Journal of Computers Vol. 34 No. 4, August 2023, pp. 255-265
doi: 10.53106/199115992023083404022

Deep Collaborative Filtering System

Xin-Yi Wang[1*], Hao-Ran Sun[2], Xu-Yang Yin[3], Chun-Zi Li[4], Sheng-Yu Liu[4]

[1] School of Economics and Management, Beijing Jiaotong University,
Beijing 100044, China
wangxy@bjtu.edu.cn

[2] School of Electronic and Information Engineering, Beijing Jiaotong University,
Beijing 100044, China
21111054@bjtu.edu.cn

[3] School of Traffic and Transportation, Beijing Jiaotong University,
Beijing 100044, China
22125796@bjtu.edu.cn

[4] School of Zhan Tianyou, Beijing Jiaotong University,
Beijing 100044, China
21241263@bjtu.edu.cn, 21311037@bjtu.edu.cn

Received 1 July 2023; Revised 20 July 2023; Accepted 25 July 2023

Abstract. Collaborative filtering-based models can use the interaction between users and products or the correlation between users and users, and between products and products. However, methods based on collaborative filtering can only grasp one type of relationship and still cannot fully fit. Various factors influencing user preferences make a lot of redundant information still not filtered out. We proposals a collaborative filtering model based on deep learning, which combines the item-item relationship learning in advance with a neural collaborative filtering network to effectually make recommendations. In the initial stage, learn low-dimensional vectors of compartments, and embed information that reflections the co-occurrence relationship between compartments. The prediction stage combines the trained embedding vector with the embedding vector of the module as a correction to the output result of the neural network. The benchmark data set MovieLens 1M is the experienced data set of this article, and the effectiveness of this method is verified on the data set. The experienced results are compared with some advanced methods on the data set. The results show that the model proposed in this paper is better than some methods based on collaborative filtering.

Keywords: collaborative filtering, deep learning, recommender system, neural networks, implicit feedback

1 Introduction

With the rapid development of information technology such as the Internet, the amount of information is gradually increasing, and the excessive amount of information leads to too complicated information. It is difficult for people to obtain the information they want from the massive information, and the problem that effective information is difficult to filter is gradually revealed. In order to solve such problems, the technical field has explored filtering services and recommendation systems.

The main content of this paper is the recommendation system algorithm based on deep learning, which uses implicit data. Through learning to understand relevant knowledge, a collaborative filtering system based on deep learning is realized. The main contributions of this paper are:

(1) A collaborative filtering system based on deep learning is implemented, and the interaction between users and goods in collaborative filtering is realized by using multi-layer perceptron (MLP) and neural network. (2) A learning model is proposed to generate low-dimensional representations of commodities, which represent commodity-commodity embedding. Commodity-commodity embedding vectors are combined with commodity embedding vectors inputted into multilayer perceptron to obtain a component of the prediction results, and the output of multilayer perceptron and the combined results are combined together in a way similar to SVD + + to

* Corresponding Author

form the prediction results of the model. (3) The algorithm is implemented based on python by using pytorch. The model is validated on MovieLens 1M dataset and compared with some benchmark models. The experimental results show that the proposed method has certain advantages.

2 Literature Review

Geoffrey Hinton [1] proposed an initialization method for deep neural networks. The idea of collaborative filtering algorithm was first proposed by Goldberg in the early 1990s [2]. At that time, this technology was also directly applied in practice-in news recommendation. On this basis, the likelihood function and statistical Bayesian method are added to [3], which greatly improves the accuracy. Many domestic scholars pay attention to the in-depth study of specific recommendation in a certain field in order to obtain better results in a specific field, that is, to concretize the generalized model, such as modeling audio information into the auxiliary information of recommendation with neural network, so as to obtain better results in music recommendation [4].

The key of recommendation system model is users and goods. User is the object of system service, that is, the meaning of system existence, and also the provider of system information. Without the information provided by users, the system cannot provide services to users. User sets are generally denoted by U. Goods are represented by I, and the corresponding English words are user and item respectively. The rating of goods i by user u is then represented by y_{ui}, and the rating matrix is $R_{M\times N}$, where M is the total number of users and N is the total number of goods. A basic recommendation system *S*needs to include these basic elements. When making recommendation prediction, it does not need $U \times I \rightarrow R_{M\times N}$ to get the results of the whole system, but only needs to get a prediction based on the part of $u \times i$ that needs to be served in order to complete the recommendation [5].

There are several types of tasks for the recommendation system. Rating prediction is the prediction of the possible ratings of goods i by user u. Top-K is to sort the products that user u may be interested in, and recommend the top K products to users, which is of more practical significance.

