A Collaborative Filtering Algorithm Integrating Balance Factor and Time Weight

Jin Zhao(✉) and Jie Sun

Tiangong University, Tianjin 300387, China
j13155513186@163.com

Abstract. Traditional collaborative filtering algorithms still have data sparsity or cold start issues, and generally don't account for changes in user interest over time. This paper proposes two improved measures to address the above problems: a balance factor based on item co-rating users and a time-based weight function, which are combined and applied to the item-based collaborative filtering algorithm to improve the traditional similarity measure. Firstly, the balance factor is calculated using the ratio of the number of co-rated users between items to the total number of users who rated the items, and the influence of the balance factor on the similarity calculation is verified by dynamically changing the value of the parameter α. Then, we introduce a time weighting function to account for the effect of a large time span of user ratings on items on item similarity calculation. Finally, in item-based collaborative filtering, the traditional Pearson correlation coefficient and cosine similarity are improved, and the Movielens dataset is applied for experiments. The experimental results show that the proposed algorithm increases recommendation quality and reduces the average absolute error of the traditional collaborative filtering algorithm.

Keywords: Collaborative filtering · Balance factor · Time weighting function

1 Introduction

With the fast expansion of the Internet in recent years, the number of social network users has grown, and information updates have become more frequent. However, the rapid expansion of network data has made it difficult for users to obtain the information they want promptly and precisely [1]. As a result, information overload occurs. Users' interests may be mined and captured by recommender systems, which then provide them with helpful information. Recommender systems are efficient, scalable, and can alleviate the information overload problem, the major study topic is now concentrated on movie recommendation [2].

Content-based recommendations and collaborative filtering recommendations are two types of recommendation systems. The former suggests comparable things to users based on their previous preferences or purchases, whereas collaborative filtering addresses some of the shortcomings of content-based recommendations by suggesting items based on the assumption that similar people have similar tastes. User-based filtering and item-based filtering are the two types of collaborative filtering. The "people

Q. Liu et al. (Eds.): CENet 2022, LNEE 961, pp. 1094–1102, 2022.
https://doi.org/10.1007/978-981-19-6901-0_114

like you" rationale is used in user-based collaborative filtering, which suggests items to users that are similar to what they enjoy. However, due to the severity of the data shortage, an item may not be evaluated by two users, leaving little or no record to compare user similarities. Collaborative filtering based on objects was created to solve this problem. Item-based collaborative filtering takes the concept of "if you like this, you might enjoy that one as well." It proposes goods that the user has previously enjoyed. The key distinction between the two is the manner in which comparable things (user/item) are acquired.

One of the most extensively used and successful algorithms nowadays is collaborative filtering [3]. However, several researchers have looked into issues including sparsity, early scoring, and cold start. Thakkar et al. [4] proposed a prediction approach that combines user-based collaborative filtering with item-based collaborative filtering using multiple linear regression to decrease prediction errors. Ninan et al. [5] established an efficient trip route framework and used it to an item-based collaborative filtering algorithm that uses the user's past location rating records to compute the route. In an item-based collaborative filtering algorithm, Singh et al. [6] ignored random k-nearest neighbor values and forecasted target items using the most comparable neighbors for each target item, employing Bhattacharyya coefficients to deal with sparse data and increase algorithm prediction accuracy. Hasan et al. [7] developed a similarity measure based on two items, taking into account their symmetry, to optimize the items' cold start problem.

Li et al. [8] advocated that a time function be introduced to give relevant time weights to users' interests in different times and create a time frame that can be constantly changed to better reflect users' short-term interests. A personalized recommendation system is developed that includes social information and a dynamic temporal frame. To tackle the data sparsity problem and increase recommendation accuracy, Suganeshwari et al. [9] employ associative data mining techniques to detect each user's unfavorable items and then fill them with low values. Feng et al. [10] suggested a better similarity model for the problem of users co-rating items in the situation of sparse data, taking into account three aspects impacting similarity: proximity, influence, and popularity, and reducing similarity calculation bias. Introducing a balancing component and offering an enhanced item-based collaborative filtering algorithm, Ajaegbu et al. [11] suggested an improved collaborative filtering approach for existing collaborative filtering techniques with sparse data and cold start problems. By detecting the user's rating propensity using a clustering approach and then conducting rating normalization based on the rating propensity, Kim et al. [12] enhanced the current preference prediction algorithm and increased the accuracy of the recommendation system.

In conclusion, the works mentioned above have enhanced collaborative filtering algorithms to varying degrees. Based on previous research, we propose and implement a collaborative filtering algorithm that integrates balancing factors and temporal weights in item-based collaborative filtering in this study. The method takes into account the effect of user co-rated items, integrates balancing factors to handle the data sparsity problem, and uses a time weighting function to give weights to each user based on the temporal position of the user rated things. The effect of early user ratings on things is reduced during the process of calculating similarity between items. The experimental findings

demonstrate that the suggested approach successfully addresses the data sparsity problem and outperforms the old technique in terms of the average absolute error evaluation measure.

