effectiveness of the proposed algorithm, the NGCF algorithm is reproduced under Gowalla set firstly.The best result from NGCF experiment under Gowalla set given in the paper [4] is recall and ndcg data, when predicting top 20 objects in recommender list. The result of reproducing experiment with the known parameters shown in the paper [4] are as shown in Table 1.

Table. 1 Reproduction results of NGCF under Gowalla dataset (experiment results from [4] are in red)

	Recall@20	**Recall@40**	**Recall@60**	**Recall@80**	**Recall@100**
Recall	**0.15361** 0.1547	0.21647	0.26218	0.29972	0.32858
Precision	0.04699	0.03343	0.02718	0.02344	0.02075
Hit	0.53527	0.64301	0.70326	0.74479	0.77172
NDCG	**0.22651** 0.2237	0.26511	0.28988	0.30888	0.32339

Table. 2 Results of NGCF – Shallow tower algorithm under Gowalla dataset

	Recall@20	**Recall@40**	**Recall@60**	**Recall@80**	**Recall@100**
Recall	**0.15586**	0.21935	0.26577	0.30299	0.33502
Precision	0.04783	0.03396	0.02768	0.02378	0.02113
Hit	0.54016	0.64646	0.70597	0.74600	0.77708
NDCG	**0.22696**	0.26544	0.29055	0.30922	0.32472

By adjusting the weight ratio of the shallow tower's output to a appropriate interval, the experiment result of the proposed NGCF-shallow tower is shown in Table 2. The recall of 20 objects is 0.15586, and ndcg is 0.22696. Both performs better compared with NGCF algorithm.

Compared with the data given in the original NGCF paper [4], the proposed algorithm output better estimation result with 0.75% and 1.46% higher in recall and ndcg for 20 objects.

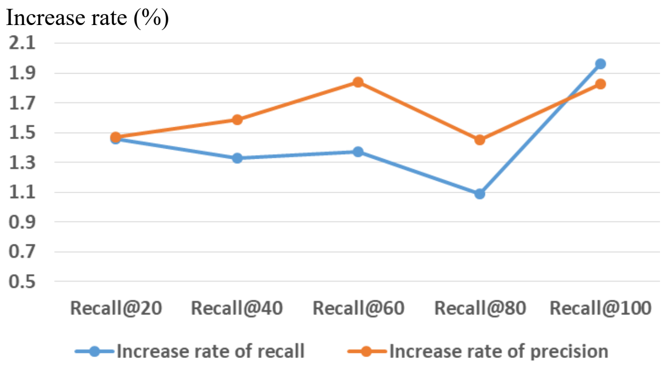

Fig. 3. The increase rate of proposed algorithm compared with NGCF

Choose recall and precision as evaluation indexes [11], then compared the results from NGCF reproduction with NGCF-Shallow tower algorithm experiment. As shown in Fig.3, the proposed algorithm achieved better result in all set project number in both recall and precision field.The increase rate improved by about 1.5% in average than NGCF.

3.2. Experiment under Hetrec2011-delicious-2k Dataset

First, carry out the experiment under the Hetrec2011-delicious-2k dataset by using NGCF as a base line.

Then use the proposed NGCF-Shallow tower algorithm for a comparison experiment.The accuracy is always lower than the NGCF result at the beginning. Try to reduce the weight ratio of shallow tower's output in the embeddings, it is obversed that the weak shallow tower reduced the accuary of recommender estimation still further. It is considered that when the weight value is not strong enough to have an impact, it may become similar to a little interference with its ranking process. In order to prove this suspect, keep the weight value constant and disrupt the time bias to add a real tiny noise in. In this case, the scalars output by shallow tower is not added to the final embeddings one-to-one, but combined randomly bewteen the projects. Compared with the result from NGCF and NGCF with weak shallow tower, the accuracy of NGCF with noise reduced further.

These comparsion experiments shows that adding a shallow tower to the NGCF algorithm to take time bias into consideration can have a positive effect on the results, but the extent of the impact and the specific weight of shallow tower depend on the situation of the data set or desired guiding direction of the use background. That means the desired degree of time sensitivity of the recommender item matters.

In subsequent experiments, by adjusting the weight ratio of shallow tower to a appropriate interval, the proposed algorithm output better performance than NGCF, which proved the above dissertation. The comparison of results and precision of each prediction number with different weights of shallow tower is shown in Fig.4 and Fig.5.

According to the figures, the algorithm archieve the best result when the recommender objects is set to 20.The maximum improvement under recall index is 4.42% and the up to 3.69% under precision index. The experiments under the two datasets proved that by adding the time forcusing shallow tower, the proposed NGCF-Shallow tower algorithm is factually more effective than NGCF algorithm. And according to the experiment under the Hetrec2011-delicious-2k dataset, it can be concluded that by changing the weight of the shallow tower, the sensitivity of the recommendation algorithm to the time characteristics can be adjusted to adapt to multiple application requirements.

