Speeding Up the Recommender Systems by Excluding the Low Rated Items

Yousef Kilani

Faculty of Prince Al-Hussein Bin Abdallah II for Information Technology, Hashemite University,
Zarqa, Jordan
Email: ymkilani@hu.edu.jo

***Abstract*—Recommender systems (RSs) have become an important tool to help users in searching for their favorite items in many real life applications. A Collaborative Filtering is a commonly used technique in RS. In order to recommend items to the active user (the user we want to make recommendation for), collaborative filtering-based RS uses similar users to the active user and/or latent factor techniques. In our project, we show that excluding the items I that have not been rated high by any user speeds up the recommendation process and gives better accuracy, precision, and recall. The RSs recommend the items that have been rated high by the similar users to the active user. No item from I has been rated high by the similar users and, hence, will not be recommended to the active user. As far as we know, there is no similar work in the literature.**

Keywords–collaborative filtering; recommender systems; speed; low rated items.

I. INTRODUCTION

Recommender systems (RSs) reduce the time users spend while choosing an item in a supermarket, shopping mall, movie shops, songs shops, travel agency, a book in book store etc. While selecting any item, people usually ask others who chose an item from the same category for a recommendation. Currently, the number of items in any category is increasing tremendously and the user has more and more number of choices which makes him confused of which item to choose. Currently, the fast increase of Web 2.0 has led to the proliferation of collaborative websites in which the number of elements that can be recommended (e.g., blogs) can increase significantly when introduced (and not voted) by the users, which generates new challenges for researchers in the field of RSs [1]. RSs are currently being applied in many different domains [2]. Because of the great important of RSs, it is interesting to note that an improvement of 10% recommendation accuracy was awarded with 1 million US dollars [3]. Applications of RSs include: DVD rental provider Netflix [4], the online book retailer Amazon [5], Album Advisor [6], the advisore at the Drugstore [7], the RS for the purchase in eBay [8], an e-learning context [9], e-commerce [10], music recommendation [11], healthcare [12], document recommendation [13][14], stock prediction in the PredictWallStreet company [15], recommending personalised news [16] and e-government [17].

The unnecessary items are the items which have not been rated high by any user. In this research, our interest is in speeding up the RSs by obtaining better precision, accuracy, and recall by excluding the unnecessary items. If the active user A is similar to user B, then the Collaborative Filtering (CF)-based RSs recommend the items to B which have been rated high by A. In other words, the RS will not recommend the items which have not been rated high by user A.

The rest of this paper is organized as follows. We start by introducing the RSs in Section II followed by Section III which shows the related work. Section IV presents our new idea. We incorporate our idea into the Navgaran et al.'s algorithm [18] and we show experimentally the results of the new algorithm using MovieLens dataset in Section V. The last section gives the concluding remarks.

II. THE RECOMMENDER SYSEMS

We can categorise RSs based on the filtering methods into four categories: collaborative [19], content [3][20][21], demographic filtering [22], and hybrid [23][24]. Kilani et al. [25] mentioned that, currently, CF is probably the most known and commonly used recommendation approach in RS [26]. CF is an excellent method for recommendation systems and its core scheme is to calculate the relation among the products and users based on preferences [27]. Two users are similar if they rated a set of items nearly the same.

The CF-based RSs uses either the Neighbourood (NM) [28] or the Model (MM) [29] methods. In the NM, the RS makes recommendation for the active user by finding the users who are similar to this user and then recommending the items which were rated high by these users. The user-based CF RS assumes that if two users A and B rated a set of items similarly and A rated a set of other items, x, high then most probably B will rate x also high. Hence, this RS recommends x for B. In the MM, the RS uses the models to recommend items for the active user. The models learn using supervised or unsupervised learning methods to make recommendation. Examples of these models are: Bayesian hierarchical, clustering, and Latent Factor (LFM). Kilani et al. [25] mentioned that Matrix Factorization (MF) methods have recently received greater exposure mainly as an unsupervised learning method for latent variable decomposition and dimensionality reduction [29].

Lu et al. [30] mentioned that content-based RSs try to recommend items similar to those a given user has liked in the past [31]. These RSs compare between the content of the items the active user rated high and the new items which are considered for recommendations. They then recommend from the new items, those that are most similar to the items which have been rated high by the active user.

The demographic-based RSs recommend items for the active user based on the demographic data like: age, gender,

 ISBN: 978-1-61208-677-4

race, place, material status, education level, etc. This data can be leveraged to power the RSs and hence to improve the accuracy of these RSs.

