2012 26th International Conference on Advanced Information Networking and Applications Workshops

RBRA: A Simple and Efficient Rating-Based Recommender Algorithm to Cope with Sparsity in Recommender Systems

Feng Xie
Department of Automation
Tsinghua University
Beijing, China
Email:
xief10@mails.tsinghua.edu.cn

Ming Xu
Department of Computer Science and Technologies
Beijing University of Posts and Telecommunications
Beijing, China
Email: xum@bupt.edu.cn

Zhen Chen
Research Institute of Information Technology (RIIT)
Tsinghua University
Beijing, China
Email: zhenchen@tsinghua.edu.cn

***Abstract*—Recommender systems have become an important research area both in industry and academia over the last decade. Memory-based collaborative filtering methods include user-based and item-based methods have been explored in many product domains for their simplicity. Memory-based collaborative filtering methods compute the average ratings between similar users or items to predict unrated entries. As a consequence, it is difficult to find similar users or items when the rating data is sparse. The recommendation quality can be poor. This paper proposed an efficient *R*ating-*B*ased *R*ecommender *A*lgorithm named RBRA. With a new model based on user behavior and item features, RBRA can achieve results that are more accurate even when the rating data is sparse. In the prediction phase, RBRA takes an adaptively weighted prediction, which utilizes both ratings of the same item by different users and different items by the same user. The final ratings are evaluated from two sources, user-based and item-based approaches. Experimental results show that RBRA achieves 400% faster recommendation speed with better accuracy.**

Keywords-Recommender Systems, Collaborative Filtering, Similarity Model

1. INTRODUCTION

The pervasive of Web brings an explosive increase of accessible information. It is no doubt that the Web is now so rich that we are overwhelmed by the number of books, movies, goods and restaurants. Recommender systems help users deal with information overload and provide personalized recommendations, content, and services to them. Since the publication of the first paper on collaborative filtering, recommender systems have become an important research area in both industry and academia.

Collaborative filtering is one of the most promising technologies and has been successful in many e-commerce sites by helping users to find their interests. Collaborative filtering aims at predicting the user interest for a target item based on a database of preferences for items by users. Commonly, these preferences either collect from asking users explicitly (e.g. a questionnaire) or trace analysis of the session logs of users. Memory-based and model-based are two most popular approaches to collaborative filtering. Moreover, memory-based, a widely used approach in practice [1, 2], can be divided into user-based approach [1, 3, 4, 5] and item-based approach [6, 7].

Given an unrated entry of a test (or target) item by a test (or target) user to be evaluated, memory-based collaborative filtering tries to find other users similar to target user (user-based), or, other items similar to target item (item-based). Then, the unknown rating is predicted by weighting the known ratings of the target item by similar users, or, the known ratings of similar items by the target user. It is based on this assumption that the similar users have the same interests and the target user will like the items which he/she has selected before.

In memory-based collaborative filtering, both user-based and item-based approaches employ only partial information from the user-item matrix to predict unrated entries, using either correlation between user data or correlation between item data. Memory-based collaborative filtering cannot make good prediction when the data present few co-rated entries between users or items due to the sparsity of data. Therefore, it is meaningful to fuse the ratings from both similar users and similar items to eliminate the dependency on sparse data. Moreover, the similarity is calculated by only co-rated entries, while the deep relationship between ratings isn't taken into account.

In this paper, we propose a simple and efficient *R*ating-*B*ased *R*ecommender *A*lgorithm, RBRA. In RBRA, we define a new model for the computation of similarity, which is based on the statistical characteristics extracted from the ratings related to the target user or target item. Then the similarity is obtained by comparing corresponding model. The experimental results indicate that this model can better represent users' behavior and items' features even when the data is sparse. The final prediction is a weighted combination between individual rating predicted by user-based approach and individual rating predicted by item-based approach. The weight is adjusted according to the entropy available. Experiments show that RBRA can generate more accurate recommendations with less recommendation time.

2. RELATED WORK

Collaborative filtering approaches, as the most successful recommender system, have been explored in many product domains, e.g., movies [21, 22], TVs [23, 25], Web pages [24]. In this section, we briefly introduce some research literatures related to collaborative filtering.

