Collaborative Filtering Fusing Label Features Based on SDAE

Huan Huo[1], Xiufeng Liu[2], Deyuan Zheng[1], Zonghan Wu[1], Shengwei Yu[1], and Liang Liu[1]

[1] School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai, China
[2] Technical University of Denmark, 2800 Kgs. Lyngby, Denmark

Abstract. Collaborative filtering (CF) is successfully applied to recommendation system by digging the latent features of users and items. However, conventional CF-based models usually suffer from the sparsity of rating matrices which would degrade model's recommendation performance. To address this sparsity problem, auxiliary information such as labels are utilized. Another approach of recommendation system is content-based model which can't be directly integrated with CF-based model due to its inherent characteristics. Considering that deep learning algorithms are capable of extracting deep latent features, this paper applies Stack Denoising Auto Encoder (SDAE) to content-based model and proposes DLCF(Deep Learning for Collaborative Filtering) algorithm by combing CF-based model which fuses label features. Experiments on real-world data sets show that DLCF can largely overcome the sparsity problem and significantly improves the state of art approaches.

Keywords: recommendation system; collaborative filtering; deep learning; label feature

1 Introduction

Recommendation system, as a hot topic in recent years' research, helps people acquire useful information through massive overloading Internet information. In this field, collaborative filtering, due to its capability of digging the latent features of users and items by using matrix decomposition technique, is widely applied to recommendation system and has made a lot of progress. However, one natural drawback of recommendation system is that it cannot address the sparsity problem of rating matrix. In order to overcome the shortcoming of conventional collaborative filtering, adding auxiliary information into sparse matrix is an effective choice. At present, many innovative collaborative recommendation approaches have made their own efforts in this field, such as [1] considering the label features of items, [2] introducing hybrid recommendation, and so on. These approaches can alleviate the sparsity problem of rating matrix to some extent. However, label matrix can be very sparse in the most cases, so merely depending on introducing original label information is not enough to overcome the shortcoming of collaborative filtering. Another approach of recommendation system

P. Perner (Ed.): ICDM 2017, LNAI 10357, pp. 223–236, 2017.
DOI: 10.1007/978-3-319-62701-4_17

is content-based recommendation. [3] maintains a feature vector or an attribute set to establish recommendation system by depicting the portraits of users or items. The drawback of content-based recommendation is that it cannot extract deep features automatically and dip the deep latent features of the items and the potential interests of users. Combined with the industry consideration towards Internet privacy, this problem can be even worse. Therefore, content-based recommendation needs to combine with collaborative filtering to establish hybrid recommendation in order to achieve better performance of recommendation.

How to increase the information of recommended data and dig the deep latent features of content information is the key of improving the performance of recommendation algorithm. Content-based recommendation algorithm usually uses the model such as LDA(Latent Dirichlet Allocation) when extracting features. This kind of model performs well in conventional content-based recommendation. On the other hand, deep learning is capable of learning deep features automatically. In this field, CNN(Convolutional Neural Network)[5] and RNN(Recurrent Neural Networks)[6] are widely used in the field of image recognition and natural language processing, which have achieved great performance. This ability of deep learning algorithms is very suitable for combining with collaborative filtering in the application of content-based model.

This paper proposes DLCF(Deep Learning for Collaborative Filtering) algorithm. Its framework generates a new matrix of label-items by using SDAE(Stacked Denoising Autoencoders)[7] training the feature vectors of items and lables, transforming sparse label matrix which contains less amount of information into label matrix which contains information of deep latent features, increasing the original data information substantially; then conducting collaborative filtering processing combining with the original rating matrix.

The contribution of this paper mainly consists of three aspects: 1)Applying deep learning algorithm to content-based model, increasing the original data information substantially by extracting deep features; 2)Combining the feature vectors of items and labels with the original rating matrix by building auxiliary matrix, playing the role of data label to the most extent, thus integrating data label, content and rating matrix into a whole framework; 3)Conducting experiments on real-world data sets for the performance of algorithm, pointing out the effect of model parameters on the performance of algorithm.

The rest of this paper is organized as follows. Section 2 reviews previous related work. Section 3 provides the details of DLCF algorithm. Section 4 conducts experiments on real-world data sets.

