2014 IEEE International Conference on Data Mining Workshop

An Improved Collaborative Method for Recommendation and Rating Prediction

Guoyong Cai
Guilin University of Electronic Technology
Guilin, China
e-mail: ccgycai@guet.edu.cn

Rui Lv
Guilin University of Electronic Technology
Guilin, China
e-mail: lxhg2001@gmail.com

Hao Wu
Guilin University of Electronic Technology
Guilin, China
e-mail: wh198910@gmail.com

Xia Hu
Computer Science and Engineering, Arizona State University, Tempe, AZ 85281
e-mail: xia.hu@asu.edu

***Abstract*—User-Item matrix (UI matrix) has been widely used in recommendation systems for data representation. However, as the amount of users and items increases, UI matrix becomes very sparse, which leads to unsatisfactory performance in traditional recommendation algorithms. To address this problem, in this paper, a rating prediction method with low sensitivity to sparse datasets is proposed. This method incorporates tag information and factor analysis approach that has been successfully applied in various areas, to discover the most similar top-N users based on the similarity of users' inner idiosyncrasies. Based on the most similar top-N users discovered, an improved collaborative filtering method is designed for rating prediction and recommendation. Extensive experiments have been done for comparing the proposed method with traditional collaborative filtering and the matrix factorization methods. The results demonstrate that our proposed method can achieve better accuracy, and it is less sensitive to sparseness of datasets.**

Keywords—dynamic dataset; tag system; factor analysis; low sensitivity to sparseness; rating prediction

I. INTRODUCTION

User-Item rating matrix model (UI matrix model) is the most classic and effective data model in the field of recommendation systems, and it is widely used in numerous practical recommendation problems for datasets representation, modeling and processing. Although the size of users and items is usually in a larger magnitude, the ratio of valid ratings of users for items in UI matrix is often very small (less than 1%), which lead to the sparseness problem of UI matrix [1, 2], e.g., the sparseness degree of Netflix dataset is 1.2%, BibSonomy is 0.35% and Delicious is 0.046%, etc.. Therefore, algorithms that can handle sparse data sets are often considered to be more promising. How to make use of sparse UI matrix for recommendation is still an open problem in the field of recommendation.

In recent years, many researchers have studied the sparseness of UI matrix from different aspects, and have proposed some methods to deal with the problem. Existing methods can be mainly classified into two categories. The first category aims to make up for the inadequacy of sparse data by introducing much more related information into the UI matrix [3,4,5]; The second category uses the existing data with matrix decomposition [6,7], linear fitting method [8], etc. to establish the prediction model of the unknown rating data; or uses diffusion [9], iterative optimization [10], transfer of similarity [11], and etc. to discover potential relationship between users and items. However, all these methods still suffer from some problems, e.g., too much auxiliary information involved, too complicated process, and etc. Furthermore, with the increase of data size, UI matrix is even sparser, and the accuracy of traditional methods mentioned above can't bring in positive effect, especially in the dynamically growing datasets of practical applications.

Recently, social tagging, review rating or commenting systems have been increasingly applied. In these systems, users can create or upload content (items), annotate them with relevant tags (reviews or comments), and share them with other users. The whole set of tags constitutes an implicitly unstructured collaborative classification scheme. This implicit classification can then be used to search for and discover items of interest. Therefore, users and items can be assigned profiles defined in terms of weighted lists of social tags (reviews or comments) [12]. By analyzing the items that users have focused on and the lists of social tags of each item, then users' preferences can be discovered or defined. Based on each user's preference distribution achieved, further analysis can be conducted to mine the latent factors to describe users' idiosyncrasies by Factor Analysis that has been widely and successfully used in various areas. The idiosyncrasies of users can be used to cluster the similar users. Based on this observation, we believe that exploiting these tags will be helpful to deal with the sparseness problem. Therefore, in this paper, we propose a method that is less sensitive to sparse data sets based on this assumption. The method uses merely a small number of additional tag information and then sets up User-Tag vectors that reflect users' preferences. Through the in-depth mining for the User-Tag vectors, a special factor model consisting of some important factors describing users' inner idiosyncrasies is built. Finally, these discovered factors are applied in neighbor discovery and rating prediction of each user.

The contributions of this paper can be summarized as the following four-fold. First, we propose to use only a small amount of additional tag information to improve the accuracy

978-1-4799-4274-9/14 $31.00 © 2014 IEEE
DOI 10.1109/ICDMW.2014.60

of the collaborative filtering algorithm. Second, we present a new latent-factor vector to define user's idiosyncrasies and choose user's top-N similar neighbors based on tag information. Third, we propose a framework to incorporate our discovery approach of top-N similar users into the collaborative recommendation framework for recommendation or rating score prediction. Fourth, we have done extensive experiments to demonstrate that the proposed improved method has a strong capability to adapt to very sparse and dynamically growing datasets.

The rest of the work is organized as follows. In Section II, we discuss the related work and Section III describes the preliminary knowledge of Tag System and Factor Analysis. Section IV gives an overview of our method, and then describes the three core components in details in Section V. Section VI presents the experimental results and analysis. Finally, we conclude the paper in Section VII.