Collaborative filtering approaches can be divided into memory-based and model-based approaches. Memory-based

978-0-7695-4652-0/12 $26.00 © 2012 IEEE
DOI 10.1109/WAINA.2012.11

approach stores all ratings into memory as a user-item rating matrix, and the ratings either collect from asking users explicitly or record log-archives. In the phase of prediction, the similarity between all pairs of users or items will be calculated and stored. Recommendations to the target user are then generated based on the similarity matrix. Some examples about memory-based collaborative filtering include both user-based methods [1, 3, 4, 5], and item-based methods [6, 7]. Because memory-based approaches are easier to be implemented than model-based approaches, so they are more useful in practice. The shortcoming of memory-based approach is the data sparsity problem. In order to address data sparsity, [8, 9, 10] have introduced dimensionality reduction technologies. However, some important information may be discarded, as mentioned in [11, 12], during this reduction.

In the model-based approach, a model would be trained to predict the ratings for items that a target user has not rated before. Such research literatures include aspect model [13, 14], decision tree [3], and latent factor models [15]. These models can solve the data sparsity problem to a certain extent. However, it is difficult to determine the correct parameters which prevented their usage in practice.

Recently, hybrid approaches by fusing memory-based and model-based methods have been proposed [12, 16, 17, 18, 19], to combine the best of the two worlds. However, this unquestionably increases the complexity of recommender algorithm. In [20], Hu and Pu integrated personality characteristic information into recommendation technologies, which can improve the accuracy and effectiveness to a certain extent. However, the difficulty lies in finding a good abstraction to users' behavior and items' features.

In this paper, we define a new model to accurately represent the users or items, with no need of abstraction of user profile (e.g., gender, age, occupation and education) or item feature (e.g., movie title, release date, actor and movie genres). In the prediction phase, we adaptively weighted the ratings from user-based approach and item-based approach.

3. MEMORY-BASED COLLABORATIVE FILTERING

Memory-based collaborative filtering is to estimate ratings for the items that have not been rated by a user. Intuitively, this estimation is based on the ratings given by this user to other items or ratings given by other users to this item, which produce two recommender approaches called item-based and user-based approaches, respectively. Once we can predict the unrated items, the item(s) with the highest predicted rating(s) will be recommended to the target user.

Commonly, the rating-based recommendation problem can be formulated as follows: Let n be the number of users, m be the number of items. $n \times m$ user-item rating matrix represents all the users' and items' preference, where the (i,j)-th entry of this matrix stands for the i-th user rating for the j-th item. Would the i-th user have not rated the j-th item yet, the *null* value is affected to the (i,j)-th entry. Then the recommendation problem is reduced to predict the unrated entries. The most common method is to find the nearest neighbors who have the same preference or find a set of items which similar to the items this user previously rated. The accuracy primarily relies on the computation of similarity between each pair of users or items.

3.1 Similarity Measure

Various methods have been used to compute the similarity $sim(i, j)$ between two objects (users or items) in recommender systems. Most of these methods are based on the ratings of items that both users have rated (user-based) or the ratings of two items that were rated by the same users (item-based). The two most popular methods are *cosine-based* and *correlation-based.* To define them, let S_{ij} be the set of all items rated by both users i and j or the set of all users who have rated both items i and j. The co-rated entries related to object i or j construct a d-dimensional vector. The similarity is measured by computing the cosine of the angle between the vectors. The bigger the value is, the more similar the two objects are:

$$sim(i, j) = \cos(\vec{i}, \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{\|\vec{i}\| * \|\vec{j}\|} = \frac{\sum_{s \in S_{ij}} i_s j_s}{\sqrt{\sum_{s \in S_{ij}} i_s^2} \sqrt{\sum_{s \in S_{ij}} j_s^2}} \quad (1)$$

Where "$\cdot$" denotes the dot-product of the two vectors, and "$\| \|$" denotes the vector module. In item-based case, i_s, j_s is the rating of user s on item i, item j, respectively. In user-based case, i_s, j_s is the rating of user i, user j on the same item s, respectively.