2 Related Work

Collaborative filtering is the most widely-used recommendation technique at present. [8] firstly proposes LFM(Latent Factor Model) based on SVD(Singular Value Decomposition); [9] adds probability distribution into LFM and introduces PMF(Probabilistic Matrix Factorization); [10] goes further to extend PMF into Bayesian PMF, and train the data by Markov Chain Monte Carlo; [11] and [12]

try to combine collaborative filtering model with content-based model. Considering the auxiliary information, [13] proposes that labels of items can be applied to collaborative filtering; [14] partially solves the sparsity of matrix and cold start problem by the category information of items; [15] proposes social label recommendation by analyzing the relationship between the objects in the label system; [16] proposes a kind of tag-based Recommendation system method called Tag-CF. However, these approaches can only utilize the auxiliary information provided by the data sets, but cannot mine out the deep latent information.

In recent years, deep learning is applied to image recognition and natural language processing. But the attempt to apply deep learning to recommendation system has also emerged. [17] uses RBM(Restricted Boltzmann Machine) to take the place of Matrix Decomposition Model in order to implement collaborative filtering; [18] develops further based on [17] by combining the correlation between users and items; [19] uses DBN(Deep Belief Networks) to dig deep content information; [20] proposes a kind of relational SDAE model;[21] proposes a kind of content-based music recommendation system called DeepMusic. But all these algorithms haven't been applied to collaborative filtering.

The core of DLCF algorithm proposed in this paper is to apply the deep learning model to content-based recommendation model and combine with collaborative filtering algorithm. The deep learning model SDAE [22] is a model formed by stacking multiple DAEs (Denoising Auto Encoder) , which can extract the deep features of content information, and have strong interpretability and lower model complexity. SDAE can extract the new feature dimension from the original content information and train the label content matrix. It greatly increases the amount of information available in the original data. We then integrate the new content label matrix into PMF, thus perfectly combing the content-based recommendation model and collaborative filtering model. Meanwhile, this paper adopts SGD (Stochastic Gradient Descent) to train the model parameters by minimizing the loss function, which overcomes the problem of slow iteration of deep learning algorithm, and solves the problem of sparsity in the conventional collaborative filtering algorithm, as well as the problem of lack of useful information in the content-based recommendation model.

2.1 Probabilistic Matrix Factorization

PMF model introduces prior probability distribution into conventional matrix factorization model. Assuming that the conditional probability of observed rating data is:

$$p(R|U,V,\sigma_R^2) = \prod_{u}^{M}\prod_{i}^{N}(\varkappa(R_{u,i}|U_u^T V_i,\sigma_R^2))^{I_{u,i}^R} \tag{1}$$

Where $\varkappa(x|\mu,\sigma^2)$ represents the probability density function of normal distribution which has the mean of μ and the variance of σ^2. $I_{u,i}^R$ means that if the user $u's$ rating towards the item i exists, the value of I equals to 1, otherwise it

equals to 0. We assume the mean equals to 0 and the variants of u and i are σ_U^2 and σ_V^2, respectively:

$$p(U|\sigma_U^2) = \prod_u^M \varkappa(U_u|0, \sigma_U^2 I)$$
$$p(V|\sigma_V^2) = \prod_i^N \varkappa(V_i|0, \sigma_V^2 I) \quad (2)$$

The model is shown in Fig. 1.

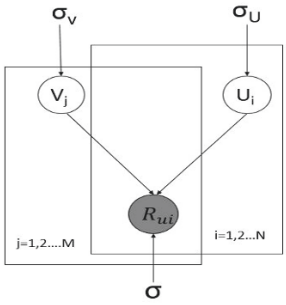

Fig. 1. PMF Schematic Diagram

2.2 Stacked Auto Encoder

DAE, namely Denoising Auto Encoder, consists of encoder and decoder. Each encoder has its corresponding decoder, and processes the noise of data through the course of encoding and decoding. Fig.2 shows that SDAE is a kind of Feed Forward Neural Network which stacks multiple DAEs similar to multilayer perceptions. Each layer's output is the input of its next layer. SDAE uses greedy layer-wise training strategy to train each layer in the network successively, and then pre-train the whole deep learning network. The idea of SDAE is to stack multiple DAEs in order to form a deep learning framework. This model trains the middle layers by the loss input and recovered output.

