Applied Mathematics & Information Sciences 5 (2) (2011), 69S-78S
– An International Journal

An Improved Personalized Filtering Recommendation Algorithm

Sun Weifeng**,** Sun Mingyang**,** Liu Xidong and Li Mingchu
School of Software, Dalian University of 0020Technology, 116620 Dalian Liaoning, China
Email Address: wfsun@dlut.edu.cn,

In the development of Web collection recommendation algorithms surrender quality to the huge and sparse dataset. A memory-based collaborative filtering method increases computational complexity. Obviously sparsity and expensive complexity of computation are trade-offs. In order to settle this problem we propose an improved recommendation algorithm based on collaborative tagging called personalized filtering (PF). PF defines and weighs the feature of tags using 4-D dataset, which can show latent personal interests and long-term personal interests. To decrease the computational complexity, PF constructs a top-N tags set to filter out the undersized dataset. To track the changes of personal interests, PF proposed a novel interest changing algorithm on the 4-D dataset. Some empirical experiments were done and the results shown that the sparsity level of PF is much lower and the computing speed is faster than traditional algorithms.

Keywords: Personalized filtering, collaborative filtering, collaborative tagging, tag.

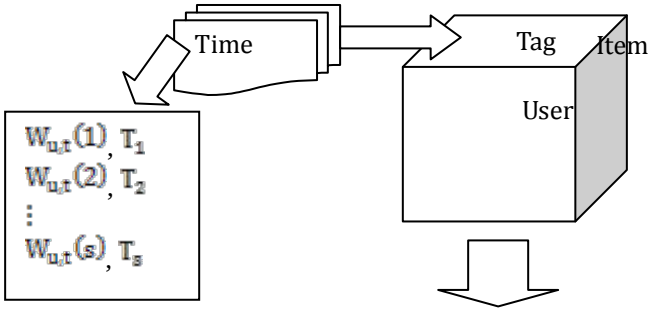
1 Introduction

Recommender system is one of the personalized systems which makes recommendations to target users with unrated or unviewed items based on their histories, such as: Amazon.com developed an item-to-item recommendation algorithm to make recommendations to signed-in users which products they may like [1,2]. Google News is capable of using Probabilistic Latent Semantic Indexing (PLSI) [3] with Dynamic Datasets to recommend news to users which they may want to read [4]. These systems, which provide users personalized services, are all built of models based on collaborative filtering (CF) [5]. It is one of the most successful technologies for recommender systems, including two traditional methods: user-based CF and item-based CF. However, there are several limitations in CF: cold starter, the expensive complexity of computation, sparsity and the quality of recommendation [6,7].

A number of studies have attempted to address problems related to CF. Some of them use Latent Semantic Indexing (LSI) or Singular Value Decomposition (SVD) to reduce the dimensionality of the sparse dataset. It does reduce the sparsity and computational complexity but it also lowers the quality of recommendation. Some of them are focusing on predicting values in the matrix (user-item) to solve the problem of sparsity and cold starter, but it is much more computationally expensive than the previous methods and increases the unstable elements to recommendation quality.

Besides, since Golder and Huberman [8] developed Collaborative Tagging (CT) system, more studies have gone further in mining personal preferences with tags. Collaborative tagging is a practice which allows any user freely to attach keywords or tags to items [8]. It is an inspiration for researchers to improve a recommendation algorithm with collaborative tagging, also called "folksonomy" [9]. Del.icio.us and Yahoo! MyWeb allow users to tag any items with tags are well-known collaborative tagging systems with millions of users. We crawl the real dataset from Del.icio.us, which is a three-dimensional data cube of collaborative tagging system, for experiments. The use of this dataset can increase the recommendation quality because of the better understanding of users' latent preferences. However, it is still computationally expensive and sparse. By far sparsity and the expensive complexity of computation are still trade-offs. In this paper we propose an improved PF recommendation algorithm to solve the trade-off problems. PF firstly reduces the sparsity and the complexity of computation to recommender Top-N interested items to user; secondly feature weighing in the user-tag matrix with consideration of time and popularity of tags scales the influence of outdated interests, which increases the quality of recommendation; thirdly predicting unrated values from neighbors' information eliminates cold-starter. The rest of this paper is organized as follows: in Section 2 we describe our PF recommendation algorithm specifically. In Section 3 the results of empirical experiments compared with previous algorithms present the effectiveness of our algorithm. Finally we conclude with a discussion and future direction of our work in Section 4.