In the correlation-based method, the Pearson correlation coefficient is used to measure the similarity, which is formulated as follows:

$$sim(i, j) = \frac{\sum_{s \in S_{ij}} (i_s - \bar{i})(j_s - \bar{j})}{\sqrt{\sum_{s \in S_{ij}} (i_s - \bar{i})^2 \sum_{s \in S_{ij}} (j_s - \bar{j})^2}} \quad (2)$$

Here $\bar{i}$, $\bar{j}$ are the average of the item (or user) i*'s,* item (or user) j*'s* rating.

Since the similarity between two objects is based on the co-rated entries, it is observed that these two similarity measures will not work well when the dataset is sparse with less intersection.

3.2 Rating Prediction

Prediction computation is the most important step in the user-based or item-based filtering system. After computing the similarities between all pairs of users or items, the kNN-based method is usually implemented to generate predictions. Weighted sum is the widely used to compute the prediction $Ur_{u,i}$ (user-based) or $Ir_{u,i}$ (item-based) on an item i for a user u by computing the sum of the ratings given by the users similar to u on item i or by the items similar to i that rated by the same user u. Each rating is weighted by the corresponding similarity $sim(u, v)$ (or $sim(i, j)$) value between user u and user v (or item i and item j). Intuitively,

ratings with high similarity will contribute more to the prediction. Prediction denoted by $Ur_{u,i}$ or $Ir_{u,i}$ can be computed by:

$$Ur_{u,i} = \bar{r}_u + \frac{1}{\sum_{v \in S(u)} sim(u,v)} \sum_{v \in S(u)} sim(u,v)(r_{v,i} - \bar{r}_v) \quad (3)$$

$$Ir_{u,i} = \frac{1}{\sum_{j \in S(i)} sim(i,j)} \sum_{j \in S(i)} sim(i,j) r_{u,j} \quad (4)$$

Where $S(u)$ ($S(i)$) is the set of users (items) similar to user u (item i).

4. RBRA

In section 3, we have presented user-based and item-based methods in detail. It can be noticed that this two recommendation methods have several shortcomings.

Item-based recommender systems cannot suggest items that the user didn't like before since it is based on the similar items that the user has rated before. In other words, the systems cannot work well when user's preference changes frequently. Another limitation is the new user problem: a user has to rate sufficient number of items so that the systems can understand user's preference to find similar items. Therefore, a new user with few ratings would not be able to get accurate recommendations.

In user-based case, the systems' recommendation is based on the set of users who have similar interest. It is based on this assumption that the users with similar interest will like the same items. Intuitively, it does not have some of the shortcomings that item-based systems have. However, user-based systems still suffers from the new user problem and new item problem. A new user cannot easily find the similar users and a new item with no ratings by any users will not be recommended.

All these problems mentioned above can be reduced to data sparsity. One effective way to overcome this problem is to use user profile information (e.g., gender, age, occupation, education) and item features (e.g., movie title, release date, actor, movie genres) when computing user (or item) similarity. However, these techniques are limited by the features that are explicitly associated with the objects.

In this paper, we implement our approach on MovieLens dataset, which is presented at section 5.3. Here, we propose a simple but efficient recommender algorithm, RBRA, which uses our new similarity computation model. The model focuses on how to make the most of the user-item rating matrix, and extract the valuable information so that accurate recommendations can be made even when the data is sparse. In the prediction phase, weighted combining user-based and item-based methods are used, which can eliminate certain limitation of user-based and item-based methods.

4.1 RBRA Similarity Computation Model

In section 3.1, we introduce two most popularly used similarity measure approaches. *Cosine-based* and *correlation-based* methods both need to find the co-rated entries, $S(u)$ or $S(i)$, and the associated ratings vector stands for each feature. In this case, it suffers from high preprocessing time and data sparsity problems, since it is time consuming to find the co-rated entries and it can't compute similarity or provides results with low accuracy when no or few co-rated entries.

In RBRA, a new similarity computation model is presented, which considers the statistical characteristics of user or item rating vector. The experimental results indicate that RBRA not only overcomes the data sparsity problem, but it can also represent the user or item better. Here, we adopt three statistical values: *mean*, *variance,* and *range*.

