Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 14, no. 6, June 2018, pp. 1109-1118
DOI: 10.23940/ijpe.18.06.p2.11091118

Restricted Boltzmann Machine Collaborative Filtering Recommendation Algorithm based on Project Tag Improvement

Xiaodong Qian[a,*] and Guoliang Liu[b]

[a]*School of Economics and Business Administration, Lanzhou Jiaotong University, Lanzhou, 730070, China*
[b]*School of Traffic and Transportation Engineering, Lanzhou Jiaotong University, Lanzhou, 730070, China*

Abstract

The collaborative filtering algorithm based on Restricted Boltzmann Machine (RBM) has the problem of heavy weight in the prediction of the "popular project" and poor discrimination of the "unpopular project", which results in reduced prediction accuracy of the model algorithm. In order to improve the personalization and accuracy of the model, this article integrates project tags into the prediction process based on the RBM model and uses the project tags to describe the user's own interest preference, which strengthens the individual needs of the user: First, it uses projects that the user has already graded to calculate the user's probability of rating the objective tag; Second, it uses the probability of the scoring to predict the probability of different scoring levels of the user's unprotected items; Then, RBM model training is used to predict the probability that the user will score different grades for items that are not scored; Finally, the two scoring probabilities are weighted to the RBM model prediction process to produce prediction results. Experimental results using Movielens datasets show that the accuracy of the proposed method is improved by 1.2% compared with the original algorithm.

Keywords: restricted boltzmann machine; project tag; collaborative filtering

(Submitted on February 16, 2018; Revised on March 28, 2018; Accepted on April 30, 2018)

1. Introduction

The ever-changing information technology is widely applied to various fields such as society, economy, life and so on. Information about related activities in various fields is recorded as data and saved. The rapid development of information technology to promote the cost of data preservation has been significantly reduced. Massive data is stored in the database or cloud. According to the IDC published research report [3]: The total amount of network data will grow from 1.8ZB in 2011 to 35ZB in 2020. The exponential growth of the volume of data has led us to gradually enter the Big Data Era, and in the Big Data Era, this huge data hides amazing value. The data in the Big Data environment shows the scale, diversity, real-time, low-value density characteristics, and has greatly increased the difficulty of data mining. In the Big Data environment, the data utilization rate is much smaller than its growth rate, while the difficulty of data mining is also increasing, which will inevitably lead to the problem of information overload getting worse.

At present, the solution to information overload problems are: classified catalogue, search engines and recommended systems [11].The classified directory can only cover a small amount of popular information, as it is difficult to meet user needs. Search engines can only meet the needs of users with a clear goal of active search. Recommender systems, however, can help users find information that users are interested in without explicit goals. The recommendation system can be divided into: user modelling module, recommended object modelling module and recommended algorithm module [10]. The recommended algorithm module is the core of the recommended system, so the research on the recommended algorithm is endless. The recommended algorithm can be divided according to the different strategies adopted: content-based recommendation algorithm, collaborative filtering recommendation algorithm and hybrid recommendation algorithm. Among them, the collaborative filtering recommendation algorithm is the most widely used and successful (such as Amazon

* Corresponding author.
E-mail address: qianxd@mail.lzjtu.cn

recommendation system, Netflix recommendation system, etc.). Breed [1] divides the collaborative filtering algorithm into memory-based collaborative filtering and model-based collaborative filtering. The memory-based collaborative filtering recommendation algorithm is performed by analyzing the user-item scoring matrix stored in memory, and the recommendation result is calculated by the user-item scoring matrix. The model-based collaborative filtering recommendation algorithm generally generates the implicit feature by learning the user-item scoring matrix, and it uses the implicit feature to calculate the recommended result. Model-based collaborative filtering has become a popular research direction in recent years because of its good theoretical basis and high accuracy. Its commonly used models include [12]: Bayesian Belief Network model, Clustering model, Regression model, Latent Semantic model, Low Rank Matrix Factorization model, and the Restricted Boltzmann Machine model. In recent years, Restricted Boltzmann Machine (RBM) can be used as the bottom of Deep-learning because of its high accuracy.