2 PF Algorithm Based on Collaborative Tagging



Tag User	t_1	t_2	t_3	⋯	t_m
u_1	3		2	2	1
u_2		1	1	4	
u_3				1	
⋮	1	1			
u_l		2			3

User-tag matrix W

Tag Item	t_1	t_2	t_3	⋯	t_m
i_1		4		6	
i_2				5	1
i_3		2	2		
⋮	1	1			
i_n			3	4	1

Item-tag matrix R

Item User	i_1	i_2	i_3	⋯	i_n
u_1	5		4		
u_2		2			2
u_3	1	1			
⋮		1	2		
u_l		1	3	4	5

User-item matrix P

Figure 2.1: 4-D dataset for PF recommendation algorithm

Figure 2.1 shows the 4-D dataset for improved PF recommendation algorithm. When one considers the influence of time on personal preference, it is necessary to add an additional dataset of time to each tag by user u to make up a 4-D dataset. There are 4 datasets to show the information--user-tag matrix, item-tag matrix, user-item matrix and tag-times dataset. In Figure 3.1 the $l \times m$ user-tag frequency matrix W contains a list of signed-in users, $U = \{u_1, u_2, u_3, u_4, \ldots, u_l\}$, and a list of tags, $T = \{t_1, t_2, t_3, t_4, ..., t_m\}$, in which $W_{u,t}$ represents the frequency of user u 's tagging with tag t. The $n \times m$ user-tag frequency matrix R contains a list of items $I = \{i_1, i_2, i_3, i_4, ..., i_n\}$ in which $R_{i,t}$ represents the frequency of users' tagging item i with tag t. The $l \times n$ user-item frequency matrix P represents user-item browsing or rating data, in which $P_{u,i}$ represents how many time user u has tagged item i. In tag-times dataset M , T_k represents how many days have been passed since the kth updating with tag t by user u. T_k is changed by days or hours. The specific judgment of one day can be set by 24 hours or other unit of time. In the application below we specify tag-times dataset.

PF algorithm is going through four steps: (I) it is to weight the values in user-tag matrix with separate time-span; (II) CTS is modelled; (III) Neighborhood is formed based on CTS; (IV) The last step is to predict the values of user's preference for unrated items.

2.1 Weighting the User-Tag Matrix

The latent preference of users can be showed in the user-tag matrix, but how much does a user like one specific item is hard to detect. Personal interests can be classified into two forms: long-term interests and short-term interests. Long-term interests always work for recommendation, short-term interests, on the other hand, have impact on recommendations in a short period. The frequency of tagging cannot work out the difference of two forms that can cause influential outdated information.

Since the information of how many times user u has attached tag t to items, how often user u tags with tag t and how popular one tag is, it is not hard to see the relationship between users and the variation of personal interests. Time is a parameter to control the influence of short-term interests and long-term interests, similar to α which acts as a parameter of controlling the influence of $\tau_{i,j}$ in the Ant Colony Optimization. The factor of controlling the influence of short-term interests and long-term interests Q_{Ts} is given by:

$$Q_{Ts} = (p)^{Ts} \tag{2.1}$$

where Ts is time spent since once tagging, p is the element of limitation similar to the element d in PageRank Algorithm which is usually 0.85 estimated from the frequency

that an average surfer uses his or her browser's bookmark feature. ($0 < p < 1$). In our study p is also set 0.85 and the experimental data shows that it is applicable and efficient.