In user-based systems, *mean* is the average of all ratings the user rated, which represents user's preference of average rating. *Variance* represents the stability of users' ratings. Finally, *range* is the area that the user usually rate. These three values not only can stand for user's profiles, but also can eliminate different conceptions between different users.

In item-based systems, *mean* represents the average rating of the item. *Variance* indicates the difference among all the users' ratings on the item. Furthermore, *range* is the area of ratings that the item has obtained. Similarly, these three attributes can describe the item's feature better.

Mean, *variance*, and *range* are formulated as follows:

$$\bar{r}_u = \frac{\sum_{i=1}^{n} r_{u,i}}{n} \quad (5)$$

$$S_u = \frac{1}{n} \sum_{i=1}^{n} (r_{u,i} - \bar{r}_u)^2 \quad (6)$$

$$R_u = \max(r_{u,i}) - \min(r_{u,i}) \quad (7)$$

Therefore, in RBRA, we use vector $(\bar{r}_u, S_u, R_u)$ to denote the rating characteristics of user (or item) u. Then, we adopt *cosine-based* similarity computation to calculate the similarity between every pairs of users or items.

4.2 RBRA Rating Prediction

In RBRA, in the phase of prediction, user-based and item-based methods are implemented separately and their results are combined together through weighted sum. This can be defined as:

$$\hat{r}_{u,i} = \alpha \cdot Ur_{u,i} + \beta \cdot Ir_{u,i} \quad (8)$$

Where $\beta = 1 - \alpha$, and α denotes the density of effective ratings during the computation for user-based approach.

In the experiments, the initial values are set to be $\alpha = \beta = 0.5$. Before predicting the unrated entries, the dens-

Table 1

Dataset Size			Rating Sparsity			Rating Distribution				
users	*items*	*ratings*	*sparsity level*	*ratings per user*	*ratings per item*	*ratings of value 1*	*ratings of value 2*	*ratings of value 3*	*ratings of value 4*	*ratings of value 5*
943	1682	100,000	93.695%	106	59	6100	11370	27145	34174	21201

ity of effective ratings of user-based and item-based methods are compared during the computations. Because having more information would generally lead to higher accuracy so the predicted rating with greater entropy should be given higher weight and the other part with lower weight accordingly. The results of the experiments indicate that this adaptively adjusting weight is accurate and efficient. The entropy can be calculated by:

$$I = \frac{\sum_k num_i}{n * k} \tag{9}$$

In user-based case, k is the number of users similar to the target user, num_i denotes the number of rating entries by i-th similar user, and n is the total number of items. In item-based case, k is the number of items similar to the target item, num_i denotes the number of rating entries associated with i-th similar item, and n is the total number of users. Then, α can be determined by:

$$\alpha = 0.5 + \mathrm{sgn}(I_{user-based} - I_{item-based}) \cdot step \tag{10}$$

Where $\mathrm{sgn}(\cdot)$ is the sign function, and $I_{user-based}$, $I_{item-based}$ denote the average amount of information of user-based approach and item-based approach used during the prediction, respectively.

5. Experiment Analysis

In this section, the experiment design, the evaluation metrics and the data set used are described to verify the accuracy and efficiency of RBRA. Then, the experimental results will be given.

5.1 Experiment Design

For testing the accuracy of RBRA, we constructed our experiments to compare the predictive accuracy with user-based and item-based approaches, respectively, on a sparse dataset. The corresponding preprocessing time and predicting time which represent their respective efficiency are also compared. We split the dataset into a training set and a test set. We randomly select 80% of the data for training and use the remaining 20% of the data for testing which is generally adopted in most of literatures.

5.2 Evaluation Metrics

We utilize Root Mean Square Error (RMSE) in our evaluation, which is frequently used for measuring the differences between predicted ratings and the users' real ratings. So it represents the average error rate. The RMSE is the broadly adopted predictive accuracy metric in information retrieval and recommender community. RMSE is formulated as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{size(TEST)} (prate_i - rate_i)^2}{size(TEST)}} \tag{11}$$

Where $prate_i$ denotes the predicted rating, $rate_i$ is the corresponding real rating. And $size(TEST)$ is the number of tested entries.