II. Related Work

Recommendation methods which are closely related to data sparseness mainly include Collaborative Filtering (CF) and Model-based methods.

Collaborative Filtering method is a kind of typical method based on Collective Intelligence. This method utilizes the past behavior or opinions of other users to predict the contents that active users may be interested in currently; and the core of CF method is to define and find out neighbor (or similar) users. In addition to the traditional methods of computing distance between users, such as Cosine Distance, Pearson Distance, Spearman Correlation Coefficient [13], Manhattan Distance, Mahalanobis Distances [14], and etc., some approaches of defining users' similarity in Data Mining domain have also been introduced, such as User Cluster [15], Graph Partitioning Techniques [16], and etc. However the sparse rating data often leads to few co-rating items among different users. Consequently, the user similarity calculation based on these few co-ratings leads to a low accuracy, which further leads to uncertainty for the recommended results. To solve this problem, scholars have done a lot of studies in recent years. These studies focus on adding relevant information through some ways to make up for data sparseness, which also lead to some improved or new recommendation methods, such as collaborative recommendation based on social relation [17], interdisciplinary and cross recommendation based on topic model [18], dynamic recommendation based on context-awareness [19], self-adaption [20] , and etc. and others like recommendation relying on Trust Relationship [21], Propagation Path [22], Linear Regression Prediction [23], Bayes Network [24], Markov Decision Process [25], Gibbs Sample [26]， Tag System [27], Association Rules [28], Knowledge Reasoning [29], etc..

Model-based method mainly includes Bipartite Graph network architecture model [30], Matrix Factorization model [31]. Bipartite Graph network architecture model implements the personalized recommendation for users through simulating the dynamical processes of complex networks such as Material Diffusion [32] and Heat Conduction [33] on a Bipartite Graph. Not only the accuracy of such an algorithm and its improved algorithms are better than classical Collaborative Filtering algorithms, but also a great improvement is achieved on the diversity of recommendation, such as "Long Tail" recommendation, personalization and other aspects [34]. Matrix Factorization model is very successful and even becomes the mainstream model of recommendation algorithm in recent years. The main idea of Matrix Factorization model is to extract a subset with a small number of potential feature factors from a UI matrix, and then establishes the feature vectors of users and items according to these factors. Finally, some methods like Gradient Descent [35] are used to solve the eigenvectors and construct a rating predictor for unknown items of the original sparse rating matrix. Since the Matrix Factorization method achieved the excellent result in 2009 Netflix contest, people give a great attention to Matrix Factorization on the application of recommender systems and propose some improved algorithms with good effects, such as SGD [36], PLSA [37], SVD++ [38], PMF [39], LFM [40], etc.. Some special analysis tools for Matrix Factorization, such as SVDFeature [41] of Apex Lab from Shanghai Jiao Tong University (SJTU) and LibFM [42] developed by Steffen Rendle et al. are also implemented.

III. Preliminary

In this section, we introduce two important concepts, Tag System and Factor Analysis, which will be used in our method.

A. Tag System

On the web, more and more content (items), especially for multimedia resources, is generated by users. To organize this information and make it accessible via current search technology, tag systems attract increasing attention and have gained tremendous popularity for network resources labeling, organizing and sharing. For example, by evaluating the resources' tags, people can make a better choice for their favorite resources. In general, the attention degree for tags not only has an influence on resource selection but also convey some potential characteristics of users' interests. In collaborative systems, mapping the relationship from User-Item to User-Tag can help us better understand the distribution of users' interests. For example, classifying movies which a user had watched using the different tags of movies, we can create a User-Tag vector between this user and all the movie tags, which reflects the user's interest distribution to each tag. For example, the data set named movieLens 1M (http://www.datatang.com/data/44521) contains tags created by movie fans that in fact reflecting uers' preference. So we can improve collaborative algorithm performance from the perspective of public tags' mining.

B. Factor Analysis

Factor analysis is an extension and development of Principal Component Analysis (PCA) [43] and it is a method to reduce dimensionality in multivariate statistical analysis. It is one of statistical models and mainly used to analyze hidden factors from some statistical data. Factor analysis studies the internal dependence relationship of the Correlation or Covariance Matrix so as to transform many variables to a few factors and reproduce the original relationship between the factors and variables. Currently, Factor analysis gets successful applications in psychology, sociology, economics and other

disciplines [44]. For example, by analyzing the athletes' scores of Decathlon in the sport competition with factor analysis method, we know that some factors such as sprint speed, explosive strength in arms and legs, and endurance play a decisive role in competition results. Based on these different factors, we can classify all the athletes into different categories and have a targeted guidance. Similarly, in the field of recommender system, the analysis for users' interest distribution with factor analysis method could obtain some hidden factors. These factors usually denote the basic and deep idiosyncrasies like a user's character, cognition, attitude, special preference, etc. and determine the outer interest characteristics and performed behavior of a user, especially in some specific circumstance. For example, the inner idiosyncrasies like the emotion of users, cognition, temperament and special preference, could determine what movies may be a user's first choice when he want to select a movie to watch. So we can distinguish different users by their different idiosyncrasies. A fact is that twins have extremely similar preference because they may have extremely similar inner idiosyncrasies. In this paper, our method also tries to find the "Twins" of having similar inner idiosyncrasies and then has a recommendation based on the preferences of their "Twins".