Formula (3) is the training model of SDAE. In this model, Z_c is the matrix consisting of several label vectors, which is the last output of SDAE. Z_0 represents the initial loss input matrix. Z_L is the middle layer of the model. The target matrix we need to achieve through the training is $Z_{L/2}$, which represents the matrix containing deep content information through the training from Z_0 to Z_C. w_l and b_l represent the weight matrix of the lth layer and the bias vector in SDAE model, respectively. λ represents regular parameter. $||\cdot||_F$ is Frobenius norm.

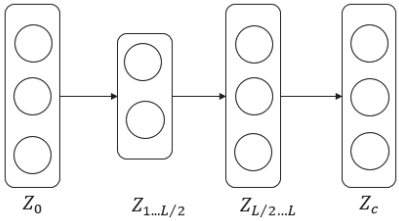

Fig. 2. SDAE Consisting of Multiple Layer DAE

$$min_{\{w_l\},\{b_l\}} = ||Z_c - Z_L||_F^2 + \lambda \sum_l (||w_l||_F^2 + ||b_l||_2^2) \tag{3}$$

3 DLCF Algorithm

3.1 Preparing Work

According to the clustering algorithm [23], we first generated the user and the item clusters, as shown in algorithm 1, where θ is the threshold that we can decide:

```
Algorithm 1: preprocessing algorithm
1. Compute rating frequency of user f_u and item f_i
2. Sort users and items based on f_u and f_i in reverse order
3. Compute similarity of users and items
4. t ← 1, U ← φ, I ← φ
5. FOR j = 1, 2, ···
6.    IF u_j ∉ U
7.       U_t ← u_j ∪ u_k|s_{u_j,u_k} > θ_j, u_k ∉ U
8.       U← U_t ∪ U, t ← t + 1
9.    ENDIF
10.ENDFOR
11.The same as cluster I
```

The clusters we have built contains (user cluster)-(item) preference information and (item cluster)-(user) preference information. By utilizing the local preferences information, we can speed up the greedy layer-wise training strategy in SDAE, and then pre-train the whole deep learning network.

3.2 Training label matrix

Firstly, based on SDAE model and its theory, output matrix Z_c and loss input matrix Z_0 are observed variables. Then Z_c is defined as follows: For each layer l

in SDAE:

$$\begin{aligned} W_{l,*n} &\sim \varkappa(0, \lambda_w^{-1} I_{K_l}) \\ b_l &\sim \varkappa(0, \lambda_w^{-1} I_{K_l}) \end{aligned} \tag{4}$$

Where $W_{l,*n}$ represents the nth column in the weight matrix of the lth layer. I_K represents the Kth diagonal value in the unitary matrix. For Z_c and Z_L:

$$\begin{aligned} Z_{l,j*} &\sim \varkappa(\sigma(Z_{l-1,j*} W_l + b_l), \lambda_s^{-1} I_{K_l}) \\ Z_{c,j*} &\sim \varkappa(Z_{L,j*}, \lambda_n^{-1} I_B) \end{aligned} \tag{5}$$

Where $\sigma(\cdot)$ represents sigmoid function. λ_w, λ_n and λ_s are all model parameters. Based on the above definitions, maximizing the maximum posterior probability of W_l,b_l,Z_l,Z_c, is the same as minimizing the joint log-likelihood function of the above variables. Thus the loss function of the model is defined as:

$$\begin{aligned} \epsilon = &- \frac{\lambda_w}{2} \sum_l (||W_l||_F^2 + ||b_l||_2^2) \\ &- \frac{\lambda_n}{2} \sum_j ||Z_{L,j*} - Z_{c,j*}||_2^2 \\ &- \frac{\lambda_s}{2} \sum_l \sum_j ||\sigma(Z_{l-1,j*} W_j + b_j) - Z_{l,j*}||_2^2 \end{aligned} \tag{6}$$