Data types are roughly divided into explicit data and implicit data, corresponding to explicit feedback and implicit feedback respectively. Explicit data, such as the rating of goods i by user u, is display data. Different from display data, implicit data can't show the user's preference for goods. It is a binary (0, 1) data, which shows whether user u has interacted with goods i (such as clicks, scores, comments, etc.). In fact, most of the data are implicit data, even explicit data can be converted into implicit data, and the task of implicit data is generally commodity ranking recommendation. The convenience of data acquisition and the practicality of recommendation task make the study of implicit data more practical. This paper uses implicit feedback, although MovieLens is an explicit data set, but this paper transformed it into implicit data, so as to predict, showing the implicit nature of the display data.

At present, there are roughly two types of recommended methods: 1) Content-based methods. The recommended features of the content-based method [6] are extracted from the description of the user or product. 2) Collaborative filtering method. The interaction history between users and goods, such as the user's rating of items, is more used by the method based on collaborative filtering [7]. In modern recommendation system, collaborative filtering (CF) is very popular in recommendation system. Matrix factorization [8, 9] is the most widely used method in CF-based methods, so many researches focus on its expansion and improvement. Many previous researches have focused on improving matrix factorization, such as integrating it with neighbor-based model [9], combining it with project content model [10], and extending it to factorization machine for feature modeling. Although matrix factorization is effective in collaborative filtering, complex interactions cannot be captured by matrix factorization, because its work is only to multiply two potential vectors, which are decomposed users and goods. Restricted Boltzmann Machine (RBM) class methods [11, 12] are also based on CF method.

The most effective way to capture complex interactions is to use deep learning technology, which has been successful in speech recognition [13, 14], computer vision [15] and natural language processing, and has been recognized by people. In recent years, there are more and more methods to use deep neural networks for recommendation, especially content-based models, and many methods to apply deep learning to recommendation are proposed [4, 16].

However, it is limited to apply deep neural network to the model based on collaborative filtering. Dziugait and Roy put forward a neural network matrix decomposition model [17], which replaces the inner product in traditional matrix decomposition with feedforward neural network for evaluation, but its performance cannot compete with the most advanced methods. X. He [18] proposed a neural collaborative filtering model, which combines

matrix factorization with neural network, but it still uses the interaction between users and projects to predict, which is obviously an extension of traditional matrix factorization method.

3 Model

In the overall model of this paper, the given users and items are input into the neural network for ranking prediction. Before the neural network training, the commodity-commodity co-occurrence vector is pre-trained by the representation learning model, as shown in Fig. 1.

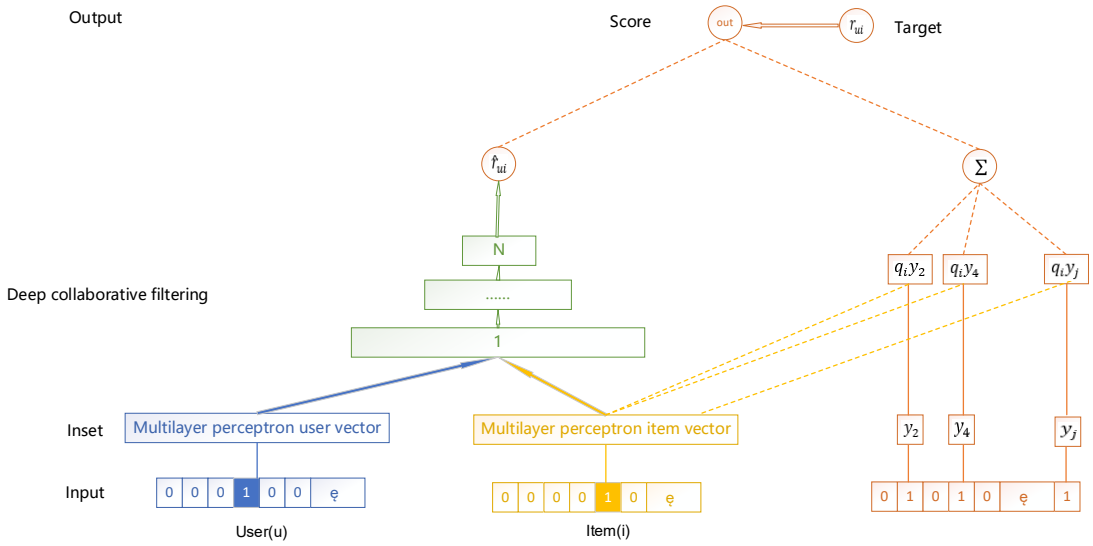

Fig. 1. The overall model of our paper

This paper argues that the similarity between products in recommendation system can be expressed by their co-occurrence. Suppose there is a recommendation system S with M commodity and N user, R_{M*N} is the rating matrix in S. Each symbol in the recommendation system S is described in Table 1. It is important to specify that, $x_i^j = \left|T_i \bigcap T_j\right|$, denotes the number of co-occurring commodities of the ith commodity with the jth commodity, which is defined here as the number of users who have evaluated these two items together, e_i is learnt in the model.