IV. Overview of Our Method

Motivated by the insights we presented in section I as well as the related work of solving data sparseness we discussed in section II, we consider the problem not only based on a small number of additional information but also with a specific matrix factorization named Factor Analysis. We develop a new algorithm which aims to discover neighbors based on users having similar inner idiosyncrasies. Henceforth, we can use the neighbors' collective intelligence to predict ratings. The algorithm mainly consists of four phases: building User-Tag matrix; mining latent factors; discovering neighbors and ratings prediction. Fig. 1 illustrates our framework of the four processing phases.

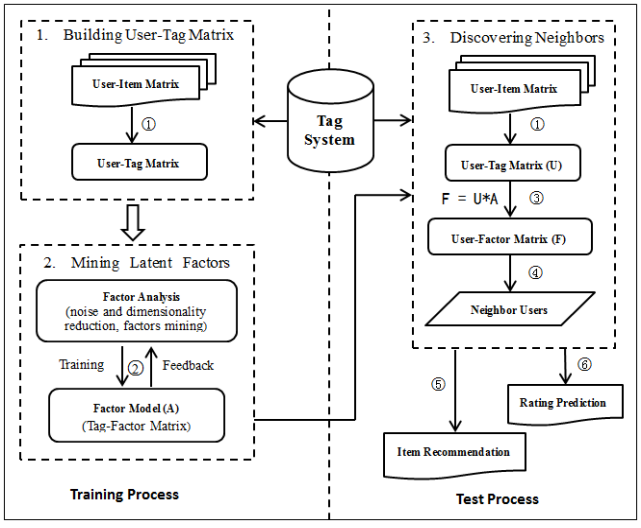

Figure 1. The flow diagram for native algorithm

In Fig. 1, the process ① shows that the User-Tag matrix consisting of each user's interest distribution on all tags is built by mapping the relationship between users and items to users and tags and the details will be discussed in the next section. Then in process ②, the User-Tag matrix is trained and obtains feedback repeatedly by factor analysis. The result of factor model (A) is a Tag-Factor matrix with a lower dimension and less noise in which each column shows a linear combination relationship between a factor and all raw tags. In process ③, the factor model (A) is applied to construct a new User-Factor vector for each user in the test dataset and discover the top-N neighbors (e.g. process ④). The detailed operation is having a matrix multiplication with User-Tag matrix (U) and fag-factor matrix (A). The most similar top-N neighbors are selected according to the similarity or distance between any two User-Factor vectors in the User-Factor matrix (F).

Based on neighbors selected, we can recommend the items that the target user hasn't focused on and his neighbors have scored a high rating (e.g. process ⑤). In addition to that, an expression can also be given to predict missing ratings based on the target user's neighbors (e.g. process ⑥). Equation (1) is an instantiation used frequently for ratings predicting [16]:

$$\widetilde{R}_{ui} = \overline{r}(u) + \frac{1}{k} \sum_{v \in N_k(u,i)} sim(u,v)(r(v,i) - \overline{r}(v)) . \qquad (1)$$

Where $N_k(u, i)$ denotes the k users who belong to the neighbors of u and have a rating on the item i. $sim(u,v)$ is a specified method, e.g. CityBlockSimilarity method, to measure the similarity between two users u and v. Thus, the predicting rating $\widetilde{R}_{ui}$ of user u for item i is computed over the average rating $\overline{r}(u)$ and the sum of u's similarities with its neighbors v, weighted by the deviations of v's ratings for i and average ratings $\overline{r}(v)$.

As is described above, the equation (1) will be used to ratings prediction in our method, and the core problem is latent factors mining and neighbors discovery for each user.

V. Latent Factor Mining and Neighbor Discovery

In this section, latent factor mining and neighbor discovery are introduced in detail. Latent factor mining constructs a factor model and the factor model will be applied in neighbors' discovery.

A. Building User-Tag Vector

Generally speaking, we tend to choose an object (item) we like and have a rating for it, e.g. movies. However, the level of rating is related to the user itself as well as the item. So in order to accurately describe a user's interest distribution, the items having been focused on should be considered and the rating level users made for items will be ignored. In UI matrix, a user's interest distribution based on all tags can be obtained using the following method.