2 Related Work

2.1 Item-Based Collaborative Filtering Algorithm

The item-based collaborative filtering algorithm primarily generates an item-user rating matrix based on the user's rating information of the item, calculates item similarity based on the matrix, obtains item neighbor information, and finally recommends the neighbor with the highest predicted rating among the rated items for the user as the item of interest to the target user. Improving the already existing similarity algorithm can enhance the overall recommendation algorithm for the optimal collaborative filtering method. Cosine similarity, Pearson correlation coefficient, Euclidean distance, and other algorithms are examples of traditional similarity calculation methods. In this study, we primarily use cosine similarity and Pearson correlation coefficient as benchmarks for algorithm improvement, as well as mean absolute error as a measure of algorithm quality.

Cosine similarity (COS) is one of the most often used similarity measures in collaborative filtering recommender systems, and it may be used to assess the degree of relevance between two users or items. The scores of items are stored in vector form, and the computation procedure is as follows: i and j represent two items, u and v represent two users.

$$COS(i,j) = \frac{\sum_{u\in}(r_{u,i})(r_{u,j})}{\sqrt{\sum_{u\in U_{ij}}(r_{u,i})^2}\sqrt{\sum_{u\in U_{ij}}(r_{u,j})^2}} \tag{1}$$

Pearson similarity (PCC) improves on cosine similarity by identifying circumstances when users rate either item i or j at the same time, then locating users who rate both items i and j, and finally adding the average score of the items to the computation. The Pearson correlation coefficient is one of the most often used metrics in collaborative filtering-based recommendation [13]. It properly shows the similarity between items. The relationship can be written as:

$$PCC(i,j) = \frac{\sum_{u\in U_{ij}}(r_{u,i}-\overline{r_i})(r_{u,j}-\overline{r_j})}{\sqrt{\sum_{u\in U_{ij}}(r_{u,i}-\overline{r_i})^2}\sqrt{\sum_{u\in U_{ij}}(r_{u,j}-\overline{r_j})^2}} \tag{2}$$

2.2 Balance Factor

The problem of data sparsity becomes increasingly apparent as the number of users and items in a recommendation system grows. Many items are seldom or rarely rated by users in item-based recommendations, which might substantially influence the accuracy of similarity calculation. An example of item rating is shown in Table 1. It has three products and five users, each having their own set of ratings, with the items not evaluated by users being recorded as 0. The Pearson correlation coefficient was used to calculate

the similarity between the items in Table 1, $Sim(Item1, Item2) = 1, Sim(Item1, Item3) = 0.683$. Due to the obvious high number of co-rated users, the similarity between items 1 and 2 is larger than the similarity between items 1 and 3, therefore this result does not appropriately measure the similarity between items. User 3 was the only user that rated both item 1 and item 2 with the same rating, as indicated in Table 1. This condition will result in two items that must be similar and will have an impact on the accuracy of the advice. As a result, this paper proposes a balancing factor to solve the concerns mentioned above.

Table 1. An example of user rating in the case of sparse data

	Item1	Item2	Item3
User1	1	0	4
User2	5	0	5
User3	4	4	4
User4	0	5	0
User5	0	1	4

As a result, this paper proposes a balancing factor that is tailored to the problem at hand. The number of common users assessing items i and j is first determined and designated as N. M represents the total number of users who have rated item i or item j. The ratio of N to M represents the influence of the number of co-rated users on the similarity between items. Finally, to optimize the similarity computation, a balance factor is employed. The following is the exact formula.

$$sim'(i,j) = \left[1 - \alpha\left(1 - \frac{N}{M}\right)\right] sim(i,j) \tag{3}$$

where, the parameter α belongs to the range [0, 1], indicating the balance factor's effect coefficient; the higher the value, the greater the weight of the balance factor. The original similarity calculation is the same when $\alpha = 0$.