Assume $T_{i,j}$ is defined as a boolean. If item j includes label i, the value of $T_{i,j}$ equals to 1, otherwise it equals to 0. Based on $Z_{L/2,j*}$ and original label matrix $T_{i,j}$, we can find latent factor vectors of labels and items t_i and v_j:

$$\begin{aligned} \gamma = &- \frac{\lambda_t}{2} \sum_i ||t_i||_2^2 - \frac{\lambda_v}{2} \sum_j ||v_j - Z_{L/2,j*}^T||_2^2 \\ &- \sum_{i,j} \frac{c_{i,j}}{2} (T_{i,j} - t_i^T v_j)^2 \end{aligned} \tag{7}$$

Where λ_t and λ_v are regular parameters, respectively. $c_{i,j}$ is set to 1 when item j includes label i, otherwise it is set to a small value, such as 0.001 or 0.

3.3 Establishing User-label Matrix

We define matrix G as pre-processed label-item matrix, which is used to establish DLCF. We get it by feature vectors of labels and items, namely t_i and v_j. $R_{u,i}$ represents the rating of item i by user u. $G_{i,t}$ represents the grade of item i for label t, which equals to 1 if i includes t, otherwise it equals to 0. By jointing matrix R and G, we can get target matrix H:

$$H_{u,t} = \frac{1}{N} \sum_i^n R_{u,i} G_{i,t} \tag{8}$$

3.4 Establishing DLCF Model

After establishing user-label matrix H, we use rating matrix R and H to improve conventional CF model, as shown in Fig.3. U and V represent the latent feature vectors included by users and items, respectively. Q represents the relationship between labels and latent features, which bridges the information circulation of R and H:

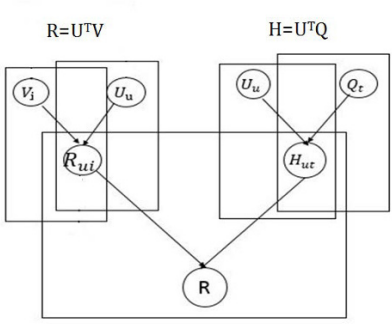

Fig. 3. Establishing Joint Matrix

After fusing H into PMF, we establish new loss function:

$$\begin{aligned} E = & \frac{1}{2}\sum_{u}\sum_{i} I_{u,i}^{R}(R_{ui} - U_u^T V_i)^2 \\ & + \frac{\omega_U}{2}\sum_{u}||U_u||_F^2 + \frac{\omega_V}{2}\sum_{i}||V_i||_F^2 \\ & + \frac{\alpha}{2}\sum_{u}\sum_{t}(H_{ut} - U_u^T Q_t)^2 \\ & + \frac{\varphi_U}{2}\sum_{u}||U_u||_F^2 + \frac{\varphi_Q}{2}\sum_{t}||Q_t||_F^2 \end{aligned} \tag{9}$$

where $\omega_U = \frac{\sigma_R}{\sigma_U}$ and $\omega_V = \frac{\sigma_R}{\sigma_V}$

$$\begin{aligned} E = & \frac{1}{2}\sum_{u}\sum_{i} I_{u,i}^{R}(R_{ui} - U_u^T V_i)^2 \\ & + \frac{\alpha}{2}\sum_{u}\sum_{t}(H_{ut} - U_u^T Q_t)^2 \\ & + \frac{\lambda_U}{2}\sum_{u}||U_u||_F^2 + \frac{\lambda_V}{2}\sum_{i}||V_i||_F^2 \\ & + \frac{\lambda_Q}{2}\sum_{t}||Q_t||_F^2 \end{aligned} \tag{10}$$

where λ_U=$\omega_U + \varphi_U$,$\lambda_V = \omega_V$,$\lambda_Q = \omega_Q$.

3.5 SGD Training Algorithm

We use SGD(Stochastic Gradient Descent) to train E.