If the rating of user u for item i exists and i consists of n tags, each tag can obtain $1/n$ attention degree. Thus, the attention degree of user u for each tag t can be represented by the following (2) and (3):

$$r_{ut} = \frac{\sum_{i \in D_k(u)} \frac{sgn(u,i,t)}{N(i)}}{k} , \qquad (2)$$

$$sgn(u,i,t)=\begin{cases}1 & t \text{ is a tag of item } i\\ 0 & otherwise\end{cases}. \quad (3)$$

Where r_{ut} stands for the attention degree of user u for tag t. $D_k(u)$ denotes the k rated items of user u. $N(i)$ is the tag amount of item i. When the total number of all tags is p, the attention degree of user u for each tag is expressed as a vector U_u in (4):

$$U_u=(r_{u1},r_{u2},...,r_{up}). \quad (4)$$

B. Latent Factor Mining

Factor analysis is an important method to find out the hidden factors from some observable statistical variables, e.g. the user's interest distribution Uu mentioned above. By building an approximate mapping relationship from raw statistics to the latent factors, we can easily map the new statistical information to the space of latent factor. Compared with the analysis based on raw statistical variables, the analysis based on latent factors has three obvious advantages at least: the first is reducing dimensions, and removing noise; and the second is that the latent factors usually have special physics meaning, which is helpful for us to search for the decisive roles; the last one is having a strong stability to adapt to very sparse datasets because of this method being less dependent on the ratio of valid ratings of users for items in UI matrix.

1) Latent factor model

Assume that $X=(X_1,X_2,X_3,...,X_p)^T$ is an observable statistical variable with p dimension. The latent factor model $A=(a_{ij})_{p*m}$ can be described in the following form:

$$\begin{cases} X_1-\mu_1=a_{11}f_1+a_{12}f_2+...+a_{1m}f_m+\varepsilon_1\\ X_2-\mu_2=a_{21}f_1+a_{22}f_2+...+a_{2m}f_m+\varepsilon_2\\ \vdots\\ X_p-\mu_p=a_{p1}f_1+a_{p2}f_2+...+a_{pm}f_m+\varepsilon_p \end{cases}. \quad (5)$$

Where $X_1,X_2,...,X_p$ are the p observable variables of X; $\mu_i(i=1,2,...,p)$ is the Mathematical Expectation of X_i; $f_1,f_2,...,f_m$ and $\varepsilon_1,\varepsilon_2,...,\varepsilon_p$ $(m<p)$ are respectively the common and the special factors for X_1 to X_p; a_{ij} is the loading of common factors. Each common factor has an effect on at least two statistical variables. If not, these factors are regarded as the special factors. Each special factor appears merely in the expression of the ith statistical variable X_i, that is, it only has an effect on the variable X_i. Equation (5) can be also written in the following matrix form:

$$X=\mu+AF+\varepsilon\approx\mu+AF. \quad (6)$$

Where $\mu=(\mu_1,\mu_2,...,\mu_p)^T$ and $F=(f_1,f_2,...,f_m)^T$ are respectively the Mathematical Expectation of X and the common factor vector; $\varepsilon=(\varepsilon_1,\varepsilon_2,...,\varepsilon_p)^T$ is special factor vector; $A=(a_{ij})_{p*m}$ is the loading matrix of the common factors for the raw statistical variables, which is also the solving factor model.

Therefore, our goal is solving the loading matrix A and the Mathematical Expectation μ.

2) Solving the loading matrix A

Assume that $X_{(1)},X_{(2)},...,X_{(n)}$ is the sample set of the variable X and $X_{(i)}=(x_{i1},x_{i2},...,x_{ip})^T$, then the Mathematical Expectation μ can be estimated using (7):

$$\mu=\overline{X}=\frac{1}{n}\sum_{i=1}^{n}X_{(i)}. \quad (7)$$

To solve the loading matrix A, the Sample Covariance Matrix Σ can be used and estimated using (8):

$$\Sigma=\frac{1}{n-1}\sum_{i=1}^{n}(X_{(i)}-\overline{X})(X_{(i)}-\overline{X})^T. \quad (8)$$

From the property of Factor Analysis method [44] we know that:

$$\Sigma\approx AA^T. \quad (9)$$

Hence, if the eigen values of Σ are $\lambda_1\ge\lambda_2\ge...\ge\lambda_p\ge0$ and the corresponding eigen vector, unit as well as orthogonal, are $l_1,l_2,...,l_p$, then

$$\begin{aligned}\Sigma&=\lambda_1l_1l_1^T+...+\lambda_ml_ml_m^T+\lambda_{m+1}l_{m+1}l_{m+1}^T+...+\lambda_pl_pl_p^T\\ &\approx\lambda_1l_1l_1^T+...+\lambda_ml_ml_m^T\\ &=AA^T\end{aligned} \quad (10)$$

When the sum of last p-m eigen values are much less than the sum of the first p eigen values.

As a result, $A=(\sqrt{\lambda_1}l_1,\sqrt{\lambda_2}l_2,...,\sqrt{\lambda_m}l_m)=(a_{ij})_{p*m}$ is a proximate solution of factor model with the special factor number m.

C. Neighbor Discovery

If the factor model of UI matrix is A (achieved by the process ① in Fig.1) and the User-Tag matrix is U (achieved by the process ② in Fig.1), the User-Factor matrix F is:

$$F=U*A=\begin{bmatrix}\sigma_{11}&\sigma_{12}&\cdots&\sigma_{1m}\\ \sigma_{21}&\sigma_{22}&\cdots&\sigma_{2m}\\ \vdots&\vdots&\ddots&\vdots\\ \sigma_{n1}&\sigma_{n2}&\cdots&\sigma_{nm}\end{bmatrix}. \quad (11)$$

Where $U=\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1p} \\ r_{21} & r_{22} & \cdots & r_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{np} \end{bmatrix}$ and $A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{p1} & a_{p2} & \ldots & a_{pm} \end{bmatrix}$.