5.3 Dataset

In our experiments, we only utilize the user-item rating matrix. The redundant information (e.g., users' personality, items' features) is not considered. Currently, most available test datasets for example, MovieLens[1], can satisfy our needs. Therefore, we evaluated our algorithm on the "MovieLens 100k" dataset, which consists of 100,000 ratings of 943 users on 1682 movies, and each user has rated at least 20 movies. Every rating is a positive integer on a 5-star scale. We have divided this data into training and test sets by a random 80%/20% split. Table 1 shows the statistical characteristics of the dataset. The sparsity level of the dataset is computed as [26]:

$$sparsity\ level = 1 - \frac{\#rating\ entries}{\#total\ entries} \tag{12}$$

5.4 Experiment Results

In this section, we present the RBRA experimental results for generating prediction. The results can be divided into four parts: the comparison of our similarity computation model (section 4.1) with *cosine-based* method, *correlation-based* method, respectively, implemented by user-based and item-based approaches; the effect of neighbor size; the selection of the adaptive step in RBRA; and the comparison of preprocessing time. In all of the experiments, the "MovieLens 100k" dataset is used.

5.4.1 Effect of Similarity Methods

We tested three different similarity methods: *correlation-based*, *cosine-based* and *model-based* (our new similarity computation model). We implemented the three similarity computation methods to find k nearest neighborhoods and weighted the ratings by the corresponding similarity to generate the prediction. Here, we set k as 400, since 400 neighborhoods are enough to obtain good performance, experimentally. Then, the test set was adopted to compute Root Mean Square Error (RMSE). The results are illustrated in Fig. 1. It is observed from the results that our new similarity computation model can obtain better performance than *cosine-based* and *correlation-based* methods in both item-based and user-based approaches, which indicates that the new model can better represent user behavior and item characteristics.

5.4.2 Sensitivity of Neighborhood Size

Since the neighborhood size has a significant effect on the quality of prediction [1], to figure out the relationship between the quality of prediction and neighborhood size, we ran an experiment to evaluate RMSE where the number of neighborhood size varied. Fig. 2(a) and Fig. 2(b) have shown the results of user-based and item-based approaches with different similarity computation models respectively. It was observed that the bigger the neighborhood size is, the more accurate the prediction is, both in user-based and item-based approaches to a certain extent. When the k is bigger than certain value, sufficient good prediction can be generated. F-

[1] http://www.grouplens.org/

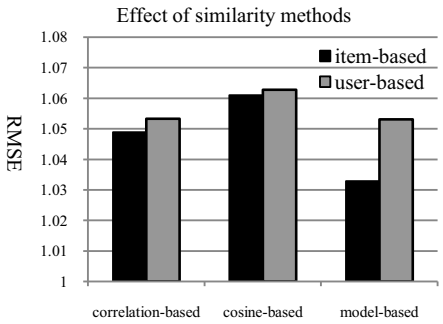

Figure 1: Effect of the similarity computation measure

urthermore, the two approaches (user-based and item-based) with our new similarity computation model have obtained better performance even when the number of neighborhood size is small.

5.4.3 Comparison between Static and Adaptive Weight in RBRA

In this experiment, we fixed *k*_user and *k*_item as 400 to generate the predictions by user-based and item-based methods separately. Then, we combined the two predictions by static weight, which was called Hybrid. In Hybrid, the similarity is calculated by cosine-based method. We varied the static weight from 0 to 1 with the step 0.04. The result is