In addition some tags are too popular to indicate latent preferences of users. Inverting tag frequency is one of methods to solve this problem, which is given by $\log\frac{N}{n_t}$, where N is the number of total users and n_t is the number of users who tagged with tag t. However, we only care about the most popular ones and so inverting tag frequency can work only when $\frac{N}{10} \le n_t \le N$. Thus the feature weighting in user-tag frequency $W_{u,t}(s)$ is given by:

$$W_{u,t}(s) = \begin{cases} [\sum_{i=1}^{s-1}(p)^{T_i} R_i + (p)^{T_s} R_s] * \log\frac{N}{n_t} & (\frac{N}{10} \le n_t \le N) \\ \sum_{i=1}^{s-1}(p)^{T_i} R_i + (p)^{T_s} R_s & (0 \le n_t \le \frac{N}{10}) \end{cases} \quad (2.2)$$

where p is the factor of restriction which lowers the impact of short-term interests and enlarges the influence of current interests, R_i is the rating or frequency of the i*th* update and T_i represents how long since the i*th* update. $W_{u,t}(s)$ represents the values of the s*th* tagging with tag t by user u. Assuming that the $(s-1)$ *th* is the latest updating and $\frac{N}{10} \le n_t \le N$, $W_{u,t(s-1)}$ is given by:

$$W_{u,t}(s-1) = \sum_{i=1}^{s-1}(p)^{\Delta Ti} R_i * \log\frac{N}{n_t} \quad (2.3)$$

where ΔT_i is the time-span between the *i*th updating and the $(s-1)$th.

$$\Delta T_i = T_i - T_S \quad (2.4)$$

If there is the *s*th updating, $W_{u,t(s-1)}$ was the value in the user-tag matrix before new information. Thus with the *s*th value the value $W_{u,t(s-1)}$ can be turned to be:

$$W_{u,t}(s-1) = (p)^{-T_S} \sum_{i=1}^{s-1}(p)^{T_i} R_i * \log\frac{N}{n_t} \quad (2.5)$$

The value $W_{u,t}(s)$ in the user-tag matrix is changed as long as there is an action tagging with tag *t* by user u. The previous value is also useful for further restriction on short-term interests. So feature weighting for tags is based on the previous value and current update information, which is given by:

$$W_{u,t}(s) = \begin{cases} (p)^{T_s}[W_{u,t}(s-1) + R_s] * \log\frac{N}{n_t} & (\frac{N}{10} \le n_t \le N) \\ (p)^{T_s}[W_{u,t}(s-1) + R_s] + R_s & (0 \le n_t \le \frac{N}{10}) \end{cases} \quad (2.6)$$

where p=0.85, Ts is the latest time-span, R_s is the latest tagging information, $W_{u,t(s-1)}$ is the previous value before updating and $W_{u,t}(s)$ is the value weighted automatically by the system in the matrix.

Obviously, Eq. (2.6) is simpler and saves more space than Eq. (2.2). It only needs to record the latest updating information and the previous value in the user-tag matrix.

2.2 Setting CTS

After preprocessing the user-tag matrix for a target user there are top-*N* tags ordered by weighted values. These tags are put into the CTS (Candidate Tagging Set) to filter neighbors and latently interested items. The value of *N* controls of the size of neighborhood. It should be neither too big nor too small. The results of experiments on the dataset of Del.icio.us and MovieLens conclude that the empirical value of N=3 is an optimal choice. In Del.icio.us there are billions of bookmarks, which number is reduced exponentially by N. (See Tab. 2.1 and 2.2).