Although the above technique improves similarity calculation by weighting the number of co-rated users per pair of items, the derived similarity values' confidence remains low for a small number of co-rated users. As a result, we consider enhancing the similarity with low confidence and refining the balancing factor in this paper. To begin, a threshold $\overline{N}$ is defined, which represents the average of the number of users who rate an item as having something in common with other items; if the number of common users of two items is greater than or equal to the threshold, the similarity is considered trustworthy; otherwise, it is not trustworthy or less trustworthy. The items are then transformed to trustworthy status using the calculation below, based on the overall number of rated users.

$$P_i = |N - M_i| \tag{4}$$

where, M_i is the total number of users who have rated item i and N is the number of users who have rated both item i and item j. A new similarity formula is derived by combining

Eqs. (4) and (3).

$$sim^{'}(i,j) = \begin{cases} [1-\alpha(1-\frac{N}{M})]sim(i,j), N \geq \overline{N_i} \; and \; N \geq \overline{N_j} \\ \left[1-\alpha\left(1-\frac{P_i}{M_i}\right)\right]sim(i,j) + \left[1-\alpha\left(1-\frac{P_j}{M_j}\right)\right]sim(i,j), \quad otherwise \end{cases} \tag{5}$$

2.3 Time Weight

Time information is a very important kind of information in recommender systems, and users' preferences change with time, and the ratings are typically more impactful on users' future selections the closer they are to the current time [14]. Users' most recently visited items are more likely to reflect their interests, whereas early evaluation data should be given less weight. To enhance the calculation of item similarity, this paper also introduces a temporal weighting function based on item assessment and combines it with a balancing factor.

The long-term assessment information of users is time-sensitive in the process of user interest-based recommendation, and the influence of earlier assessments on the present predicted value is inversely proportional to the time span [15]. In this research, the traditional time function is improved and applied to item-based collaborative filtering to adequately reflect the influence of time on recommendation results. Table 2 illustrates the process symbol.

Table 2. Symbol explanation table for the time weighting function

Symbol	Definition
$t_{u,i}$	Time for item i to be rated by user u
$max(t_i)$	The latest time item i was rated by the user
$min(t_i)$	The earliest time item i was rated by the user

The final obtained time weighting function is formulated as follows.

$$\omega_{u,i} = \frac{1}{1+\exp\left(-\frac{t_{u,i}-min(t_i)}{\max(t_i)-min(t_i)}\right)} \tag{6}$$

where, the time weighting function is monotonically growing and lies between (0,1). As a result, users' recent ratings of items have more weight, whereas users' early ratings carry less weight.

By combining the balance factor and the time weighting function, the traditional cosine similarity and Pearson correlation coefficients are improved in this paper and used to the item-based collaborative filtering algorithm. Equations (7) and (8) reveal the final similarity calculation formula. Then, in order of similarity, k items are chosen as neighbors, and the user's rating of the recommended items is predicted using Eq. (9) [16], and finally, the user's interested items are recommended.

Improved cosine similarity:

$$NSim(i,j) = \begin{cases} [1-\alpha(1-\frac{N}{M})]sim^1(i,j), & N \geq \overline{N_i} \text{ and } N \geq \overline{N_j} \\ \left[1-\alpha\left(1-\frac{P_i}{M_i}\right)\right]sim^1(i,j) + \left[1-\alpha\left(1-\frac{P_j}{M_j}\right)\right]sim^1(i,j), & otherwise \\ sim^1(i,j) = \frac{\sum_{u\in U_{ij}}(r_{u,i}\cdot\omega_{u,i})(r_{u,j}\cdot\omega_{u,j})}{\sqrt{\sum_{u\in U_{ij}}(r_{u,i}\cdot\omega_{u,i})^2}\sqrt{\sum_{u\in U_{ij}}(r_{u,j}\cdot\omega_{u,j})^2}}, \end{cases} \tag{7}$$

Improved Pearson correlation coefficient:

$$NSim(i,j) = \begin{cases} [1-\alpha(1-\frac{N}{M})]sim^2(i,j), & N \geq \overline{N_i} \text{ and } N \geq \overline{N_j} \\ \left[1-\alpha\left(1-\frac{P_i}{M_i}\right)\right]sim^2(i,j) + \left[1-\alpha\left(1-\frac{P_j}{M_j}\right)\right]sim^2(i,j), & otherwise \\ sim^2(i,j) = \frac{\sum_{u\in U_{ij}}(r_{u,i}\cdot\omega_{u,i}-\overline{r_i})(r_{u,j}\cdot\omega_{u,j}-\overline{r_j})}{\sqrt{\sum_{u\in U_{ij}}(r_{u,i}\cdot\omega_{u,i}-\overline{r_i})^2}\sqrt{\sum_{u\in U_{ij}}(r_{u,j}\cdot\omega_{u,j}-\overline{r_j})^2}}, \end{cases} \tag{8}$$

The predictive rating formula is as follows:

$$r_{u,i} = \overline{r_i} + \frac{\sum_j sim(i,j)(r_{u,j}-\overline{r_j})}{\sum_j sim(i,j)} \tag{9}$$

where, $sim(i,j)$ represents the similarity between item i and item j, $r_{u,j}$ the rating of item j by user u, and $\overline{r_i}$ is the average value of item i as rated by users.