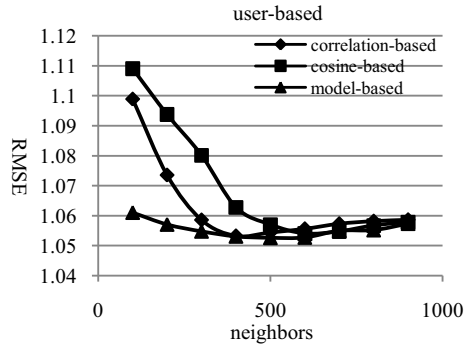

Figure 2(a): Effect of neighborhood size for user-based approach

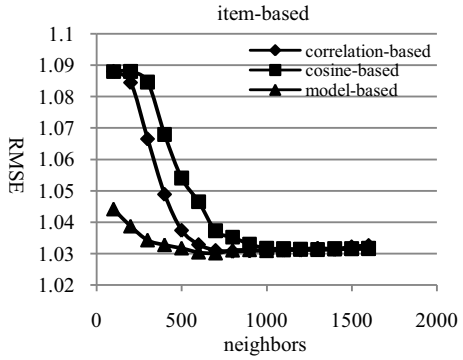

Figure 2(b): Effect of neighborhood size for item-based approach

shown in Fig. 3(a). We can observe that the Hybrid algorithm can obtain best performance when the weight of user-based approach is 0.5 and the value of RMSE is 1.00779. Moreover, in RBRA, we initiated the weight of user-based and item-based methods by equal (0.5). Then, we increased or decreased the weight by the average amount of information (section 4.2) used with a static step, vary from 0 to 0.2, the final results are shown in Fig. 3(b). It is observed that RBRA produces lower prediction error than Hybrid algorithms which combine user-based and item-based methods with static weight and without using our new similarity model, since the maximum value of RMSE in RBRA is 0.996562. Furthermore, we can also observe that when adaptive adjusting weight is equal to 0.1, the predicted error minimized, which reduces the error by 8% compared to Hybrid algorithms.

5.4.4 Efficiency of RBRA

The efficiency of recommender algorithms is a significant guideline to evaluate its performance. In our experiments, RBRA has a lower prediction time than user-based and item-based approaches. Since the similarity computation is based on our new similarity model, it does not require the co-rated entries between all pairs of users or items. Experimentally, RBRA can generate better prediction in 8 seconds, while the prediction time of user-based and item-based approaches are far more than 40 seconds in the same executive environment. RBRA achieves 400% faster speed.

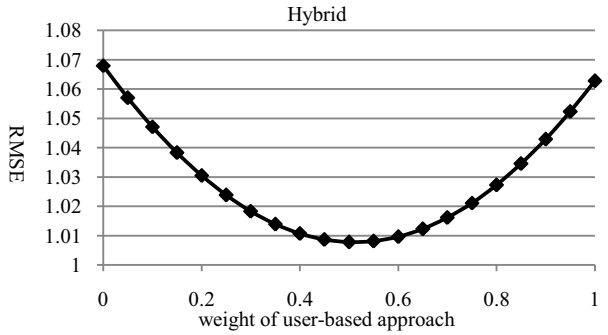

Figure 3(a): The variety of the static weight of user-based approach in hybrid algorithm with *cosine-based* similarity computation

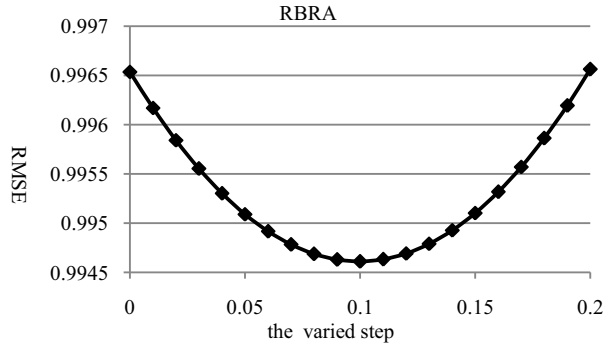

Figure 3(b): The variety of the step in RBRA

5.5 *Performance Analysis*

Our experiments showed that the using of new similarity model can obtain more accurate results even when the neighborhood size is small. However, as the neighborhood size is larger than certain value, the other two similarity computation methods can also get enough information to provide a good prediction. We should point out that our new similarity model can not only generate more accurate prediction, but also obtain more efficient in prediction time. Moreover, our new similarity model and adaptive weight in RBRA helps it perform better than other hybrid algorithms with static weight.

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed a simple and efficient rating-based recommender algorithm, RBRA. It uses a hybrid approach by weighted combining user-based and item-based methods with our new similarity computation model. Experimentally, the recommendation quality in terms of RMSE and the comparison of the preprocessing time show that RBRA generates more accurate recommendations with less prediction error and lower preprocessing time than traditional pure user-based and item-based methods. The extraction of statistical information and adaptive selection of weight make RBRA an efficient algorithm for rating-based recommender systems.