Table 1. Relevant parameter symbols of co-occurrence model

Symbol	Meaning
t_i	The *i*th product in the system
U_i	User collection of evaluated products *i*
x_i^j	Co-occurrence quantity of t_i and t_j
C	Commodity co-occurrence matrix $C_{ij} = x_i^j$
e_i	Embedding of products t_i

3.1 Model Training

In order to learn the commodity embedding of the co-occurrence matrix C, this paper adopts a matrix decomposition-like approach by representing C as:

$$C = \tilde{E}\hat{E}^T = [\tilde{e}_1, \tilde{e}_2, ..., \tilde{e}_i] * [\hat{e}_1, e_2, ..., e_i]^T. \quad (1)$$

Where $\tilde{E}$ and $\hat{E}$ are commodity embedding matrices containing different information, and $\tilde{e}_i$ and $\hat{e}_i$ are column vectors representing commodity i, which belong to $\tilde{E}$ and $\hat{E}$, respectively.

Because $x_i^j = \tilde{e}_i \hat{e}_j^T$, and the actual value of x_i^j can be obtained from matrix R_{M*N}, $\tilde{e}_i$ and $\hat{e}_j$ can be learned by minimising the difference between x_i^j and $\tilde{e}_i \hat{e}_j^T$.

Therefore, expressions that capture co-occurrence between t_i and t_j can be expressed as

$$x_i^j = \tilde{e}_i \hat{e}_j T. \quad (2)$$

Where $\tilde{e}_i$ and $\hat{e}_i$ is two vectors learned by x_i^j, and they contain different information. Here x_i^j, the actual number of users who have jointly evaluated the goods i and j the goods is indicated.

Then the information of the entire product i can be represented as

$$e_i = [\tilde{e}_i, \hat{e}_i]. \quad (3)$$

Among them, e_i is the direct connection between vectors $\tilde{e}_i$ and $\hat{e}_i$, containing information about $\tilde{e}_i$ and $\hat{e}_i$. Directly connecting vectors can reduce overfitting and noise.

In addition, adding additional bias terms b_i and b_j can further improve the effectiveness of embedding, and the final expression is

$$x_i^j = \tilde{e}_i \hat{e}_j^T + b_i + b_j. \quad (4)$$

Therefore, the embedding vector of commodity object can be learned by minimizing the mean square error between the predicted co-occurrence value and the true co-occurrence number.

$$\min_{\tilde{e}_*, \hat{e}_*, b_*} \sum_{t_i, t_j \in T, i \neq j, x_i^j > 0}^{T} (\tilde{e}_i \hat{e}_j^T + b_i + b_j - x_i^j)^2. \quad (5)$$

Among them T is the collection of commodities, and the main process of embedded vector learning is shown in Fig. 2.