Salakhutdinov et al. [9] successfully applied the RBM model to the collaborative filtering recommendation algorithm for the first time, while also giving the specific implementation process of the algorithm and comparing the influence of different visible layer unit functions on the model. At the same time, the Conditional Restricted Boltzmann Machine (CRBM) was proposed for highlighting the importance of rated data; however, the model proposed by Salakhutdinov needs to convert the real value into K-dimensional binary value, which increases the training parameters in the model. In response to this question, Georgiev et al. [4] proposed a real-valued Boltzmann-based machine and improved the training of the model so that the model can directly process real-valued score data. Luo Heng [8] analyzed the Restricted Boltzmann Machine model from the point of view of collaborative filtering, and expounded the internal relationship between the Restricted Boltzmann Machine and collaborative filtering; He Jieyue et al [5] used the real-valued CRBM to train the predictive score and used the user trust network to calculate the user's trusted friend; Chen Da et al. [2] used the multi-layer RBM to construct the depth structure model and used this model to extract the abstract feature reduction dimension. In this abstract feature, the nearest neighbor recommendation method is used to form a recommendation algorithm that can quickly converge and recommend high accuracy.

The above scholars have contributed to the application of the RBM model to the collaborative filtering recommendation algorithm, and the improved algorithm performance has been greatly improved but there are still some problems. In this paper, we track each iteration of the original RBM collaborative filtering recommendation algorithm and calculate the user's prediction score for all projects. As the number of iterations increases, the accuracy of the algorithm gets higher and higher, and the variance of the predictions of all the items also increases. The variance of prediction score is larger, and the prediction score is more and more consistent with the user's personalized preference. The accuracy of the algorithm is higher and higher. But, by tracking calculation, we find that the variance of the "unpopular project" with a lower score tends to go to 0, and the variance value is basically unchanged with an increase of number of iterations. This shows that the algorithm can't accurately predict the "unpopular project"; at the same time, this paper also believes that when the algorithm predicts the "popular project", there is a partial score that corresponds to the excessive updating of the weights, resulting in the ability to predict the "popular project".

In view of the above problems, this paper proposes an improved RBM collaborative filtering recommendation algorithm based on objective tag: we use the objective tag of the user's rated item to predict the user's probability of scoring the unrated item and incorporate the probability into the prediction process of the RBM collaborative filtering recommendation algorithm. This paper analyses the feasibility of the objective tag of the project to improve the accuracy of the algorithm and improves the concrete implementation method of the algorithm. The experimental results on the MovieLens dataset show that it cannot only improve the prediction accuracy of the algorithm by 1.2%, but also that its anti-over-fitting ability is greatly improved.

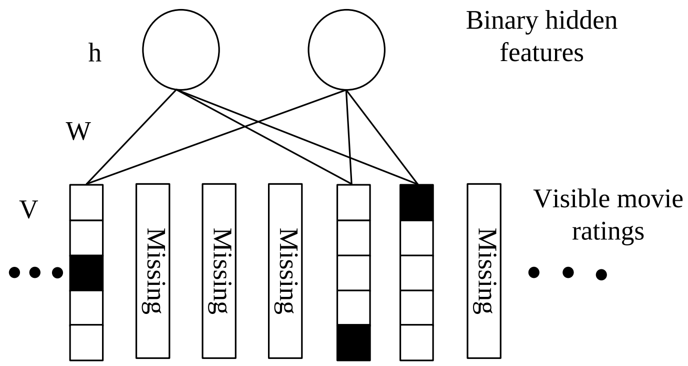

Figure 1. RBM model applied to collaborative filtering

2. Collaborative Filtering Framework Based on RBM Model

The main problems faced by applying the RBM model to the collaborative filtering algorithm are how to represent the user's unrated items and how to use the binary variable to represent the integer score. [9] is the first successful application of the RBM model to the collaborative filtering algorithm, and the specific process for the above two questions is as follows: first, the soft-max unit is used as the visible layer unit to convert the integer score value into a binary representation of the K dimension; then, the special visual unit "Missing" is introduced to represent the user's non-scoring items, and such a visible unit is not connected to any hidden units. The model is shown in Figure 1:

The RBM model is an energy-based model that defines its energy function as:

$$E(V,h) = -\sum_{i=1}^{M}\sum_{j=1}^{F}\sum_{k=1}^{K} W_{ij}^{k} h_j v_i^k + \sum_{i=1}^{M} \log Z_i - \sum_{i=1}^{M}\sum_{k=1}^{K} v_i^k a_i^k - \sum_{j=1}^{F} h_j b_j \tag{1}$$

$$Z_i = \sum_{k=1}^{K} \exp(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k) \tag{2}$$