Table 2.1: Filtered information by tags (network, api, data, mashup, sensors)

Tags	Bookmarks	Items	Users
network	967,373	27,926	13,041
network, api	4,018	1,544	563
network, api, data	202	86	47
network, api, data, mashup	37	17	9
network, api, data, mashup, sensors	15	1	1

Table 2.2: Filtered information by tags (biology, science, online, genetics, DNA)

Tags	Bookmarks	Items	Users
Biology	314,531	16,217	10,192
biology, science	148,219	4,893	2,369
biology, science, online	1392	459	307
biology, science, online, genetics	68	31	12
biology, science, online, genetics, DNA	23	8	3

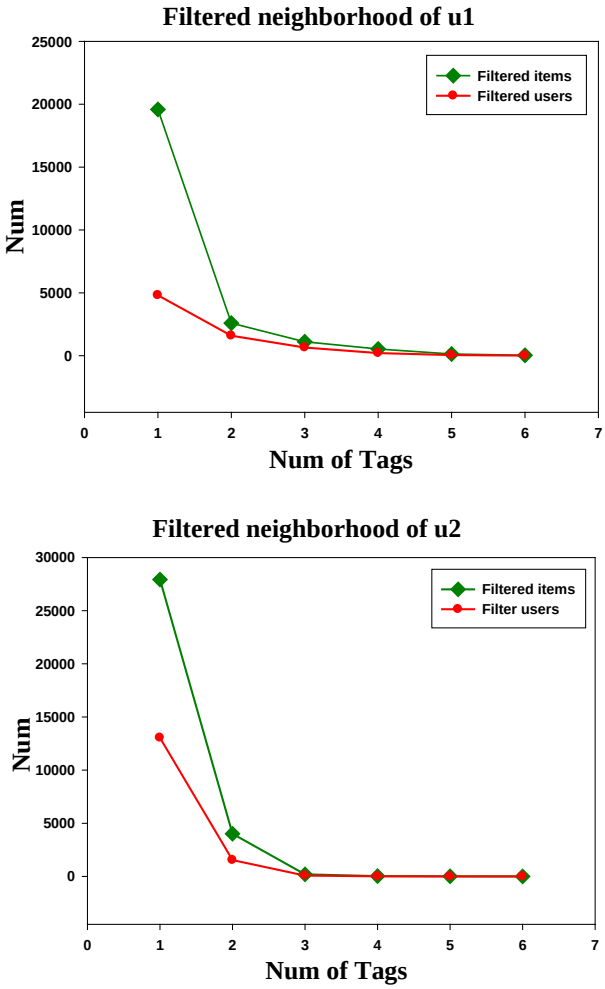


Figure 2.2: Filtered Neighborhood of two users

The data in Fig. 2.2 proves that N=3 is applicable and optimal for filtering out the neighbors. So in our research Candidate Tag Set (CTS) consists of three tags with higher weights. Moreover N can be changed up to the needs of systems.

2.3 Neighborhood Formation and Predict User's Interests

Instead of the cosine method in CF, our PF recommendation algorithm filters the neighbors by CTS based on that the users with the same preference for tags like items with tags in which they are interested. Tags with higher weight are most likely to be of interest to the target user, which may also interest neighbors. CTS acts as a gribble, which filters users who added all of these three tags and items with these three tags. The filtered user-item matrix is shrunk and is not that sparse any more. The user-item matrix is split

into serial collaborative parts. If someone's interests change, he or she is transferred to other parts and does not hurt the balance of the user-item matrix.

The final step of improved PF algorithm is to order the target user's unrated items by prediction scores. The predictive algorithm predicts the value $P_{u,i}$, which represents the probability of user u's preference for item i. $P_{u,i}$ is controlled by the information of neighbors and the correlation with neighbors.