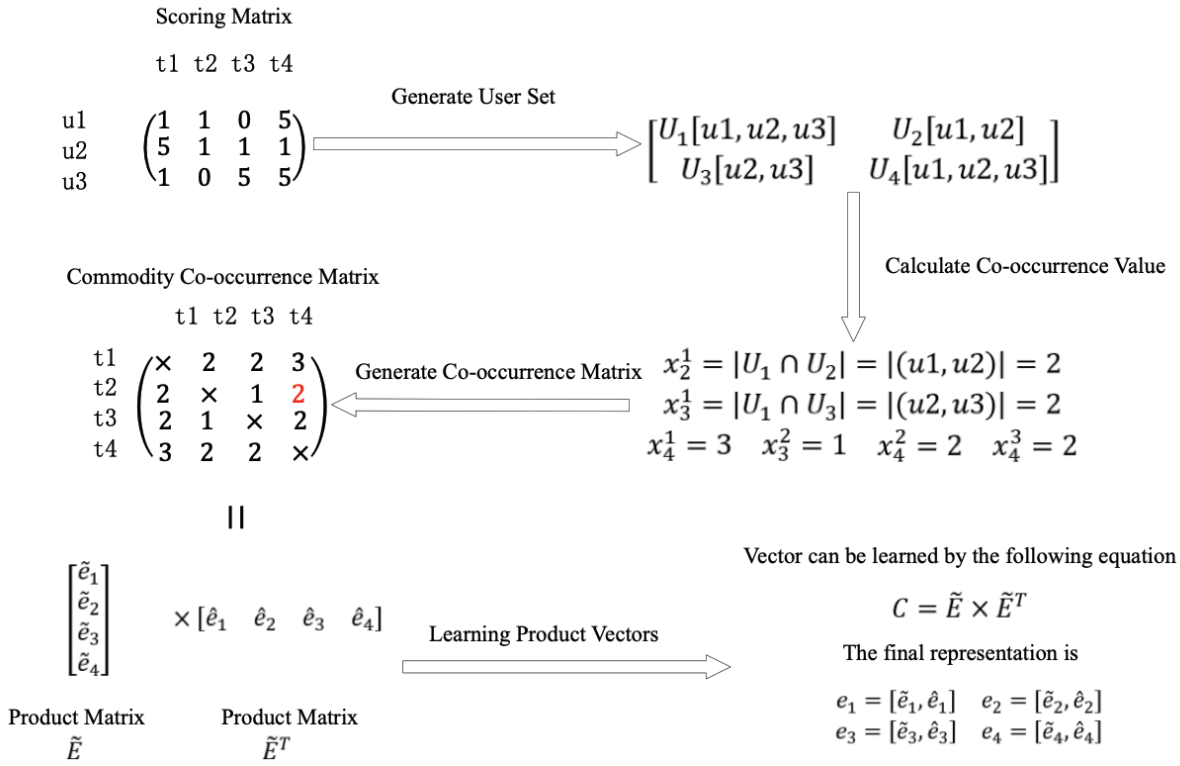

Fig. 2. Generation process of co-occurrence embedding vector

3.2 Neural NetworksMatrix Factorization

In this paper, the potential vectors of user u and commodity I are represented as p_u and q_i respectively, and the interaction y_{ui} between user and commodity is estimated by matrix decomposition as the inner product of p_u and q_i. Using matrix decomposition, the two (the underlying eigenvectors of the user and the product) are combined.

$$\hat{y}_{ui} = f(u,i|p_u,q_i) = p_u^T q_i = \sum_{k=1}^{K} p_{uk} q_{ik}. \tag{6}$$

Where the dimension of potential space is represented as K. It can be seen that matrix factorization models the potential factors of users and items, and each dimension in the potential space is independent. It can be seen from the formula that it is linearly combined and uses the same weight. Matrix factorization can be regarded as a linear model of potential factors.

It can be seen that matrix factorization is actually a simple linear product, which makes it possible to have limitations. Complex relations cannot be expressed by simple linear products, which will reduce the actual performance when the estimation result is, as shown in Fig. 3.

	i_1	i_2	i_3	i_4	i_5
u_1	1	1	1	0	1
u_2	0	1	1	0	0
u_3	0	1	1	1	0
u_4	1	1	1	1	1

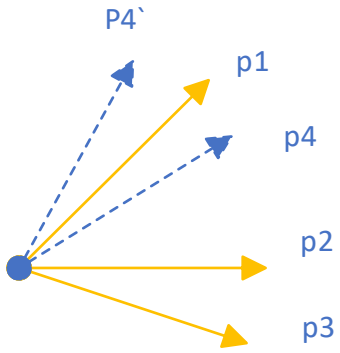

Fig. 3. Problems with matrix factorization

The cosine angle between vectors can be used to express the similarity between two vectors.

Here, Jekkard coefficient is used to simply calculate the similarity between two users [19]. First, calculate the similarity of the first three users in Fig. 3, which can be obtained $s_{23}(0.66) > s_{12}(0.5) > s_{13}(0.4)$, so the geometric relationship of vectors p_1, p_2, p_3 in potential space is obtained, as shown in the right of Fig. 3, and then add a new user u_4, which can be obtained by calculating the similarity by Jekkard coefficient $s_{41}(0.8) > s_{43}(0.6) > s_{42}(0.4)$, indicating that its similarity is $u_1 > u_3 > u_2$. However, placing p_4 near p_1, no matter how it is placed, will cause the new user's vector to be closer to the second rather than the same as the third, resulting in a large ranking loss.

1) E-learning

In order to learn model parameters, the existing point-by-point method adopts square loss regression:

$$L = \sum_{(u,i)\in Y\cup Y^-} w_{ui}(y_{ui} - \hat{y}_{ui})^2. \tag{7}$$

Y represents the set of observed interactions between users and items. Y^- represents the set of negative instances, which can be all unobserved interactions or a sample from unobserved interactions. w_{ui} is a hyperparameter that represents the weight assigned to training instances (u, i). However, the squared loss is not suitable for implicit data because the target variable y_{ui} in implicit data is a binary variable (0 or 1), indicating whether the user u interacts with item i. Therefore, this paper adopts the probability-based approach used in MLP, which pays particular attention to the binary nature of implicit data. The value y_{ui} can be seen as an attribute representing the relevance of user u to item i, where 1 indicates a relevant interaction and 0 indicates an irrelevant interaction. The predicted value $\hat{y}_{ui}$ represents the likelihood of user u being relevant to item i, and its value is constrained to the range [0, 1]. This likelihood (i.e. Probability function) is used as the activation function of the output layer, defining the likelihood function as follows:

$$p(Y, Y^- \mid P, Q, \Theta_f) = \prod_{(u,i)\in Y} \hat{y}_{ui} \prod_{(u,j)\in Y^-} (1 - y_{ui}). \tag{8}$$