Where W_{ij}^k represents the connection weight of the j hidden unit and the i item with the score k; h_j represents the j hidden unit; v_i^k represents the user's score for the i item k; a_i^k represents the bias of the visible unit i; b_j represents the bias of the hidden unit j. According to Equations (1) and (2), given the visible layer, the conditional probability (activation probability) of the hidden layer is:

$$p(h_j = 1|V) = \sigma(b_j + \sum_{i=1}^{M}\sum_{k=1}^{K} v_i^k W_{ij}^k) \tag{3}$$

Where $\sigma(x)$ is the sigmoid function, $\sigma(x) = 1/(1+e^{-x})$. Given the hidden layer, the conditional probability (activation probability) of the visible layer is:

$$p(v_i^k = 1|h) = \frac{\exp(a_i^k + \sum_{j=1}^{F} h_j W_{ij}^k)}{\sum_{l=1}^{K} \exp(a_i^l + \sum_{j=1}^{F} h_j W_{ij}^l)} \tag{4}$$

RBM model training maximizes its generation probability, according to the above probability formula, using Hinton [6] in the 2002 proposed Contrast Divergence (CD) algorithm, the parameters of the update:

$$\Delta W_{ij}^k = \frac{\partial \log p(V)}{\partial W_{ij}^k} = (< v_i^k h_j >_{data} - < v_i^k h_j >_{cd-model}) \tag{5}$$

$$\Delta a_i^k = \frac{\partial \log p(V)}{\partial a_i^k} = (< v_i^k >_{data} - < v_i^k >_{cd-model}) \tag{6}$$

$$\Delta b_j = \frac{\partial \log p(V)}{\partial b_j} = (< h_j >_{data} - < h_j >_{cd-model}) \tag{7}$$

$<\bullet>_{data}$ indicates the connection frequency of the visible layer soft max unit and the hidden layer unit obtained by Equation (4); $<\bullet>_{cd-model}$ represents the distribution of the reconstructed model after Gibbs sampling in the CD algorithm. After the training, the Mean Field Method is used to approximate the evaluation of a user's score for unrated movies. The formula is:

$$\hat{p}_j = p(h_j = 1|V) = \sigma(b_j + \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k W_{ij}^k) \tag{8}$$

$$p(v_i^k = 1 | \hat{p}) = \frac{\exp(a_i^k + \sum_{j=1}^{F} p_j \hat{W}_{ij}^k)}{\sum_{l=1}^{K} \exp(a_i^l + \sum_{j=1}^{F} p_j \hat{W}_{ij}^l)} \tag{9}$$

In this model, each user corresponds to a single RBM model, but all RBM model uses a common hidden layer, and the weight of the visible layer to the hidden layer in all RBM models and the respective biased terms of the two layers are shared (For example, users U_1 and U_2 both rated the project M_1 and have the same score. Then, the two users can see the connection between the layer and the hidden layer on the project M_1 using the same weight). The number of items rated by each user is much smaller than the total number of items in the model. The weight of the user in the model will have very few parts of overlap, so the corresponding RBM model can share the weight. The RBM model can be viewed as a global perspective, and the global information is used to predict the items that the users did not score.

Through the analysis of the experimental results, it is found that the final prediction score of the model is almost zero for the "unpopular project". This paper argues that the reason for this problem is that the model predicts that the " unpopular project " (which is only a small number of users who like or score) can only use a small number of users to score a project from a global perspective to predict the majority of users. So, the model can't extract effective features from less scoring times, and the model has poor ability to identify and predict such projects.

At the same time, the model part of the weight is too large, causing the model to reduce in accuracy. In the process of scoring the "popular project" (such projects are often the popular type, and its score is also more concentrated), every time the user who reviews the "popular project (i)" goes into the model training, the corresponding weight $w_{i.}$ will be updated. Especially when most users tend to score r on this project, the weight $w_{i.}^r$ of the project's rating score r will be updated more times to minimize the overall reconstruction error of the model. When $w_{i.}^r$ is updated to be large enough, it will affect the model's prediction of the rating score for the project.