The hybrid RS merges any two or three techniques of the previous three categories (for instance, see [23][24][30]). Kilani et al. [25] mentioned that MF methods have recently received greater exposure mainly as an unsupervised learning method for latent variable decomposition and dimensionality reduction [29].

III. Related Work

This section presents the related works that have been done to speed up the RSs.

Boim [32] built new techniques to boost CF based RS. Boim focuses on four challenges: improving the quality of the prediction, creating new methods to choose items from the recommended items to the user, improving the efficiency of the CF algorithms and related data structures, and incorporating recommendation algorithms in different application domains.

Bachrach et al. [33] introduced a novel order preserving transformation in order to match the inner product between the active user and the items in the matrix factorization technique to Euclidean space nearest neighbor search problem. Our final solution is based on a novel use of the Principal Component Analysis (PCA)-Tree data structure in which results are augmented using paths one hamming distance away from the query (neighborhood boosting). The end result is a system which allows approximate matches (items with relatively high inner product, but not necessarily the highest one) [33].

Formoso et al. [34] showed how compression techniques using coding techniques from Information Retrieval reduces the size of the rating matrix (up to 75 %) and hence speed up (almost doubling) recommendations.

IV. Proposed Method

This section presents the details of our proposed method. Each user gives a rate of 1, 2,..., or 5 to any item in the system. The higher the value indicates that the user prefers this item more; 1 is the lowest and 5 is the highest. Our idea states that the RS recommends items that have been rated high by similar users to the active user. Therefore, if an item has not been rated high by any user, including the similar users, then no need to consider this item in the recommendation process. This improves the RSs' accuracy. We show in Section V that we may exclude the items which has not given any rate greater than 2, 3, 4, or 5. Our idea can be implemented by any RS.

V. Experiments

We have incorporated our idea into the Navgaran et al.'s algorithm [18] (NA). We have chosen NA since we have implemented its source code while experimenting with other researches. We experiment in this section the results of NA after excluding the items which have not given a rate value greater than or equal to 1, 2, 3, or 4 by any user. We name the new algorithm after excluding items, NAO. We use the MovieLens (downloadable from [35]). MovieLens has 943 users and 1682 movies. It has 100,000 ratings, each from 1-5. Each user has rated at least 20 movies. In our experiments, we used the Visual Basic programming language [36], the Process Explorer Software to measure the CPU time, and 8-GB-RAM i7 PC.

TABLE I: THE NUMBER OF EXCLUDED ITEMS WHEN $minRate =$ 1, 2, 3, 4, and 5.

minRate	Number of Excluded Items	(Number of Excluded Items)/ (Total Items) X 100%
1	0	0%
2	70	0.04%
3	108	0.06%
4	235	0.14%
5	510	0.3%

We split the items into two parts: training data and testing data. We used 80% of the items as training data. The remaining 20% of the items are used for testing. The 20% of the whole items are different for each users. To do so, we pass by every user, we choose randomly 20% of the items that he/she rated and we consider them as not rated.

We ran NA and NAO for 100 times for each user and we take the average of the 100 runs. We used the default parameters of the NA: mutation rate $=$ 0.3, number of latent features $=$ 6, number of generations $=$ 200, crossover rate is equal to 0.8, population size $=$ 50, α = 0.0002, and β = 0.02. We calculated the Mean Absolute Error (MAE) in the same way as calculated by Navgaran et al. [18].

Table I presents the number of excluded items and the percentage of the excluded items to the total items when excluding the items which have not given any rate greater than ($minRate$) 1, 2, 3, 4, or 5. The percentage of the excluded items is equal to (number of excluded items)/(total number of items) X 100%. For instance, the number of excluded items and the percentage of the excluded items to the total items when $minRate = 3$ are 108 and 0.06 respectively. This means that there is no need to consider 108 items while searching for items to recommend to any active user since there is no any user gave any of these items a rate value of 3 or more. In other words, no any user gave a high rate to any of these items and hence NAO will not recommend any of these items to the active user. It is clear in Table I that the number of of excluded items and the percentage of the excluded items are increased as the $minRate$ increased. This happens since the number of items which has not given any rate greater than or equal to 1, 2, 3, 4 is less than the number of items which has not given any rate greater than or equal to 2, 3, 4, 5 respectively.

Tables II and III present the results of NAO and NA respectively. They show the average CPU time in seconds, MAE, accuracy, precision, and recall. In Table II, we excluded the results of NAO when $minRate = 5$ since it happens that many users will not have similar users and NAO cannot give recommendations for the users who has no other similar users.