The similarity between the target user and neighbors is given by:

$$sim(u,v) = \frac{\sum_I M_{u,i} M_{v,i}}{\sqrt{\sum_I (M_{u,i})^2} * \sqrt{\sum_I (M_{v,i})^2}} \tag{2.7}$$

where $sim(u,v)$ is the similarity between user u and user v, $M_{u,i}$ is the frequency of user u's browsing item i, and u and v are neighbors. So with the neighbors' information the prediction value $P_{u,i}$ can be:

$$P_{u,i} = \sum_U M_{v,i} * sim(u,v) \tag{2.8}$$

where $P_{u,i}$ is the predictive value of user u for item i. Even though Eq. (2.8) takes neighbors' information into account, it misses the impact of popularity of items. Some items are sufficiently popular that they should be recommended regardless of the lower predictive value. So through enlargement of the impact of popularity of tags $P_{u,i}$ is improved in Eq. (2.9):

$$P_{u,i}' = \sum_U M_{v,i} * sim(u,v) * (\log \frac{n_i}{N+1})^2 \tag{2.9}$$

where n_i is the number of users who have browsed item i and N' is the number of neighbors of target user. $\log(\frac{n_i}{N+1})^2$ is to enlarge the impact of popular items among neighbors.

Unbrowsed Items by the target user are generated in descending order of predictive value $P_{u,i}$. Then the top-N items with higher predictive scores are recommended to the target user u. N is variable; it is up to the different requirements of systems.

3 Evaluations and Analysis

We test the sparsity of the filtered dataset by sparsity level and the quality of recommendation of our PF recommendation algorithm by the value of recall. To compare experimentally the performance of our algorithm with traditional algorithms, user-based CF, item-based CF and CF based on collaborative tagging, we crawl a part of the real

dataset from Del.icio.us, which contains 1,876 million users, 18,521unique bookmarked URLs, 12,037 tags and 31,670 bookmarks.

The sparsity level is measured by d:

$$d = 1 - \frac{Non - zeroentries}{Totalentries} \tag{3.1}$$

The sparsity level of user-based CF, item-based CF, CF based on collaborative tagging and our approach is given in Table 3.1:

Table 3.1: Sparsity level of recommendation algorithms

	User-based	Item-based	CF based on CT	Personalized Filtering
Sparsity Level	0.99908	0.99908	0.99849	0.41592

In Table 3.1 the sparsity of dataset is the average level for our algorithm. It is much lower than the other three methods and much more computationally economic because the computational complexity of the undersized matrix is smaller than $O(M+N)$.

So obviously our approach reduces the sparsity and the complexity of computation of recommender system by personalized filtering and we also ensure the quality of recommendation. Recall is defined as the ratio of relevant items selected to total number of relevant items available, which shows the quality of recommendation. Recall representing the probability that a relevant item is selected is shown in Eq. (3.2):

$$R = \frac{N_{rs}}{N_s} \tag{3.2}$$

where N_{rs} is the number of relevant and selected items and N_s is the number of total selected items. In our experiments N_{rs} is the number of selected items from recommendation list and N_s is the number of filtered items. For user-based, item-based CF and CF based on collaborative tagging, the Recall value is changed by the number of neighborhood formation. Our algorithm does not state the size of neighborhood, which is up to the filtered dataset. Thus the value of Recall of one user is stable compared with the traditional methods. The average value of sample users is shown in Figure 3.1.

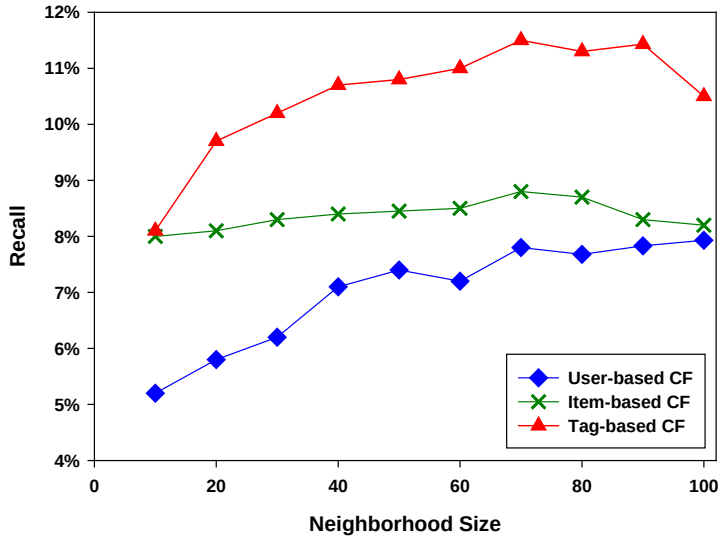


Figure 3.1: Comparison of recall value of Tag-based CF, Item-based CF and User-based CF