Taking the film scoring system as an example, the training and forecasting process of the RBM model are analyzed: In the RBM model training stage, the parameter is updated after Gibbs sampling of the CD algorithm, while the CD algorithm is biased towards learning the characteristics that reduce the errors of reconstruction. After the user who scores the film i into the model, if the refactoring score and the original data are different, the corresponding weight of the film will be updated. For "popular movies", this type of movie will be updated several times, but when most users focus on the film score r , the corresponding weight $w_{i.}^r$ will be greatly updated, and the film's other rating weight will be far less than $w_{i.}^r$; For "unpopular movies", such films will only receive very little updates, and the weights will not change significantly. In the prediction phase of the RBM model, the Mean Field Method is used in the prediction of the score, and according to Equations (8) and (9) of the Mean Field Method, it can be seen that the weight w plays a great role in the final prediction result. Therefore, for the "popular movie (i)", a certain weight is too large, the other weight is too small, the final prediction results will be biased towards the weight of the relatively large score, resulting in a predictive score variance that is too small. For the "unpopular movie (j)", all the weight leads to a small update, and the final forecast score variance will tend to be zero.

3. Algorithm Description of Improved RBM Model Based on Project Tag

In view of the above problems, this paper proposes to improve the RBM model to solve the above problems by adopting the project objective tags (such as the movie theme in the movie scoring system, commodity classification in the e-commerce system, etc.). When the objective tag and the user's rating item are used to predict the score of the unqualified item, the score fully reflects the individual preference of the user. The objective item label will not change with the user, and its stability and objective authenticity are better. The method does not need to consider whether other users are scoring or not for the predicted project, so the method can easily give a predictor score based on the user's preference even when the "unpopular" is predicted.

Taking the film scoring system as an example, the score of the movie is analyzed, and the score is mapped to the objective film theme. The change of the subject score of the film represents the user's preference for such film. In the RBM model, for the "popular movie" ("Iron Man" series as an example), most of the people who read the film rated the film higher, and most of the users who like this movie will also like other series of the film. The model can make an accurate predictive judgment of the users who like the Marie series of films. However, when a user who does not like this type of film is in the model, the difference in the score will be modelled for less reconstruction errors and weaken, so the model can't make accurate predictions of such users. When the objective tag and the user's rating item are used to predict the score of the unqualified item, the score fully reflects the individual preference of the user, and the objective item label will not change with the user. Its stability and objective authenticity are better.

Taking advantage of objective film labels to predict can play a better effect, assuming that users do not like the "Iron Man" series and the film score is very low, indicating that users do not like the theme of the film and the same type of Marie series of movies are also likely to get a lower score. The film's score will not be affected by other users, so taking advantage of film labels can correctly predict such users on the type of film rating. For the "unpopular movie" (the war theme of the "Bode Street Island" as an example), assume that there were few people who saw the movie (assume that an extreme value of only one user read and scored 5 points), then the RBM model of all users of the film's score will be predicted 5 points, which is obviously not correct. It is more real to score using the label of the movie that the user has watched. If a user likes a war type movie, the user may have a higher rating, while another user who does not like war movies may have a lower score. Thus, the use of user project tag information can play a very good supporting role for predicting the RBM model.

In addition, the objective tags of a project in a system are repeated, even if a project can correspond to multiple objective tags. The number of objective tags for all items will still be much smaller than the number of the same items. For example, in the movie system, the movie theme is a kind of objectively existing label. A movie may correspond to two to three themes, but the movie theme only includes a very small number of themes (Movielens100k data set, 1682 movies only 19 Movie theme).

3.1. Improve the Specific Process

- Step 1. The score of the item rated by the target user is mapped to the project objective tag;
- Step 2. Statistics of the number of different rating of the target user for all objective tags;
- Step 3. Calculate the probability that the target user has a different rating level for objective tags for all items. The formula is as follows:

$$T_{ut}^{k} = \frac{num_{ut}{}^{k}}{sum_{t}} \tag{10}$$

Where T_{ut}^k represents the probability that the target user u gives the item objective tag t a score of k; $num_{ut}{}^k$ represents the number of items with the tag t and the item score k in the item that the user u has scored; sum_t indicates the sum of the number of items with tag t in the item that the user has rated.

- Step 4. Calculate the probability of the target user's rating level for the unrated item. The formula is as follows：

$$p_i^k = \frac{\sum_{t=1}^{n} a_{it} T_{ut}^k}{\sum_{t=1}^{n} a_{it}} \tag{11}$$

Where p_i^k is the probability that the target user is rated k for item i; n represents the number of objective tags for all items; $a_{it} = \begin{cases} 1 & t \in i \\ 0 & t \notin i \end{cases}$, if the tag t is the tag of item i, then $a_{it} = 1$, otherwise $a_{it} = 0$.