Algorithm 2: DLCF-SGD training algorithm
Input: rating matrix R, the dimension of the feature vectors K, learning rate η, scale parameter α, regular parameters $\lambda_U, \lambda_V, \lambda_Q$
Output: U, V
1. Initialization: establish U, V and Q by using a small value stochastically.
2. While(error on validation set decrease):
3. $\nabla_U E = I(U^T V - R)V + \alpha(U^T Q - H)Q + \lambda_U U$
$\nabla_V E = [I(U^T V - R)]^T U + \lambda_V V$
$\nabla_Q E = \alpha(U^T Q - H)^T U + \lambda_Q Q$
4. Set $\eta = 0.1$
5. While $E(U - \eta\nabla_U E, V - \eta\nabla_V E, Q - \eta\nabla_Q E) > E(U,\ V,\ Q)$
6. Set $\eta = \eta/2$
7. End while
8. $U = U - \eta\nabla_U E$
$V = V - \eta\nabla_V E$
$Q = Q - \eta\nabla_Q E$
9. End while
10. Return U,V

4 Experiment Results and Analysis

4.1 Data sets

We use the data from Douban Reading (https://book.douban.com/), a social network well-known for users to rate different published books in China. Each book in this website has been rated from 1 to 5 by users. Moreover, each book has the features labeled by users, which can be applied to the feature vectors of content-based model. The data from this website is perfect for the application of DLCF algorithm. The data set includes 384341 users, 89798 books, and 13456139 rating data. The data record is formatted as (UserID, BookID, Rating, Labels). In this experiment, we choose the data sets with different sparsity. For these data sets, we choose 80% as training data and 20% as testing data. During the experiment, we divide the data into 5 parts stochastically for cross-check.

4.2 Algorithm Evaluation Criteria

In general, there are two ways to evaluate the recommendation system. One is evaluating the distance between predicted rating and users' actual rating, which is very common. The other is to evaluate the accuracy of the prediction.

In this paper, we use RMSE (Root Mean Square Error) as the criteria measuring the distance between the predicted rating and the users' rating, as shown in equation 11.

$$RMSE = \sqrt{\frac{1}{||\tau||} \sum_{(u,i)\in\tau} (r_{u,i} - \hat{r})^2} \tag{11}$$

Where τ represents the set including the existing rating of item i by user u.

To evaluate the accuracy of the algorithm prediction, we use recall@R as the metric. By choosing the testing users, recall@R sorts the recommended results and chooses the top R most favored by the users. Then we calculate the ratio of the number of the items in top R results to total number of the items the users like. The greater the ratio is, the more effective the performance of the algorithm is.

$$recall@R = \frac{number\_of\_items\_that\_users\_like\_among\_top\_R}{total\_number\_of\_items\_that\_users\_like} \tag{12}$$

4.3 Comparison Model and Experimental Settings

To evaluate the performance of DLCF algorithm, we choose three algorithms to compare. These algorithms are PMF [9] (collaborative filtering with matrix decomposition), Tag-CF [16](algorithm combining label information), and DBN [19](deep learning algorithm without tag). Firstly, we compare the above three algorithms with DLCF horizontally. Then we observe vertical comparison performance of DLCF under different experimental settings. In such case, we conduct comprehensive evaluation of the performance of each algorithm.

Before making further experimental comparison, we study the parameter α in (10), which represents the effect factor of user-label matrix in the whole model. If we set α to 0, it represents that user-label matrix is not put into consideration. In this case, algorithm is degraded to conventional collaborative filtering without considering label. In the experiment depicted in this paper, when the other parameters are fixed, RMSE can be minimized on condition that α equals to 0.9.

Table 1: The Effect of Parameter α on RMSE

α	RMSE
0	0.98
0.1	0.93
0.5	0.87
0.9	0.82
1.2	0.85
2	0.99
10	1.33

Other parameters $\lambda_U = \omega_U + \varphi_U, \lambda_V = \omega_V, \lambda_Q = \omega_Q, \omega_U = \sigma_R/\sigma_U, \omega_V = \sigma_R/\sigma_V$ which represent the regular parameters in the model, are compounded by other latent parameters. Theoretically, we need to take the value of each latent parameter, and then calculate the value of λ_U,λ_V,λ_Q. However, in practice, we can set λ_U, λ_V, λ_Q to small values respectively. For example, $\lambda_U = \lambda_V = \lambda_Q = 0.001$. Then we adjust them by cross validation in the experiment. The result proved that such an approach does not produce a significant impact on the performance of the algorithm.

In the next comparison experiment, we divide experimental data into two parts according to their different sparsity. Then we compare the performance of each algorithm based on these two kinds of evaluation methods.