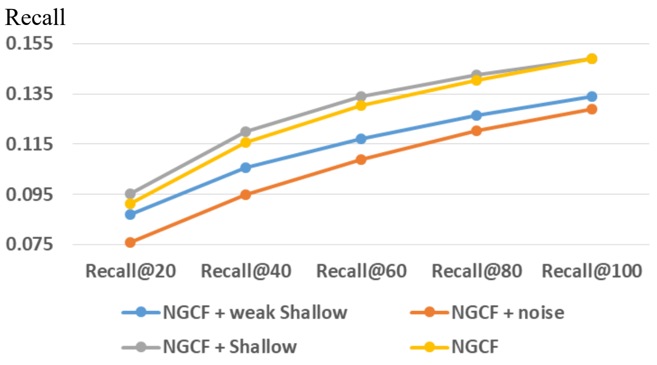

Fig. 4. Comparison of recall results of each prediction number under different weights

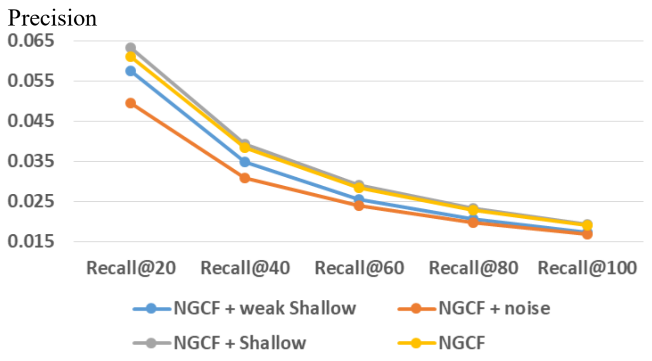

Fig. 5. Comparison of precision results of each prediction number under different weights

4. CONCLUSION

To reduce the error caused by time bias to the accuracy of recommendation algorithm, a NGCF-Shallow tower structure based on neural graph collaborative filtering is proposed. By inputing the users' and items' interaction social relationships and time features, the recommender objects for target user is obtained. In the proposed algorithm, the NGCF is used to learn the embedding representation of users and items through high-order connectivity. The embedding of user(or item) in each layer is jointed together to form their final embedding. The shallow tower is used to calculate the linearly distributed scalars for each user and item. Add the scalars one-to-one to the final embedding representation from NGCF, then use inner product to get the score for ranking. A recommendation list is output according to the set number of recommended objects.

Compared with collaborative filtering recommender algorithms, the proposed algorithm is based on graph convolutional neural network, so it can operate on a large amount of data. On the other hand, a shallow tower structure is designed to fit the impact of users' and items' time characteristics on user selection, so the proposed NGCF-Shallow tower algorithm is able to improve the accuracy of the recommendation estimation in time-sensitive application sites.

The effect of algorithm is verified on Gowalla dataset and Hetrec2011-delicious-2k dataset. Compared with NGCF algorithm, the accuracy of the proposed algorithm is improved by about 1.5% on Gowalla dataset. In the experiment on Hetrec2011-delicious-2k dataset, the accuracy rate can be increased up to 4.42%.

In future studies, we will try to use more effective algorithms to improve the shallow tower.The proposed algorithm only uses the activity time feature of users and items as the shallow tower's input, then obtain linearly distributed scalars.However, linear structure is difficult to describe more complex time feature information. Moreover, the scalar output of shallow tower is simply added with embedding. Therefore, at the initial stage of training, it may affect the convergence process of the recall module, and finish the training ahead of time.To take those shortcoming into consideration, a nonlinear shallow tower structure is needed, and the shallow tower's output should be added after the recall module converges stably.Through refining these details, the algorithm will provide more accurate and reasonable recommendation estimation.

Acknowledgement: This work was supported in part by the Thousand Talents Plan (the State Recruitment Program of Global Experts) under Grant No. WQ20141100198, in part by the Beijing Municipal Natural Science Foundation under Grant No. 3192028, and in part by the Beijing Municipal Natural Science Foundation under Grant No. L191020

REFERENCES:

[1] M.D. Ekstrand, J.T. Riedl and J.A. Konstan, "Collaborative filtering recommender systems," Foundations and Trends in Human-Computer Interaction, vol.4, no.2, pp.81-173,2011.

[2] P. Lops, M.D. Gemmis and G. Semeraro, "Content-based recommender systems: State of the art and trends," Recommender systems handbook. Springer, pp.73-105.2011.

[3] Z. Zhao, L. Hong, L.Wei and et al, "Recommending what video to watch next: a multitask ranking system," Proceedings of the 13th ACM Conference on Recommender Systems,pp.43-51,Sep.2019.

[4] X. Wang , X. N. He, M. Wang, F. L.Feng and T. S. Chua, "Neural Graph Collaborative Filtering," 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, Jul. 2019.

[5] G.H.Sun, D.Q.Liu and M.Y.Li, "A Survey of Personalized Recommendation Algorithms," Computer Engineering and Software, vol. 38, no. 7, pp. 70-78, 2017.

[6] X.N. He, L.Z. Liao, H.W. Zhang, L.Q. Nie, X. Hu and T.S Chua, "Neural Collaborative Filtering," International World Wide Web Conference, Perth, Australia, April. 2017

[7] P.Goyal and E. Ferrara, " Graph Embedding Techniques, Applications, and Performance: A Survey," Knowledge Based Systems, no.151(JUL.1), pp.78-94,2017.

[8] P. Covington, J.Adams and E.Sargin, "Deep Neural Networks for YouTube Recommendations," Acm Conference on Recommender Systems. ACM, pp. 191-198, 2016.

[9] R. Ying, R. He, K.F. Chen and et al, "Graph Convolutional Neural Networks for Web-Scale Recommender Systems," 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, London, United Kingdom, Aug. 2018.

[10] H.T. Cheng, L. Koc, J. Hermasen and et al, "Wide & Deep Learning for Recommender Systems," Deep Learning for Recommender Systems, Boston, USA, Sep.2016.

[11] M.F. Dacrema, P. Cremonesi and D. Jannach, "Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches," In Thirteenth ACM Conference on Recommender Systems(RecSys'19), Copenhagen, Denmark, Sep.2019.