In view of the User-Factor matrix, different methods to calculate the distance or similarity of any two row vectors can be used, such as the Manhattan distance, Pearson distance, etc.. According to the distance between two different users, and the most similar top-N user with target user can be found and selected out.

VI. EXPERIMENTS

In order to empirically verify whether the top-N similar neighbors generated by the method proposed in this paper are able to have a more accurate prediction for the missing ratings in UI matrix and have a lower sensitivity for different sparseness of datasets, the following experiments are carried out based on the publicly available dataset MovieLens 1M[1]. This dataset contains 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 MovieLens users in 2000. In the dataset, each movie is depicted by some of 18 tags, e.g. Comedy, Action, etc.. About 50% of the dataset (including 3000 users) is retained for training the latent factor model and the rest 50% (including 3040 users) is for testing.

A. *Experiment 1： Parameters Selection*

In the process of building factor model, in order to determine how many factors should be extracted and whether the users chosen to establish the factor model is enough, the following experiment is made and the experiment result is shown in Fig. 2:

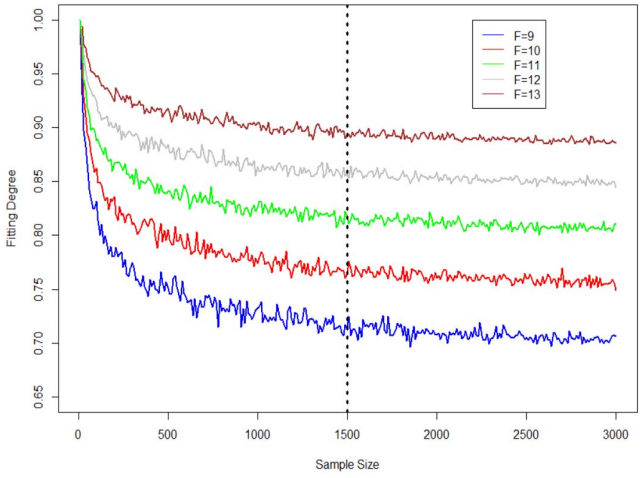

Figure. 2 Relation between fitting degree and sample size

In Fig. 2, the x-axis shows the number of sample users selected at random from the training set and y-axis denotes the fitting degree of the factor model for the sample users chosen. F is the number of factors that is predefined. From the diagram, the fitting degree descends gradually when the number of sample users chosen increases. The reason is that when the factor number is fixed, the difficulty of fitting all sample users increases. However, when the sample size used increases up to 1500, the fitting degree has converted to a constant value. So we can claim that 1500 sample users extracted from the training dataset is sufficient to describe other users. When the sample size is fixed, with factor number increasing, fitting degree also will increase. Because more factors will have stronger ability of fitting more original data, in order to prevent over-fitting, the fitting degree is usually set at 80%. Taking these aspects into consideration, in Fig. 2, the sample users chosen is 1500 and the factor number is 11. The factor model A is shown as follow in Table I:

TABLE I. FACTOR MODEL

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
X1	0.04	0.05	-0.42	0.79	-0.04	-0.28	-0.67	0.85	-0.08	0.25	-0.03
X2	0.67	-0.03	0.05	-0.01	-0.19	0.07	0.00	0.00	0.08	0.57	-0.02
X3	-0.02	0.04	-0.13	-0.02	-0.14	-0.64	-0.04	-0.05	-0.02	0.11	-0.04
X4	-0.83	0.12	0.00	-0.01	-0.13	-0.52	0.10	0.01	-0.14	0.45	0.10
X5	-0.04	-0.18	-0.13	0.07	0.02	0.85	0.03	-0.04	-0.13	-0.01	0.02
X6	0.03	-0.04	-0.02	-0.07	0.04	-0.02	0.04	-0.01	0.09	0.02	-0.06
X7	-0.08	0.01	0.93	-0.02	0.44	0.00	0.09	-0.01	-0.07	0.01	0.92
X8	0.00	0.10	-0.02	0.00	-0.04	0.02	0.01	-0.08	-0.03	-0.07	0.02
X9	0.41	-0.01	0.07	-0.05	-0.11	-0.07	0.02	-0.63	-0.25	-0.08	-0.12
X10	0.10	0.85	0.33	0.01	-0.07	0.01	-0.02	-0.06	0.04	0.02	0.03
X11	-0.01	0.00	-0.07	0.04	0.00	-0.01	-0.01	-0.03	0.02	-0.03	0.92
X12	0.01	-0.07	0.01	0.47	0.00	-0.01	0.00	-0.06	0.00	-0.01	0.06
X13	0.00	-0.02	0.02	-0.02	0.08	0.00	-0.01	-0.02	-0.01	-0.02	0.01
X14	0.99	0.01	0.91	0.10	0.29	-0.01	-0.25	-0.10	-0.18	-0.19	-0.35
X15	-0.16	0.00	0.03	-0.10	0.03	0.03	-0.03	0.00	-0.01	0.11	0.04
X16	0.09	0.00	0.53	-0.11	-0.17	-0.85	-0.22	0.17	0.15	0.02	0.03
X17	0.02	-0.07	-0.04	0.00	0.95	0.02	-0.09	0.00	0.12	0.08	-0.03
X18	0.11	0.03	0.02	0.14	-0.12	0.05	-0.03	0.04	-0.04	-0.18	-0.01

X1-X18 represent raw tags, F1-F11 stand for the eleven latent factors, in which each factor has a linear combination relation with all the raw tags. However different tags have different impact on a special factor.