It is clear from Table II that the average times in seconds taken are decreasing slightly when $minRate$ increased from 1 to 4. The MAE is decreasing as the $minRate$ increased. This happens since the number of items is decreased. The accuracy is increased as the $minRate$ increased since reducing the number of items reduces the search space and hence NAO needs to search fewer number of items as $minRate$ increased. There is a slight change in the precision as the $minRate$ increased. The recall is increased as $minRate$ increases from 1-3. After that the recall is decreased as $minRate$ is increased to 4.

Tables II and III show that the average CPU time taken by

TABLE II: THE RESULT OF RUNNING NAO.

minRate	Avg. Time(s)	MAE	Accuracy	Precision	Recall
1	2,775	2,866,456	0.6	0.67	0.8
2	2,679	2,825,625	0.64	0.6	0.93
3	2,685	2,805,111	0.73	0.71	0.96
4	2,652	2,794,745	0.87	0.7	0.87

TABLE III: THE RESULT OF RUNNING NA.

Avg. Time(s)	2,962
MAE	2,781,442
Accuracy	0.5
Precision	0.4
Recall	0.75

NA (2,962s) is greater than the average CPU time taken by NAO for any value of $minRate$. In addition, it shows that the accuracy, precision, and recall of NAO are better that of NA for any value of $minRate$. But, the MAE of NA is less than the MAE of NAO for for any value of $minRate$. This happens since we have excluded some items and the MAE computed the actual difference between the actual rate and the expected rates.

VI. CONCLUSION AND FUTURE WORK

In this paper, we showed that if an item has not been rated high by any user, then no need to consider this item in the recommendation process. This reduces the time taken for the recommendation process and results in a better accuracy, precision, and recall values. We have proven this idea by incorporating it into Navgaran et al.'s algorithm [18] and testing it using the MovieLens dataset.

Nowadays, the RSs are overloaded with huge number of users and items. Therefore, further research is needed to reduce the number of items and users and to heuristically search for the recommended items in order to enable the RSs give recommendation in a short time.

ACKNOWLEDGMENT

Special thanks to my university, Hashemite University, for providing me the environment to do this research, department head, and my colleagues for their continuous encouragement.

Hashemite University supported us by a research grant to do this research.

REFERENCES

[1] J. Bobadilla, F. Ortega, A. Hernando, and J. Alcala, "Improving collaborative filtering recommender system results and performance using genetic algorithms," Journal of Knowledge-Based Systems, vol. 24, 2011, pp. 1310–1316.

[2] P. Moradi and S. Ahmadian, "a reliability-based recommendation method to improve trust-aware recommender systems," Journal of Expert Systems with Applications, vol. 42(21), 2015, pp. 7386–7398.

[3] X. Yang, Y. Guo, Y. Liu, and H. Steck, "A survey of collaborative filtering based social recommender systems," Journal of Computer Communications, vol. 41, 2014, pp. 1–10.

[4] "http://www.netflix.com," 2018, URL: [accessed: 2018-10-11].

[5] "amazon," 2018, URL: http://www.amazon.com [accessed: 2018-10-11].

[6] "cdnow," 2018, URL: http://www.cdnow.com [accessed: 2018-10-11].

[7] "drugstore," 2018, URL: http://www.drugstore.com [accessed: 2018-10-11].

[8] "drugstore," 2018, URL: http://www.eBay.com [accessed: 2018-10-11].

[9] R. Sikka, A. Dhankhar, and C. Rana, "A Survey Paper on E-Learning Recommender System," International Journal of Computer Applications, vol. 47(9), 2012, pp. 888–975.

[10] J. J. Castro-Sanchez, R. Miguel, D. Vallejo, and L. M. Lopez-Lopez, "A highly adaptive recommender system based on fuzzy logic for B2C e-commerce Portala," Expert systems with applications, vol. 38(3), 2011, pp. 2441–2454.

[11] S. Tan, J. Bu, C. Chen, and X. He, "Using rich social media information for music recommendation via hypergraph model," ACM Transactions on Multimedia Computing. Communications, and Applications, vol. 7(1), 2011, pp. 213–237, article 7.

[12] L. Duan, W. Streat, and E. Xu, "Healthcare information systems: data mining methods in the creation of a clinical recommender system," Enterprise Information Systems, vol. 5(2), 2011, pp. 169–181.

[13] C. Porcel and E. Herrera-Viedma, "Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries," Journal of Knowledge-based Systems, vol. 23(1), 2010., pp. 32–39.

[14] C. Porcel, A. Tejeda-Lorente, M. Martinez, and E. Herrera-Viedma, "A hybrid recommender system for the selective dissemination of research resources in a technology transfer office," Information Sciences, vol. 184, 2012, pp. 1–19.