Take negative logarithm to minimize the objective function:

$$L = -\sum_{(u,i)\in Y} \log y_{ui} - \sum_{(u,j)\in Y^-} \log(1 - y_{uj}) = -\sum_{(u,i)\in Y\cup Y^-} y_{ui} \log \hat{y}_{ui} + (1 + y_{ui}) \log(1 - y_{uj}). \tag{9}$$

Formula 9 are objective functions to be minimized, and adaptive moment estimation (Adam) is used to optimize them in experiments.

In this paper, the implicit feedback problem is solved as a binary classification problem. For negative instances Y^-, samples are taken in interactions that were never observed in each iteration, and the sampling ratio is controlled to control the number of samples taken.

3.3 Combination of Co-occurrence Vector and Multilayer Perceptron

At present, this paper presents a training model of commodity co-occurrence vectors, which obtains the co-occurrence embedding vectors between commodities by a similar matrix decomposition method, and adopts a neural network of multilayer perceptron, so the problem arises. How to combine the trained co-occurrence embedding vectors with the prediction results of multilayer perceptron network under the whole framework in this paper.

In this paper, besides the interaction between users and products, the co-occurrence relationship between products can also be used as a factor to reflect the recommendation results. Inspired by SVD + +, a more detailed model is proposed, as shown in Fig. 1.

$$\begin{aligned} \phi^{MLP} &= a_L\left(W_L^T\left(a_{L-1}\left(...a_2\left(W_2^T\begin{bmatrix}p_u\\q_i\end{bmatrix}+b_2\right)...\right)\right)+b_L\right) \\ y_j &= e_i = [\tilde{e}_i, \hat{e}_i] \\ \hat{r}_{ui} &= a\phi^{MLP} + b\sum\nolimits_{j\in T_u} y_j q_i \Big/ |T_u|. \end{aligned} \tag{10}$$

y_j represents e_i in Formula 3, which denotes the multiplication and addition of the input embedding vector of the item in Multi-Layer Perceptron (MLP) and the previously trained co-occurrence embedding vector. T_u represents the set of items evaluated by user u. a and b are hyperparameters that determine the trade-off between the two models. As mentioned earlier, the ReLU function is used as the activation function for the MLP. This model combines the interaction between users and items with the co-occurrence relationships among items to make predictions. In this paper, this model is referred to as item-co-Neucf.

1) Co-occurrence Embedding Vector Training

In order to learn the embedding vector in the embedding matrix C, similar to matrix factorization, it will be expressed C as the product of two embedding matrices $C = \tilde{E}\hat{E}^T = [\tilde{e}_1, \tilde{e}_2, ..., \tilde{e}_i] * [\hat{e}_1, e_2, ..., e_i]^T$.

Where $\tilde{E}$ and $\hat{E}$ are commodity embedding matrices containing different information, and $\tilde{e}_i$ and $\hat{e}_i$ are column vectors representing commodity i, which belong to $\tilde{E}$ and $\hat{E}$, respectively.

To train and learn the co-occurrence embedding vector e_i, it is done by minimizing Formula 5, which can be simplified as $e_i = [\tilde{e}_i, \hat{e}_i]$.

The specific flow is shown in Fig. 2.

In the recommendation system R, there is a rating matrix R_{3*4}. The rating data 0 in the rating matrix R_{3*4} represents the user's non-rating, and 1-5 represents the user's rating level for goods. In this paper, 1-5 is regarded as 1, which means the interaction between users and goods. Because this paper pays attention to implicit feedback, the value of rating level is not very important in this paper, but whether users interact with goods is important.

The interaction behavior here is divided by whether the user has rated the product, generating a matrix composed of the set of users U_i who have evaluated product i. It can be seen in Fig. 2 that users who have evaluated goods 1 include users 1, 2 and 3, so U_1 contains u_1, u_2, u_3, and the rest is the same.

To calculate co-occurrence values, let's take the example of co-occurrence between items 1 and 2. The users u_1, u_2, u_3 have rated item 1, while users u_1, u_2 have rated item 2. Therefore, the co-occurrence value between item 1 and item 2 is $x_1^2 = x_2^1 = 2$. Similarly, we can calculate co-occurrence values for other items, resulting in the generation of the co-occurrence matrix for the items.