B. *Experiment 2： Accuracy of Algorithms*

In order to compare the proposed method (Here we named it Native method) with other methods in rating predicting, evaluation criteria of the Root Mean Square Error (RMSE) is adopted, which is a frequently used measure of the difference between values predicted by a model and the values actually observed. A lower RMSE means higher accuracy. The expression of RMSE is shown as the following (12):

$$RMSE=\sqrt{\frac{\sum_1^n (r_i-\hat{r}_i)^2}{n}}. \qquad (12)$$

where r_i is an actual score of an item and $\hat{r}_i$ is the predicted score. n is the sum of items being predicted.

We compare the performance of the proposed algorithm with some other state-of-the-art recommendation approaches. These approaches mainly include three categories: one is the classical CF methods (PearsonCorrelationSimilarity and CityBlockSimilarity), another is the effective methods of Matrix Factorization (SGD [36] and SVD++ [38]) as well as the Random method which is used to be as a basic standard. The methods of PearsonCorrelationSimilarity (Pearson) and

[1]Available at http://www.datatang.com/data/44521

CityBlockSimilarity (CityBlock) compute users' similarity directly with the User-Item vector in UI matrix and then predict ratings based on the neighbors of users.

Some parameters and methods used in our algorithm are shown in Table II, in which the CityBlockSimilarity method calculates users' similarity according to the Manhattan distance of two vectors.

TABLE II. OTHER PARAMETERS AND METHODS

Parameters (methods)	Value
sim(u,v)	CityBlockSimilarity
Ratings prediction in process ⑥	Equation (1)
Value of *N* in process ④	5

All the compared methods in this experiment are derived from Apache Mahout[2]. In CF, the quantity of neighbors referred is equal to our Native method referred, and iterations of matrix factorization method SGD and SVD++ is set at 5000. In order to make the RMSE value is credible, each experiment is repeated 10 times and the final RMSE value is set the average value. Fig. 3 presents the RMSE performance of different algorithms. The Random method performs worse than other algorithms and the Matrix Factorization methods outperform the CF methods and perform worse than Native method. Therefore the method we proposed has lower RMSE value, namely the algorithm accuracy is higher than all other state-of -art methods.

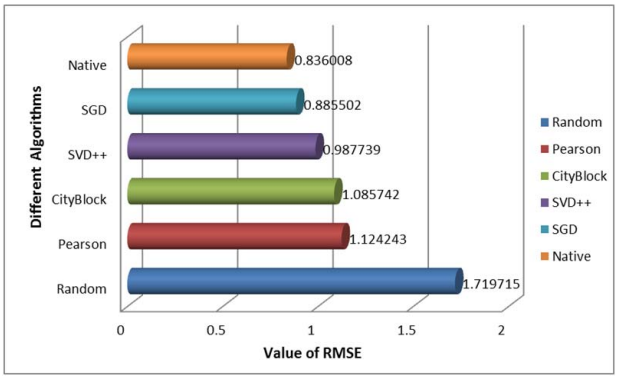

Figure 3. Comparison of different algorithms

C. Experiment 3 : Sensibility of Sparseness

The data sensitivity mainly refers to the influence of sparse dataset on accuracy of recommendation algorithms. In practical application, with items and users added to the dataset continuously, ratings in UI matrix will become more sparse (e.g. the sparseness of UI matrix in Taobao[3] has reached millionth), but there are more ratings for every single user. To validate superiority of our method in dealing with the more sparse ratings matrix, the following comparative experiments are conducted. By selecting some sample users in random, the sparseness degree of raw datasets are respectively reduced to 0.5%, 1.0%, 1.5%, 2.0%, 2.5%, 3.0%, 3.5%, 4.2%, then the accuracy rate of different algorithms were compared under different sparse datasets. The experimental results are shown in Fig. 4:

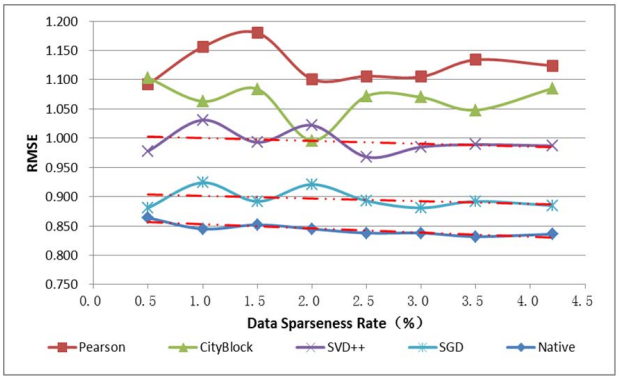

Figure 4. Comparison of sparseness sensitivity

As observed in Fig. 4, the algorithms of Pearson and CityBlock have poor stability and accuracy. However, the three dotted red lines of linear fitting for SVD++, SGD and Native method present that these methods have a little improvement in the aspect of algorithmic accuracy as data sparseness weakened gradually (from 0.5% to 4.2%). However, in practical applications, sparseness degree of UI matrix is increasing, so the accuracy of matrix factorization will also gradually decreases. For the Native method, the accuracy is only associated with the number of each user's ratings and has nothing to do with the sparseness degree of datasets. So, when each user has more ratings, the algorithm accuracy has a small increasing, which has been verified by the lower value of RMSE when having more users ratings (e.g. 4.0%). Therefore, the method proposed in this paper has lower sensitivity for sparse datasets.

D. Analysis of Time Complexity

The time complexity is a very important indicator for recommendation system, and it is closely related to user's experience. The time complexity of our method is: $O(kn+n^2) \approx O(n^2)$, where $O(n^2)$ lies in the cost of computing similarity between different users. Due to dimension reduction of the item, our method has better performance in speed compared with other methods that measures similarity directly based on user-item vector, the improvement ratio is:

$$k = \frac{N(Item)}{N(Factor)} . \qquad (13)$$

Where $N(Item)$ and $N(Factor)$ represent the number of items and potential factor respectively.

VII. CONCLUSIONS

Comparing with traditional methods that combine ratings (i.e. for numerical scores) with user reviews (i.e. for text), the method proposed in this paper has obvious differences. It

[2]Available at http://mahout.apache.org/
[3]Available at http://www.taobao.com/

firstly has a deeper mining for each user's surface interest and then finds out the most important latent factors, at the same time the noise data is filtered. As a result, native method has a higher accuracy. Excepting for the dataset named Movielens 1M mentioned above,the method can also be applied to multiple types of data sets with semantic gap and the very high dimensionality of tag space, such as video, music recommendation, goods recommendation based on user comments, and so on, because of the method having the same theory and a similar process.

When dealing with dynamic data sets in traditional recommendation algorithm, it is inevitable to suffer from the problem of data sparseness. To tackle this problem, we propose a method with low sensitivity to the sparse datasets in the paper. The key idea of the proposed method is to use the tag information of items to construct a factor model for better user neighbors' discovery in rating score prediction while solving the sparseness problem of a dynamic data set. The experiments show that our Native method performs much better in rating accuracy for sparse datasets comparing with the state-of-the-art methods. However, our method assumes that the distribution of user's interest remains stable in a short time, consequently gains better performance on rating prediction compared to traditional recommendation methods. When users' interests change fast in a short time, there is still a large limitation for our method. Accordingly, our future work is to model the process of dynamic changing of users' interests.

ACKNOWLEDGMENT

This work is supported by Guangxi Key Lab of Trusted Software under grant #kx201202, Guangxi Science Foundation under grant #2011GXNSFA018156 and the Innovation Project of Graduate Student under Grant No. GDYCSZ201464.

REFERENCES

[1] Sarwar B, Karypis G, Konstan J, et al. Analysis of recommendation algorithms for e-commerce[C]//Proceedings of the 2nd ACM conference on Electronic commerce. ACM, 2000: 158-167.

[2] Lam X N, Vu T, Le T D, et al. Addressing cold-start problem in recommendation systems[C]//Proceedings of the 2nd international conference on Ubiquitous information management and communication. ACM, 2008: 208-211.

[3] Basu C, Hirsh H, Cohen W. Recommendation as classification: Using social and content-based information in recommendation[C]//AAAI/IAAI. 1998: 714-720.

[4] Bao J, Zheng Y, Mokbel M F. Location-based and preference-aware recommendation using sparse geo-social networking data[C]//Proceedings of the 20th International Conference on Advances in Geographic Information Systems. ACM, 2012: 199-208.

[5] Zhang Z K, Zhou T, Zhang Y C. Personalized recommendation via integrated diffusion on user–item–tag tripartite graphs[J]. Physica A: Statistical Mechanics and its Applications, 2010, 389(1): 179-186.

[6] Sarwar B, Karypis G, Konstan J, et al. Application of dimensionality reduction in recommender system-a case study[R]. Minnesota Univ Minneapolis Dept of Computer Science, 2000.

[7] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems[J]. Computer, 2009, 42(8): 30-37.

[8] Jacobs D W. Linear fitting with missing data for structure-from-motion[J]. Computer Vision and Image Understanding, 2001, 82(1): 57-81.

[9] Huang Z, Chen H, Zeng D. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1): 116-142.

[10] Ren J, Zhou T, Zhang Y C. Information filtering via self-consistent refinement[J]. EPL (Europhysics Letters), 2008, 82(5): 58007.

[11] Sun D, Zhou T, Liu J G, et al. Information filtering based on transferring similarity[J]. Physical Review E, 2009, 80(1): 017101.