- Step 5. The probability of the above steps is added to the RBM model prediction process in the form of mixed weighting. The formula is as follows:

$$Q_i^k = \lambda * p(v_i^k = 1 | \hat{p}) + (1-\lambda) * p_i^k \tag{12}$$

Where $p(v_i^k = 1 | \hat{p})$ and p_i^k are calculated from Equations (9) and (11) respectively; λ represents the weight of the scoring probability of the two computational methods in the final result.

- Step 6. According to the Q_i^k, calculate the estimated of the target user's rating of the remaining movies.

$$R(u,i) = \sum_{k=1}^{K} Q_i^k * k \tag{13}$$

3.2. Algorithm Pseudo-Code Description

Step 1 Calculate the probability of a score for an unrated item based on an objective item tag

Algorithm 3.1 Calculates the probability of a user's rating score for an unrated item

for u =1:number_user **do**
Use the Equation (10) to calculate the probability that the user u has different ratings for all
item objective tags;
Use the formula (11) to calculate the probability that the user u has a rating level for the
unrated item;
end for

Step 2 Training RBM

Algorithm 3.2 RBM-Training algorithm

repeat
epoch = 1:max_epoch;
for all mini-batch of users in S_{batch} and $S_{batch} \in S$ **do**
for all $user \in S_{batch}$ **do**
Translate the ratings of user to Soft-max as visible units v_i^k;
Use Equation (3) compute all the hidden units h_j;
Record samples $v_i^k h_j$ 、 v_i^k 、 h_j;
Run CD algorithm to the Gibbs sampler;
for step=1:CD_step **do**
Gibbs sampler all the hidden units $< h_j >^{step}$;
Use Equation (4) compute all the visible units $P(v_i^k = 1 | h)$;
Gibbs sampler all the visible units $< v_i^k >^{step}$;
Use Equation (3) compute all the hidden units h_j;
end for
Record samples $< v_i^k h_j >^{step}$ 、 $< v_i^k >^{step}$ 、 $< h_j >^{step}$;
end for
Average the first samples to get $< v_i^k \cdot h_j >_{data}$ 、 $< v_i^k >_{data}$ 、 $< h_j >_{data}$;
Average the second samples to get $< v_i^k \cdot h_j >_{cd-model}$ 、 $< v_i^k >_{cd-model}$ 、 $< h_j >_{cd-model}$;
Use Equation (5)、(6)、(7)compute ΔW_{ij}^k 、 Δa_i^k 、 Δb_j;

20. Update $W_{ij}^{k} = \rho * W_{ij}^{k} + \theta * \Delta W_{ij}^{k}$;
21. Update $a_{i}^{k} = \rho * a_{i}^{k} + \theta * \Delta a_{i}^{k}$;
22. Update $b_{j} = \rho * b_{j} + \theta * \Delta b_{j}$;
23. **end for**
24. epoch = epoch + 1;
25. Compute the error Err_{epoch};
26. **until** $Err_{epoch-1} - Err_{epoch} > \varepsilon$ **or** epoch = max epoch.

Step 3 Prediction

Algorithm 3.3 Recommendations algorithm

1. Translate the ratings of user u to Soft-max units;
2. Use Equation (8) compute $\hat{p}_{j}$ for all hidden units j;
3. Use Equation (9) compute $p(v_{q}^{k} = 1 \mid \hat{p})$ for all $k = 1,2...,K$;
4. Use Equation (11) compute p_{i}^{k};
5. Use Equation (12) compute Q_{i}^{k};
6. Use the Equation (13) to calculate the forecast score $R(u,i)$.

4. Experimental Analysis

4.1. Data Sources and Experimental Settings

The experimental data were based on the MovieLens100K data set (http://www.grouplens.org) published by the GroupLens research team. The data set is a movie scoring system, including 943 users of 1682 movies 100000 score information, a rating level of 1 to 5 points. The release of the data set also includes the movie theme information, movie IMDB link and user information and other related data.

The experiment uses Matlab2015b as the data processing platform, extracts 80% of the data set as the experimental training set, and the remaining 20% as the experimental test set. In the experiment, the standard RBM model of [9] was used as the contrast test, and the average of 10 experiments was taken as the final result of the experiment.