However, user profile and item features are not considered, because it is difficult to describe and abstract, although it contains useful information. This would be taken into account as a future work. Additionally, more information can still be extracted from the user-item rating matrix so that it can better represent user behavior and item characteristics.

ACKNOWLEDGEMENT

This work is supported by Ministry of Science and Technology of China under National 973 Basic Research Program Grant No.2011CD302600, Grant No.2011CB302805, Grant No.2011CB302601, Grant No.2012CB315800, and this work is also supported by China NSFC A3 Program (No. 61161140320). Thanks for the help of professor Jun Li and Jeffrey.

REFERENCES

[1] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In Proc. of SIGIR, 1999.

[2] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing, Jan/Feb: 76-80, 2003.

[3] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proc. of UAI, 1998.

[4] R. Jin, J. Y. Chai, and L. Si. An automatic weighting scheme for collaborative filtering. In Proc. of SIGIR, 2004.

[5] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In Proc. of ACM CSCW, 1994.

[6] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst., 22(1):143–177, 2004.

[7] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In Proc. of the WWW Conference, 2001.

[8] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval Journal, 4(2):133–151, July 2001.

[9] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In Proc. of ICML, 2005.

[10] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of dimensionality reduction in recommender system-a case study. In Proc. of ACM WebKDD Workshop, 2000.

[11] Z. Huang, H. Chen, and D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. ACM Trans. Inf. Syst., 22(1):116–142, 2004.

[12] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In Proc. of SIGIR, 2005.

[13] T. Hofmann. Latent semantic models for collaborative filtering. ACM Trans. Info. Syst., Vol 22(1):89–115, 2004.

[14] L. Si and R. Jin. Flexible mixture model for collaborative filtering. In ICML, 2003.

[15] J. Canny. Collaborative filtering with privacy via factor analysis. In Proc. of SIGIR, 1999.

[16] D. M. Pennock, E. Horvitz, S. Lawrence, and C. Giles. Collaborative filtering by personality diagnosis: a hybrid memory and model based approach. In Proc. Of UAI, 2000.

[17] R. Hu and P. Pu. Using Personality Information in Collaborative Filtering for New Users. Recommender Systems and the Social Web, 2010.

[18] Gediminas Adomavicius and Alexander Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 17, NO. 6, JUNE 2005.

[19] Jun Wang, Arjen P. de Vries and Marcel J.T. Reinders. Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. SIGIR'06, August 6–11, 2006.

[20] Hu, R. and Pu, P. 2010. A Study on User Perception of Personality-Based Recommender Systems. In: P. De Bra, A. Kobsa, and D. Chin (Eds.): UMAP 2010, LNCS 6075, pp. 291–302.

[21] C. Christakou and A. Stafylopatis. 2005. A hybrid movie recommender system based on neural networks. In Proceedings of the 5th International Conference on Intelligent Systems Design and Applications, Wroclaw, Poland.

[22] Bo Yang, Tao Mei, Xiansheng Hua, Linjun Yang, Shiqiang Yang and Mingjing Li. 2007. Online Video Recommendation Based on Multimodal Fusion and Relevance Feedback. In Proceedings of CIVR'07, pages 73–80, Amsterdam, The Netherlands.

[23] M. V. Setten and M. Veenstra. 2003. Prediction strategies in a TV recommender system – method and experiments. In Proceedings of International World Wide Web Conference, Budapest, Hungary.

[24] M. Balabanovic. 1998. Exploring versus exploiting when learning user models for text recommendation. User Modeling and User-Adapted Interaction, 8(4):71–102.

[25] Jonghun Park, Sang-Jin Lee, Sung-Jun Lee, Kwanho Kim, Beom-Suk Chung, Yong-Ki Lee. 2010. An Online Video Recommendation Framework Using View Based Tag Cloud Aggregation. IEEE MultiMedia, IEEE computer Society Digital Library. IEEE Computer Society.

[26] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J.2000. Analysis of recommendation algorithms for Ecommerce. In Proceedings of the 2nd ACM Conference on Electronic Commerce (EC'00). ACM, New York. 285–295.