[12] Cantador I, Bellogín A, Vallet D. Content-based recommendation in social tag systems[C]//Proceedings of the fourth ACM conference on Recommender systems. ACM, 2010: 237-240.

[13] Zar J H. Significance testing of the Spearman rank correlation coefficient[J]. Journal of the American Statistical Association, 1972, 67(339): 578-580.

[14] Mark H L, Tunnell D. Qualitative near-infrared reflectance analysis using Mahalanobis distances[J]. Analytical Chemistry, 1985, 57(7): 1449-1456.

[15] Xue G R, Lin C, Yang Q, et al. Scalable collaborative filtering using cluster-based smoothing[C]//Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2005: 114-121.

[16] Bellogin A, Parapar J. Using graph partitioning techniques for neighbour selection in user-based collaborative filtering[C]//Proceedings of the sixth ACM conference on Recommender systems. ACM, 2012: 213-216.

[17] Konstas I, Stathopoulos V, Jose J M. On social networks and collaborative recommendation[C]//Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval. ACM, 2009: 195-202.

[18] Wang C, Blei D M. Collaborative topic modeling for recommending scientific articles[C]//Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2011: 448-456.

[19] Karatzoglou A, Amatriain X, Baltrunas L, et al. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering[C]//Proceedings of the fourth ACM conference on Recommender systems. ACM, 2010: 79-86.

[20] Lin W, Alvarez S A, Ruiz C. Collaborative recommendation via adaptive association rule mining[C]//Proceedings of the International Workshop on Web Mining for E-Commerce (WEBKDD. 2000.

[21] Walter F E, Battiston S, Schweitzer F. A model of a trust-based recommendation system on a social network[J]. Autonomous Agents and Multi-Agent Systems, 2008, 16(1): 57-74.

[22] Jamali M, Ester M. A matrix factorization technique with trust propagation for recommendation in social networks[C]//Proceedings of the fourth ACM conference on Recommender systems. ACM, 2010: 135-142.

[23] McCullagh P. Generalized linear models[J]. European Journal of Operational Research, 1984, 16(3): 285-292.

[24] Zhang Y, Koren J. Efficient bayesian hierarchical user modeling for recommendation system[C]//Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2007: 47-54.

[25] White D J. Real applications of Markov decision processes[J]. Interfaces, 1985, 15(6): 73-83.

[26] Chib S. Bayes regression with autoregressive errors: A Gibbs sampling approach[J]. Journal of Econometrics, 1993, 58(3): 275-294.

[27] Sigurbjörnsson B, Van Zwol R. Flickr tag recommendation based on collective knowledge[C]//Proceedings of the 17th international conference on World Wide Web. ACM, 2008: 327-336.

[28] Kim C, Kim J. A recommendation algorithm using multi-level association rules[C]//Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC International Conference on. IEEE, 2003: 524-527.

[29] Fan B, Liu L, Li M, et al. Knowledge recommendation based on social network theory[C]//Advanced Management of Information for Globalized Enterprises, 2008. AMIGE 2008. IEEE Symposium on. IEEE, 2008: 1-3.

[30] Fouss F, Pirotte A, Renders J M, et al. Random-walk computation of similarities between nodes of a graph with application to collaborative

recommendation[J]. Knowledge and Data Engineering, IEEE Transactions on, 2007, 19(3): 355-369.

[31] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems[J]. Computer, 2009, 42(8): 30-37.

[32] Liu J G, Wang B H, Guo Q. Improved collaborative filtering algorithm via information transformation[J]. International Journal of Modern Physics C, 2009, 20(02): 285-293.

[33] Zhang Y C, Blattner M, Yu Y K. Heat conduction process on community networks as a recommendation model[J]. Physical review letters, 2007, 99(15): 154301.

[34] Zhou T, Ren J, Medo M, et al. Bipartite network projection and personal recommendation[J]. Physical Review E, 2007, 76(4): 046115.

[35] Baird L, Moore A W. Gradient descent for general reinforcement learning[J]. Advances in neural information processing systems, 1999: 968-974.

[36] Hogwild!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent

[37] Umbrath A S R W W, Hennig L. A hybrid PLSA approach for warmer cold start in folksonomy recommendation[J]. Recommender Systems & the Social Web, 2009: 10-13.

[38] Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C]//Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008: 426-434.

[39] Ma H, Yang H, Lyu M R, et al. Sorec: social recommendation using probabilistic matrix factorization[C]//Proceedings of the 17th ACM conference on Information and knowledge management. ACM, 2008: 931-940.

[40] Agarwal D, Chen B C. Regression-based latent factor models[C]//Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009: 19-28.

[41] Chen T, Zhang W, Lu Q, et al. SVDFeature: a toolkit for feature-based collaborative filtering[J]. The Journal of Machine Learning Research, 2012, 13(1): 3619-3622.

[42] Rendle S. Factorization machines with libFM[J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2012, 3(3): 57.

[43] Wold S, Esbensen K, Geladi P. Principal component analysis[J]. Chemometrics and intelligent laboratory systems, 1987, 2(1): 37-52.

[44] Yi X et al. statistical modeling and R software[J]. Tsinghua University Press, 2006.