Additionally the average recall values of user-based CF, item-based CF, CT-based CF and our proposed method tag-based CF (PF) are shown in Table 3.2. The PF recommendation algorithm shows a higher recall value than the others.

Table 3.2: Average of recall value of recommendation algorithms

	User-based CF	Item-based CF	CT-based CF	Tag-based CF(PF)
Average	0.06896	0.08435	0.10644	0.10915

The data from Table 3.1 and Table 3.2 show that our approach is efficient with little sparseness and high recommendation quality. Moreover, as stated above, the PF method for neighborhood formation is computationally economic by comparison.

4 Conclusions

The improved PF recommendation algorithm based on tagging is proposed and it has advantages in solving sparsity problem and lessens the computational complexity. The improved PF recommendation algorithm can track the change of latent personal interests by using the 4-D dataset and it can give the top-*N* results to users according to their interest. Analysis and experimental results show the advantages of recommendation of PF algorithm, which is much better than user-based CF and item-based CF.

Acknowledgements

This work is supported in part by Nature Science Foundation of China-Japan Science and Technology Agency under grant Project No. 51021140004, partially supported by Nature Science Foundation of China under grants No. 90715037, 61070181 and 60903153.

References

[1] Zi-Ke Zhang, Tao Zhou and Yi-Cheng Zhang, Personalized recommendation via integrated diffusion on user-item-tag tripartite graphs. *Physica A: Statistical Mechanics and its Applications*, 389(1), 2010, 179-186.

[2] Greg Linden, Brent Smith and Jeremy York, Amazon.com recommendations item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1), 2003, 76-80.

[3] Thomas Hofmann, Latent semantic models for collaborative filtering. *Transaction on Information Systems (TOIS)*, 22(1), 2004.

[4] Abhinandan Das, Mayur Datar and Ashutosh Garg, Google news personalization: scalable online collaborative filtering. *The 16th international conference on World Wide Web*, 2007.

[5] J.Ben Schafer, Dan Frankowski, Jon Herlocker and Shilad Sen, Collaborative filtering recommender systems. *The adaptive web: methods and strategies of web personalization*, 2007.

[6] Scott A. Golder and Bernardo A.Huberman, Usage patterns of collaborative tagging systems. *Journal of Information Science*， 32(2), 2006, 198-208.

[7] Badrul Sarwar, George Karpis, Joseph Konstan and John Riedl, Multi-task Learning for Recommender Systems. *The 2nd ACM conference on Electronic commerce*, 2000.

[8] Heung-Nam Kim, Ae-Ttie Ji, Inary Ha and Geun-Sik Jo, Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation. *Electronic Commerce Research and Applications*, 9(1), 2010, 73-83.

[9] Herlocker, J.L, Konstan, J.A. and Riedl, J., Evaluating collaborative filtering recommender systems. *ACM conference on Computer supported cooperative work*, 22(1), 2000.

Weifeng Sun received the bachelor degree in Computer Science from the University of Science and Technology of China (USTC) in 2002 and the PhD degree in Computer Science from USTC in 2007. He is currently an Assistant Professor in Dalian University of Technology (DLUT), supervisor of the graduates. His research interests are in the areas of Wireless and Mobile Network, Distributed Systems and QoS.

Mingyang Sun is currently an undergraduate in Dalian University of Technology (DLUT) and will graduate in 2011. Her research interests are in the areas of Data Mining and Information Retrieval, and Network Security.