Table 2: Sparsity of Dataset-a and Dataset-b

	Dataset-a	Dataset-b
Sparsity	97.82%	90.61%

4.4 Horizontal Comparison

The purpose of horizontal comparison is comparing the performance of different algorithms under the same scenario. For the evaluation metric recall@R, we compare the performance of each algorithm with different R. For metric RMSE, we compare the performance by choosing different values of the feature vector decomposition dimension K. The comparisons above will be experimented on different data sets with different sparsity.

Firstly, aiming at the performance of these four kinds of algorithms on recall@R, we can obviously find that collaborative filtering without auxiliary information performs the worst. The performance difference between DBN and Tag-CF is not significant because both of them apply part of auxiliary information. DLCF performs significantly better than the other three algorithms because of fully utilizing of the latent information. Meanwhile, we can also find out that these algorithms perform better in the dense data sets than in the sparse ones.

Next, evaluating the RMSE for the four algorithms, we find that the situation is quite similar to that of recall@R. DLCF still outperforms the rest three. Similarly, these algorithms perform worse in the sparse data sets than in the dense ones. We can also find that, on the dense data sets, the performance difference between DBN and Tag-CF is not so significant compared with the one on the sparse data sets. Together with above horizontal comparison, we can find that the key of the performance of the algorithm is whether the algorithm can use existing data of the latent information as much as possible.

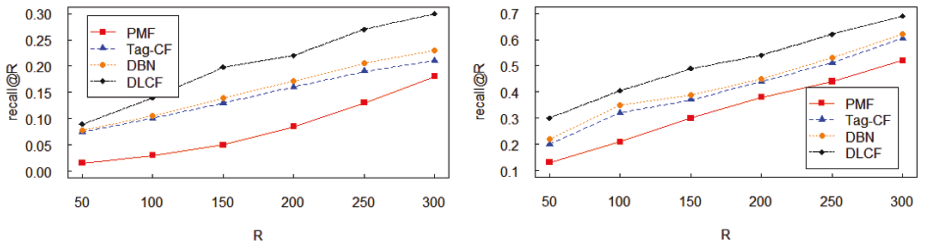

Fig. 4. Recall@R Comparison (dataset-a vs dataset-b)

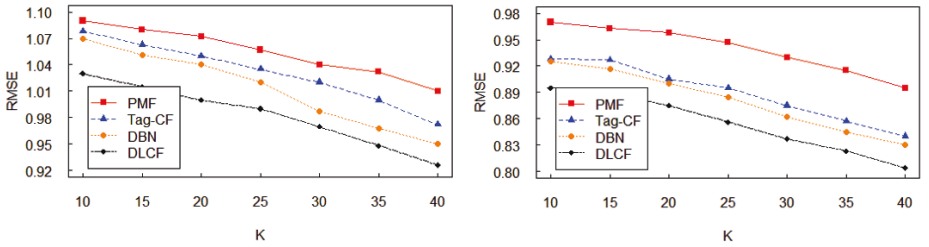

Fig. 5. RMSE Comparison (dataset-a vs dataset-b)

4.5 Vertical Comparison

Vertical comparison mainly study the effect of key parameters on DLCF. Through several experiments in vertical comparison, the performance difference based on recall@R is more significant than the one based on RMSE. So this part focuses on showing the experiment results based on recall@R. Considering the characteristics of deep learning and collaborative filtering, after the repeating experiments, this paper chooses middle layer $L/2$ and model parameter λ_n in SDAE as the key parameters.

Firstly, we conduct the experiment aiming at the parameter λ_n. λ_n is used in SDAE to train the middle layer, which is the key parameter to generate new label matrix in DLCF. By observing the result, we can find that the value of λ_n is neither the bigger the better nor the smaller the better, and there is a range where the algorithm performs well. When the value of λ_n is very small (usually less than 1), the algorithm performs badly. In this case, the increase of the value λ_n may improve the performance of the algorithm. However, when the value of λ_n reaches after three digits, the improvement is significant by continuing increasing the value of λ_n. Similarly, the model performs significantly better in the dense data sets than in the sparse ones.