4.2. Evaluation Index

At present, the accuracy of recommender systems is usually measured by Root Mean Square Error (RMSE), and the formula is:

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in R_{test}} (R_{u,i} - \hat{R}_{u,i})^2}{\left|N_{R_{test}}\right|}} \tag{14}$$

Where R_{test} denotes the test data set; $R_{u,i}$ denotes the actual score of the user u for the movie i; $\hat{R}_{u,i}$ denotes the predicted score of the user u for the movie i; $N_{R_{test}}$ denotes the number of data in the test data set; The smaller the results of the two evaluation indicators, the higher the recommended accuracy.

4.3. Experimental Results and Analysis

In this paper, the standard RBM model algorithm in [9] and the tag-based prediction algorithm are used as the contrast experiments of the algorithm. The tag-based prediction algorithm is a predictor score based on the expected value of the tag score. In this paper, the algorithm and the standard RBM model algorithm have many parameters to be set. To ensure the comparability of the experimental results, the two algorithms use the same experimental parameters. The specific parameters are shown in Table 1:

Table 1. The main parameters of the model settings

Parameter	Parameter Value
Number of hidden units	60
Weight attenuation coefficient	0.0005
Weight learning efficiency	0.001
Visible units bias learning efficiency	0.001
Hidden units bias learning efficiency	0.01
Number of iterations	100
Number of CD algorithm iterations	3

The above parameters have a great influence on the prediction results of the two algorithms, so the setting of the parameters also occupies an important position in the algorithm. However, the main purpose of this paper is to compare the algorithm with the standard RBM model algorithm, so the parameter setting part does not give detailed experimental results, and a detailed discussion of the effects of RBM parameter settings on the model can be found in references [7,13]. In this paper, we only need to use the same parameters to ensure that the two algorithms can be comparable.

Calculate the effect of the RBM algorithm and the improved algorithm on the variance of the "unpopular project": This paper will be judge "unpopular movies" and calculate the variance of these "unpopular movies" in two different algorithms. The specific experimental results are shown in Figure 2:

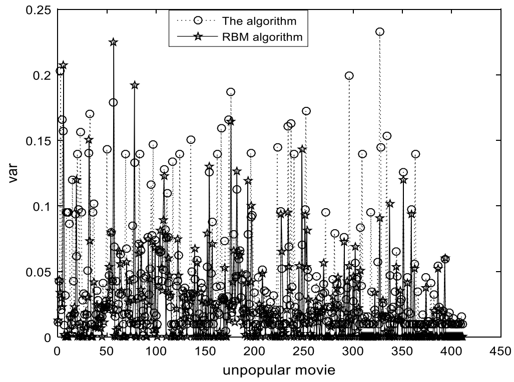

Figure 2. “Unpopular movies” score prediction variance contrast

It can be seen from Figure 2 that the variance of most of the "unpopular movies" calculated by this algorithm is larger than the variance calculated by the original RBM algorithm. This shows that this algorithm can be based on different interests of the user preferences generated by the differential score prediction, which is more in line with the actual situation. However, it is impossible to judge whether the performance of the improved algorithm is better according to the experiment. Therefore, the accuracy index RMSE calculated from Equation (14) is used as the criterion to judge the algorithm. The calculation results are shown in Figure 3:

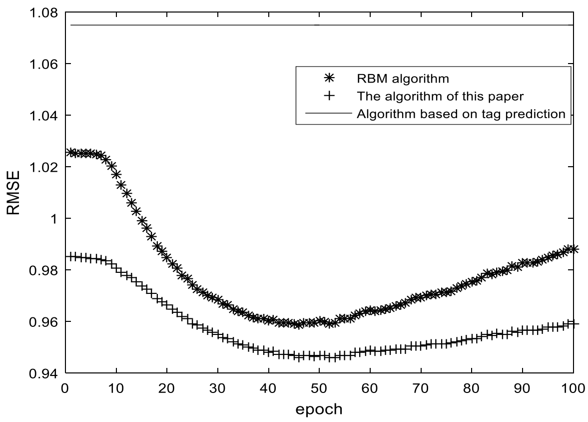

Figure 3. Comparison of the RMSE values of the algorithm, RBM algorithm and tag prediction algorithm