[15] "predictwallstreet," 2018, URL: http://www.predictwallstreet.com [accessed: 2018-10-11].

[16] E. V. Epure, B. Kille, J. E. Ingvaldsen, R. Deneckere, C. Salinesi, and S. Albayrak, "Recommending personalized news in short user sessions," Proceedings of the Eleventh ACM Conference on Recommender Systems, 2017, pp. 121–129.

[17] X. Guo and J. Lu, "Intelligent E-Government services with personalized recommendation techniques ," International Journal of Intelligent Systems, vol. 22, 2007, pp. 401–417.

[18] D. Z. Navgaran, P. Moradi, and F. Akhlaghian, "Evolutionary based matrix factorization method for collaborative filtering systems," Iranian Conference on Electrical Engineering, 2013.

[19] N. E. I. Karabadj, S. Beldjoudi, H. Seridib, S. Aridhic, and W. Dhifli, "improving memory-based user collaborative filtering with evolutionary multi-objective optimization." Expert Systems with Applications, 2018, pp. 153–165.

[20] M. Volkovs, G. W. Yu, and T. Poutanen, "Content-based neighbor models for cold start in recommender systems ," 11th ACM Conference on Recommender Systems, 2017.

[21] S. B. CL1, C. Ramos, A. Rizo, and L. Fernandez-Luquecor, "HealthRecSys: a semantic content-based recommender system to complement health videos," BMC Med Inform Decis Mak, vol. 17(1):63, 2017.

[22] L. Safoury and A. Salah, "Exploiting User Demographic Attributes for Solving Cold-Start Problem in Recommender System," Lecture Notes on Software Engineering, vol. 1(3), 2013.

[23] H. R. Zhang, F. Min, X. He, and Y. Y. Xu, "A Hybrid Recommender System Based on User-Recommender Interaction ," Journal of Mathematical Problems in Engineering, 2015, article ID 145636.

[24] S. Yanga, M. Korayemb, K. AlJadda, T. Grainger, and S. Natarajana, "Combining content-based and collaborative filtering for job recommendation system: A cost-sensitive Statistical Relational Learning approach," Journal of Knowledge-Based Systems, vol. 136, 2017, pp. 37–45.

[25] Y. Kilani, A. Alsarhan, M. Bsoul, and S. El-Salhi, "Local search-based recommender system for computing the similarity matrix," Int. Journal of Intelligent Systems Technologies and Applications, 2018, forthcoming.

[26] Q. Shambour and J. Lu, "A hybrid multi-criteria semantic-enhanced collaborative filtering approach for personalized recommendations ," Proceedings of the International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT), 2011, pp. 71–78.

[27] B. Jadoon, A. Mansha, F. H. Khan, and S. Bashir, "collaborative filtering based online recommendation systems: a survey," International Conference on Information & Communication Technologies, 2017.

[28] C. C. Aggarwal, "recommender systems: chapter 2 ," Springer International Publishing Switzerland, 2016.

[29] D. Bokde, S. Girase, and D. Mukhopadhyay, “Matrix Factorization Model in Collaborative Filtering Algorithms: A survey ,” Procedia Computer Science, vol. 49, 2015, pp. 136–146.

[30] Z. Lu, Z. Dou, J. Lian, X. Xie, and Q. Yang, “content-based collaborative filtering for news topic recommendation ,” Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015, pp. 217–223.

[31] P. Lops, M. De Gemmis, and G. Semeraro, “Content-based recommender systems: state of the art and trends,” Recommender systems handbook, 2011, pp. 73–105.

[32] R. Boim and T. Milo, “Methods for boosting recommender systems,” 2011 IEEE 27th International Conference on Data Engineering Workshops, 2011.

[33] Y. Bachrach, Y. Finkelstein, and R. BachrachRan, “Speeding up the xbox recommender system using a euclidean transformation for inner-product spaces,” Proceedings of the 8th ACM Conference on Recommender systems, 2014, pp. 257–264.

[34] V. Formoso, D. Fernandez, F. Cacheda, and V. Carneiro, “Using rating matrix compression techniques to speed up collaborative recommendations,” Journal of Information Retrieval, vol. 16(6), 2013, pp. 680–696.

[35] “MovieLens,” 2018, URL: http://grouplens.org/datasets/movielens/ [accessed:2018-10-11].

[36] J. Foxall, “Visual Basic 2015 in 24 Hours,” Sams Publishing, vol. 1st edition, 2016.

 ISBN: 978-1-61208-677-4