At last, we look at the result from the experiment where we set the middle layer $L/2$ as observed variable in SDAE. It is obvious that the algorithms perform significantly better on the dense datasets than on the sparse ones. For the middle layer, when $L/2$ equals to 1, the performance is worse than the results where $L/2$ equals to 2 or 3 because of the fewer layers. But for the results where $L/2$ equals to 2 or 3, we can find that the performance difference between them is

very small. With the change of the value R, there are ups and downs on both sides. In the deep learning model, each time we add a layer in the middle layers, the complexity will increase, and it will be much harder to call parameters. So in order to ensure the performance of the algorithm, we don't recommend adding too many layers.

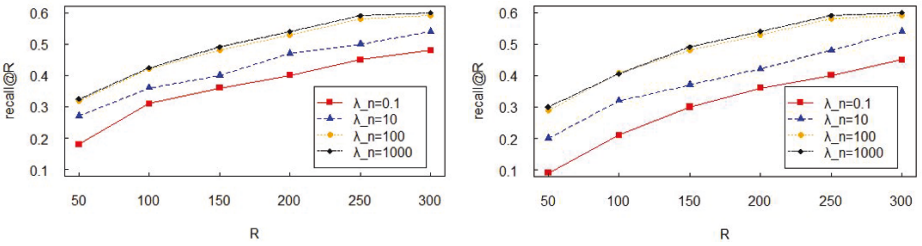

Fig. 6. Recall@R under different λ_n (dataset-a vs dataset-b)

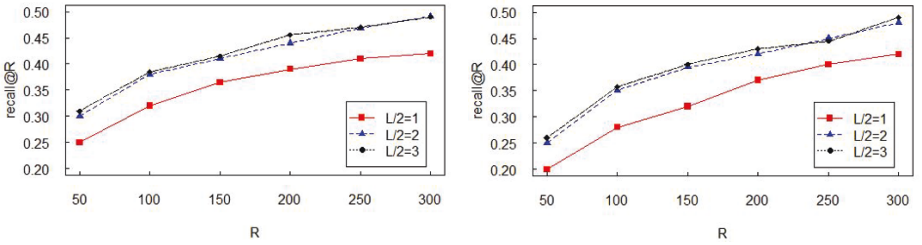

Fig. 7. Recall@R under different $L/2$ (dataset-a vs dataset-b)

5 Conclusions

Conventional collaborative filtering can't overcome the problem caused by sparse matrix. Even if introducing label information, label matrix also has the sparsity problem, which hasn't ideal performance. But conventional content-based recommendation algorithm is not suitable for fusing with collaborative filtering because of its own characteristics. The ability of extracting deep latent information in deep learning model is sufficiently verified in the field of image recognition and natural language processing.

This paper takes advantage of the characteristics of deep learning model and improves the information of original data substantially by processing original label information of items. By combining content-based recommendation model and collaborative filtering, we propose DLCF algorithm and fully apply deep learning model to the recommendation system. Meanwhile, the experiments on real-world data sets show that it can achieve better performance than conventional recommendation model. On the other hand, introducing deep learning

model undoubtedly puts forward higher requirements for the calling of the parameters, and the complexity is much higher than conventional model. Future work will focus on addressing these problems in order to improve the interpretability and engineering significance of the algorithm.

6 Acknowledgment

This work is supported by National Natural Science Foundation of China (61003031, 61202376), Shanghai Engineering Research Center Project (GCZX14014), Shanghai Key Science and Technology Project in IT(14511107902), Shanghai Leading Academic Discipline Project(XTKX2012) and Hujiang Research Center Special Foundation(C14001).