It can be seen from Figure 3 that the RMSE value of this algorithm is always smaller than the original RBM algorithm, and it is far less than the result of the algorithm based on the tag prediction algorithm, which indicates that the algorithm is more accurate. Secondly, when the number of iterations (epoch) is small (between 0 and 15 times), the RMSE difference between the algorithm and the original RBM algorithm is larger. Because both of the algorithms are based on the RBM model, when the number of iterations is small, the original RBM model algorithm does not learn enough features and its prediction accuracy is low. The improved algorithm, even in the lower number of iterations, gets enough user information, so the forecast is significantly better. With the number of iterations reaches the optimal value of the model (50), the original RBM model algorithm can learn enough features, and its prediction ability is stronger. But, there are still some items that the original RBM model algorithm cannot accurately judge (such as "unpopular projects", "popular projects" in the difference score). This algorithm can be combined with the project objective tag to predict this part of the project, so the accuracy of the improved algorithm is still higher than the original algorithm. At the same time, after the model reaches the optimal iteration number, the fitting problem occurs, resulting in a larger RMSE value. However, the RMSE value of this algorithm is much smaller than that of the original RBM algorithm, which indicates that the algorithm has better anti-over-fitting ability than the original RBM algorithm.

5. Conclusions

A good recommendation algorithm must first ensure the accuracy of its recommendation, so it is an important research direction to improve the recommendation accuracy rate. Highly accurate recommendation algorithms can provide customers with goods that meet their interests and increase the user's satisfaction with the recommended system to enhance user adherence to the recommender system. In this paper, when the RBM model is applied to collaborative filtering, it integrates the objective tag of the project and improves the discrimination ability of the algorithm. Especially in the prediction of excessive concentration of "popular projects" and scoring less "unpopular projects", the accuracy is higher. At the same time, the algorithm also shows better anti-over-fitting ability. Improving the algorithm into the depth of learning will be a main future direction.

Acknowledgements

This article is supported by the National Natural Science Foundation of China under grant No.71461017.

References

1. J. S. Breese, D. Heckerman, C. Kadie, “Empirical Analysis of Predictive Algorithms for Collaborative Filtering”, Fourteenth Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers, Inc, pp.43-52, 1998.
2. D. Chen, S. Gao, Z. Q. Lin, “A Survey on Recommendation System Algorithm Based on Restricted Boltzmann Machine”, Software, vol.12, pp.156-159, 2013.
3. J. Gantz, D. Reinsel, “2011 Digital Universe Study: Extracting Value from Chaos”, IDC Go-to-Market Services, 2011.
4. K. Georgiev, P. Nakov, “A non-IID Framework for Collaborative Filtering with Restricted Boltzmann Machines”, 8International Conference on Machine Learning, pp.1148-1156, 2013.
5. J. Y. He, B. Ma, “Based on Real-Valued Conditional Restricted Boltzmann Machine and Social Network for Collaborative Filtering”, Chinese Journal of Computers, vol.1, pp.183-195, 2016.
6. G. Hinton, “Training Products of Experts by Minimizing Contrastive Divergence”, Neural Computation, vol.14, no.8, pp.1771-1800, 2002.
7. G. Louppe, “Collaborative Filtering: Scalable Approaches Using Restricted Boltzmann Machine”. English, 2010.
8. H. Luo, “Restricted Boltzmann Machines: A Collaborative Filtering Perspective”, Shanghai Jiao Tong University, 2011.
9. R. Salakhutdinov, A. Mnih, G. Hinton, “Restricted Boltzmann Machines for Collaborative Filtering”. Machine Learning, Proceedings of the Twenty-Fourth International Conference, pp.791-798, 2007.
10. G. X. Wang, H. P. Liu, “Survey of Personalized Recommendation System”, Computer Engineering and Applications, Vol 48, pp. 66-76, 2012.
11. L. Xiang, “Recommender Systems Practice”, The people’s Posts and Telecommunications Press, 2012.
12. H. L. Xu, X. Wu, X. D. Li, et al., “Comparison Study of Internet Recommendation System”. Journal of Software, vol.20, no.2, pp.350-362, 2009.
13. C. X. Zhang, N. N. Ji, G. W. Wang, “Restricted Boltzmann Machines. Chinese Journal of Engineering Mathematics”, no.2, pp.159-173, 2015.

Xiaodong Qian graduated with a doctorate degree from the School of Management Science and Engineering, Tianjin University. He entered the post-doctoral mobile station of the Control Science and Engineering of Tianjin University. Now, he is a professor at Lanzhou Jiaotong University. He is also a member of CCF. His current research interests include data mining, data analysis, business intelligence.

Guoliang Liu is a Master's student from the School of Traffic and Transportation Engineering, Lanzhou Jiaotong University. His research interests include complex network and intelligent algorithms.