The commodity co-occurrence matrix is decomposed. The decomposition method here is matrix decomposition. See 3.2.1 for details. Each co-occurrence value is decomposed into the product of two potential space vectors, which contain different information respectively. Each vector $\tilde{e}_i$ and $\hat{e}_i$ is a column vector of product i, belonging to the product matrices $\tilde{E}$ and $\hat{E}$, respectively.

Finally, the learned vectors $\tilde{e}_i$ and $\hat{e}_i$ will be connected, and the final representation of the co-occurrence embedding vector for each product i will be $e_i = [\tilde{e}_i, \hat{e}_i]$.

2) Training Multilayer Perceptron and Obtaining Prediction Results

Because the multilayer perceptron is a nonlinear model, the gradient-based optimization method will have some problems in this model, because it can only find the local optimal solution of the model, so it is necessary to consider initialization. In the aspect of initialization, the embedded layer is initialized in normal distribution, and the linear layer is initialized in uniform distribution.

After that, the multilayer perceptron model is trained until convergence. In the output layer, the output of multilayer perceptron is adjusted, and the correction term combined with over-co-occurrence embedding vector is added to the prediction.

The adaptive moment estimation (Adam) method is used to train the model from scratch, which applies smaller updates to frequent parameters and larger updates to infrequent parameters, so that the learning rate of each parameter can be adapted. Adam method converges faster than ordinary random gradient descent (SGD) for multilayer perceptron model, and does not need to adjust the learning rate, which makes SGD relatively troublesome.

4 Experimental Results

This paper evaluated the performance of the method on real data sets. The main selection is MovieLens 1M data set published by GroupLens Lab, which is an explicit data set. Details of the data set are as follows:

The data set includes four parts: user ID, commodity ID, score and score generation time.

MovieLens is a data set that records users' ratings on movies, and it is one of the most used data sets for testing recommendation systems. Its number of users is 6,040, and its number of products is about 3,900 movies. The scoring matrix is composed of users and products, with a dimension of 6040*1682. About 1 million ratings are included, and each user also scored at least 20 movies, resulting in a matrix with a sparsity of 4.2%. In this paper, the one-leave method, which is widely used in various literatures [18, 20] and [3], is used to evaluate the recommendation performance of commodities. For each user, the test suite is their latest interaction, and the remaining interaction data is used for training. The performance of the ranking list is judged by two indicators. They are accuracy (HR, which measures whether the test item appears in the top 10 list) and normalized loss cumulative gain (NDCG, which assigns higher scores to the top test items) [21]. Show the top ten effects of these two indicators (TOP-10) in Fig. 4 to Fig.13.