References

1. Kim, B.S., Kim, H., Lee, J., et al.: Improving a Recommendation System by Collective Matrix Factorization with Tag Information. In: Soft Computing and Intelligent Systems (SCIS), 2014 Joint 7th International Conference on and Advanced Intelligent Systems (ISIS), 15th International Symposium on. IEEE, pp.980–984(2014)
2. Grivolla, J., Badia, T., Campo, D., et al.: A Hybrid Recommendation Combining User, Item and Interaction Data. In: Computational Science and Computational Intelligence (CSCI), 2014 International Conference on. IEEE, pp.297–301(2014)
3. Shen , Y., Fan , J.: Leveraging Loosely-tagged Images and Inter-object Correlations for Tag Recommendation. In: Proceedings of the 18th ACM, International Conference on. Multimedia, pp.5–14(2010)
4. Blei, D.M., Ng, A.Y., Jordan M.I.: Latent Dirichlet Allocation. The Journal of Machine Learning Research. 3, 993–1022(2003)
5. Krizhevsky, A., Sutskever, I., Hinton, G.E.:Imagenet Classification with Deep Convolutional Neural Networks. In: Advances in Neural Information Processing Systems, pp.1097–1105(2012)
6. Graves, A., Fernndez, S., Gomez, F., et al.: Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In: Proceedings of the 23rd International Conference on Machine Learning, pp.369–376(2006)
7. Vincent, P., Larochelle, H., Lajoie, I., et al.: Stacked Denoising autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. The Journal of Machine Learning Research. 11,3371–3408(2010)
8. Funk, S.: Netflix Update: Try This at Home. `http://sifter.org/~simon/journal/20061211.html`
9. Salakhutdinov, R., Mnih, A.: Probabilistic Matrix Factorization. In: NIPS(2011)
10. Salakhutdinov, R., Mnih, A.: Bayesian Probabilistic Matrix Factorization. In: Proceedings of the 25th International Conference on Machine Learning, pp.880–887(2008)
11. Hu, L., Cao, J., Xu, G., et al.: Personalized Recommendation via Cross-domaintriadic Factorization. In: Proceedings of the 22nd international conference on World Wide Web, pp.595–606(2013)

12. Li, W.J., Yeung, D.Y., Zhang, Z.: Generalized Latent Factor Models for Social Network Analysis. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI), pp.1705(2011)
13. Vig, J., Sen, S., Riedl, J.: The Tag Genome: Encoding Community Knowledge to Support Novel Interaction. ACM Transactions on Interactive Intelligent Systems (TiiS). 2(13),13(2012)
14. Pirasteh, P., Jung, J.J., Hwang, D.: Item-based Collaborative Filtering with Attribute Correlation:A Case Study on Movie Recommendation. Intelligent Information and Database Systems, pp.245–252. Springer International Publishing(2014)
15. Zhang, B., Zhang, Y., Gao, K.N., Guo, P.W., Sun, D.M.: Combining Relation and Content Analysis for Social Tagging Recommendation. Journal of Software. 23(3),476–488(2012)
16. Kim, B.S., Kim, H., Lee, J., et al.: Improving a Recommendation System by Collective Matrix Factorization with Tag Information. In: Soft Computing and Intelligent Systems (SCIS), 2014 Joint 7th International Conference on and Advanced Intelligent Systems (ISIS), 15th International Symposium on IEEE, pp.980-984(2014)
17. Salakhutdinov R, Mnih A, Hinton G.: Restricted Boltzmann Machines for Collaborative Filtering. In: Proceedings of the 24th International Conference on Machine Learning, pp.791–798(2007)
18. Georgiev, K., Nakov, P.: A Non-iid Framework for Collaborative Filtering with Restricted Boltzmann Machines. In: Proceedings of the 30th nternational Conference on Machine Learning(ICML-13), pp.1148-1156(2013)
19. Wang, X., Wang, Y.: Improving Content-based and Hybrid Music Recommendation Using Deep Learning. In: Proceedings of the ACM International Conference on Multimedia, pp.627–63(2014)
20. Wang, H., Shi, X., Yeung, D.Y.: Relational Stacked Denoising Autoencoder for Tag Recommendation. In: AAAI, pp.3052–3058(2015).
21. Van den Oord, A., Dieleman, S., Schrauwen, B.: Deep Content-based Music Recommendation. In: Advances in Neural Information Processing Systems, pp.2643–2651(2013)
22. Vincent, P., Larochelle, H., Bengio, Y., et al. Extracting and Composing Robust Features with Denoising Autoencoders. In: Proceedings of the 25th International Conference on Machine Learning, pp.1096-1103(2008)
23. Xin, W., Congfu X.. SBMF: Similarity-Based Matrix Factorization for Collaborative Recommendation. In: Proceedings of the 26th International Conference on Tools with Artificial Intelligence, pp.379-383(2014)