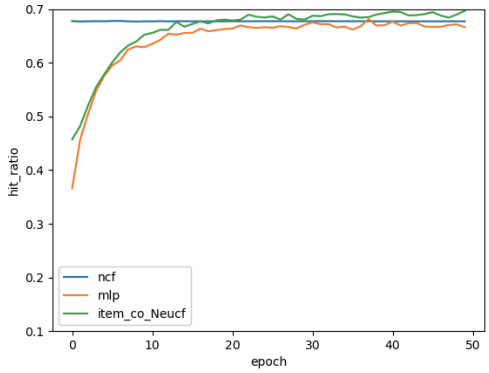

Fig. 4. Factors is 8, HR @ 10

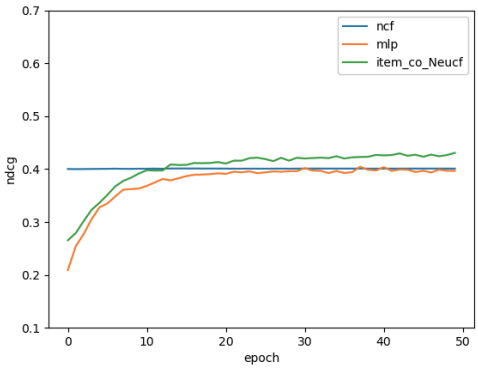

Fig. 5. Factors is 8, NDCG @ 10

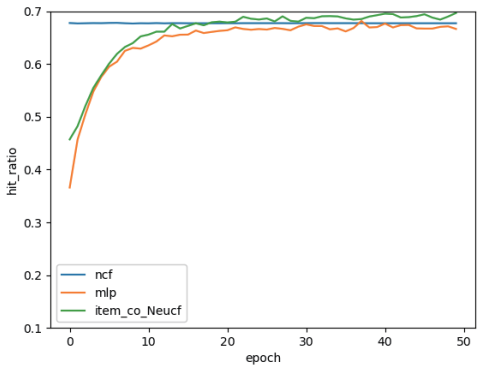

Fig. 6. Factors is 8, HR @ 10

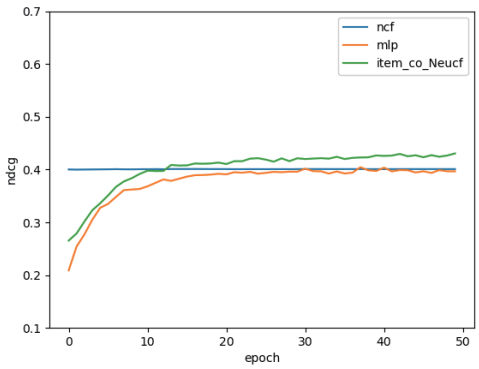

Fig. 7. Factors is 8, NDCG @ 10

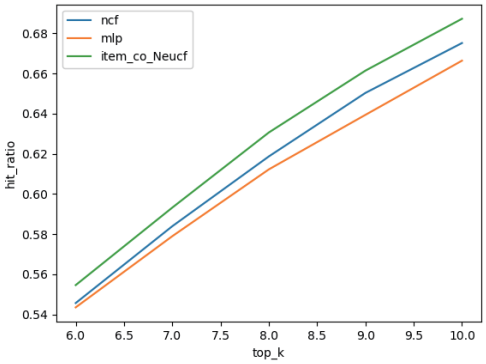

Fig. 8. Factors is 8, epoch is 20, negative sampling rate is 4

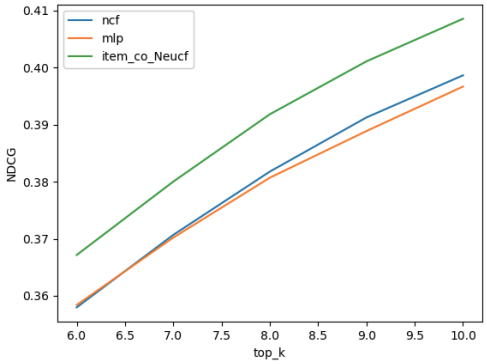

Fig. 9. Factors is 8, epoch is 20, negative sampling rate is 4

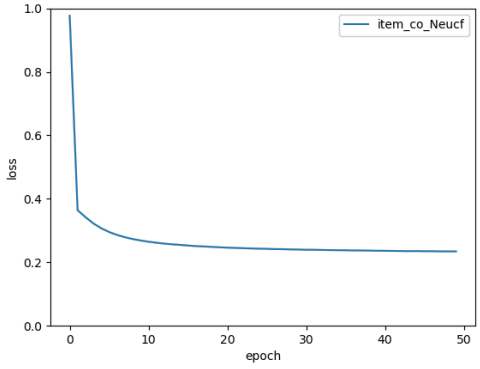

Fig. 10. The number of factors is 8, the change of Loss with the number of epoch

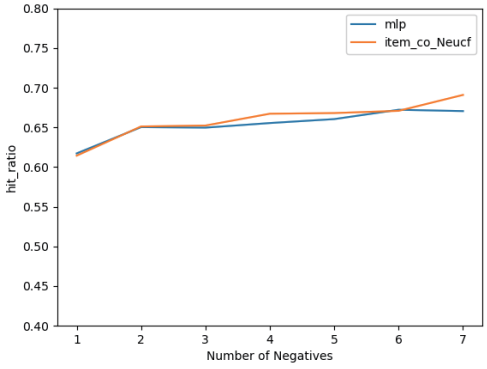

Fig. 11. Factors is 8, epoch is 20, HR @ 10

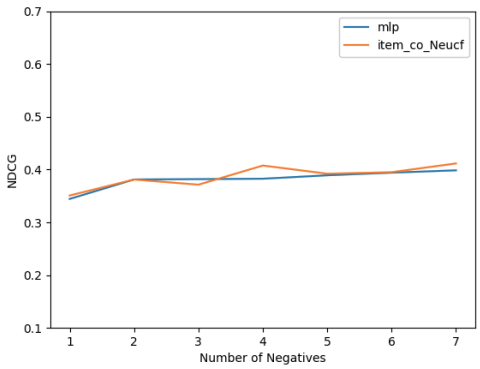

Fig. 12. Factors is 8, epoch is 20, NDCG @ 10

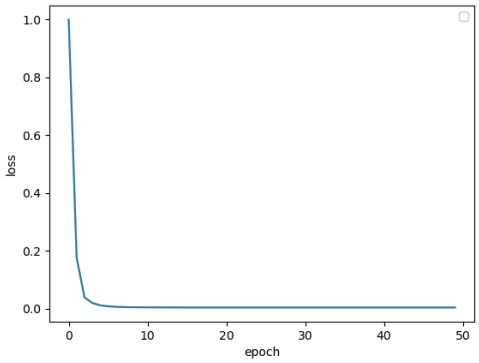

Fig. 13. The change of Loss with the number of epoch

5 Conclusion

This paper explores the neural network architecture for collaborative filtering, and proposes a recommendation system based on the relationship between deep collaborative filtering and commodity co-occurrence. Innovatively, a combination of commodity co-occurrence embedding vector and commodity embedding vector of neural network is proposed to carry out recommendation ranking based on various interactive relationships. Using matrix decomposition co-occurrence matrix to obtain co-occurrence embedding vector, the correlation between different commodities is fully considered, so that the co-occurrence features of commodities can improve the recommendation results. Multi-layer perceptron is used to better simulate the complex interaction between users and commodities, and non-linear factors in data are used to extract feature vectors of users and commodities. Finally, the multi-layer perceptron output is combined with product co-occurrence features to achieve better recommendation results.

References

[1] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Computation 18(7)(2006) 1527-1554.
[2] D. Goldberg, D. Nichols, B.M. Oki, D. Terry, Using collaborative filtering to weave all information tapestry, Communications of the ACM 35(12)(1922) 61-70.
[3] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, in: Proc. UAI'09: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, 2009.
[4] A. Van den Oord, S. Dieleman, B. Schrauwen, Deep content-based music recommendation, in: Proc. 2013 Advances in Neural Information Processing Systems 26, 2013.
[5] S. Wang, L. Hu, Y. Wang, L. Cao, Q.Z. Sheng, M. Orgun, Sequential recommender systems: challenges, progress and prospects, in: Proc. of the Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019.
[6] G. Linden, B. Smith, J. York, Amazon.com recommendations: Item-to-item collaborative filtering, IEEE Internet Computer 7(1)(2003) 76-80.
[7] R.M. Bell, Y. Koren, Improved neighbor-based collaborative filtering, in: Proc. 2007 KDD-Cup and Workshop at the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data, 2007.
[8] X. He, H. Zhang, M.-Y. Kan, T.-S. Chua, Fast matrix factorization for online recommendation with implicit feedback, in: Proc. SIGIR'16: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information, 2016.
[9] Y. Koren, Factorization Meets the Neighborhood: A multifaceted collaborative filtering model, in: Proc. 2008 the 14th ACM SIGKDD international conference on knowledge discovery and data mining, 2008.
[10] H. Wang, N. Wang, D.-Y. Yeung, Collaborative deep learning for recommender systems, in: Proc. KDD'15: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015.
[11] R.A. Salakhutdinov, A. Mnih, G. Hinton, Restricted boltzmann machines for collaborative filtering, in: ICML'07:

Proceedings of the 24th international conference on Machine learning, 2007.
[12] S. Sedhain, K.S. Menon, S. Sanner, L. Xie, Autorec: Autoencoders meet collaborative filtering, in: WWW'15 Companion: Proceedings of the 24th International Conference on World Wide Web, 2015.
[13] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, B. Kingsbury, Deep Neural Networks for acoustic modeling in speech recognition: The shared views of four research groups, IEEE Signal Processing Magazine 29(6)(2012) 82-97.
[14] G.E. Dahl, D. Yu, L. Deng, A. Acero, Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition, IEEE Transactions on Audio, Speech, and Language Processing 20(1)(2012) 30-42.
[15] P. Rodriguez, G. Cucurull, J. Gonzàlez, J.M. Gonfaus, K. Nasrollahi, T.B. Moeslund, F.X. Roca, Deep pain: Exploiting long short-term memory networks for facial expression classification, IEEE Transactions on Cybernetics 52(5)(2022) 3314-3324.
[16] A.M. Elkahky, Y. Song, X. He, A multi-view deep learning approach for cross domain user modeling in recommendation systems, in: Proc. 2015 24th International Conference on World Wide Web, 2015.
[17] G.K. Dziugaite, D.M. Roy, Neural network matrix factorization. <https://arxiv.org/abs/1511.06443>, 2015.
[18] X. He, L. Liao, H. Zhang, L. Nie, H. Xia, T.-S. Chua, Neural Collaborative Filtering, in: Proc. 2017 the 26th International Conference on World Wide Web, 2017.
[19] B. Sarwar. J. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proc. of the 10th international conference on World Wide Web, 2001.
[20] I. Bayer, X. He, B. Kanagal, S. Rendle, A generic coordinate descent framework for learning from implicit feedback, in: Proc. 2017 World Wide Web Conference, 2017.
[21] X. He, T. Chen, M.-Y. Kan, X. Chen, TriRank: Review-aware explainable recommendation by modeling aspects, in: Proc. 2015 Conference on Information and Knowledge Management, 2015.