3 Experimental Results and Analysis

3.1 Experimental Data

The MovieLens-100k dataset generated by the GroupLens team at the University of Minnesota was used for this paper to evaluate the algorithm's performance. This dataset contains 100,000 ratings data from 943 people for 1682 movie items, as seen in Table 3, represented as integers from 1 to 5. The greater the rating value, the more the user enjoyed the movie. The sparsity of this dataset is: 1 - 100,000/946 × 1682 = 0.937. The sparsity of the dataset increases as it approaches 1. In this paper, the dataset is separated into a training set and a test set in the ratio of 8:2 in order to evaluate the algorithm's recommendation results.

3.2 Evaluation Metrics

The assessment metrics reveal the algorithms' strengths and shortcomings, and this paper applies the mean absolute error (MAE) to evaluate the accuracy of the predictions. The difference between the predicted score and the actual score is used to evaluate the algorithm's performance. The lower the MAE value, the better the algorithm's accuracy is indicated.

$$MAE = \frac{\sum_{i=1}^{N}|p_i - q_i|}{N} \tag{10}$$

where, N is the number of predicted ratings, $\{p_1, p_2, \cdots p_N\}$ represents the predicted ratings, $\{q_1, q_2, \cdots q_N\}$ represents the actual ratings.

Table 3. The experimental data set is shown in part.

User ID	Item ID	Rating	Timestamp
1	226	3	878543176
⋮	⋮	⋮	⋮
703	845	4	875243028
943	1067	2	875501756

3.3 Results and Discussion

The traditional Pearson correlation coefficient (PCC) and cosine similarity (COS) are used as algorithm benchmarks for comparison in this paper. In the above two similarity measures, the balance factor and time weighting function are introduced, and the algorithm is improved by dynamically altering the value of α and the number of item neighbors, and then MAE evaluates the algorithm strengths and weaknesses, as shown in Fig. 1 and Fig. 2.

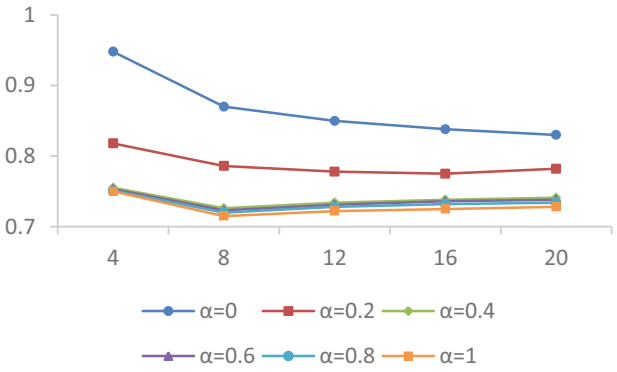

Fig. 1. The improved Pearson correlation coefficient MAE results for various values of α

As shown in Fig. 1, $\alpha = 0$ is the algorithm before improvement. When $\alpha = 0.2$, the algorithm is obviously improved, and gradually stabilizes when $\alpha = 0.4$. And with the increasing number of item neighbors, the average absolute error of the algorithm reaches the minimum when the number of neighbors is 4. At this time, the accuracy of the prediction is the highest.

The general trend of enhanced cosine similarity, as shown in Fig. 2, is similar to that of Fig. 1: it gradually tends to stabilize at $\alpha = 0.4$, and the algorithm reaches the optimum when the number of item neighbors is 4. However, the average absolute error is generally lower than the improved Pearson similarity, and the algorithm is more accurate.

In reality, the way similarity metrics are specified affects the effectiveness of collaborative filtering algorithm recommendations. For example, Pearson correlation coefficients take into consideration user rating averages, whereas cosine metric approaches

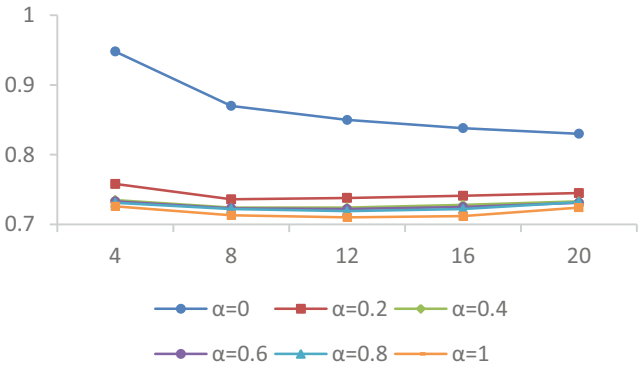

Fig. 2. The improved cosine similarity MAE results for various values of α

treat users as vectors and ignore rating averages. The performance of similarity metrics and recommendations, on the other hand, is determined by issue characteristics such as the number of users, the number of items, sparsity, and so on. The average absolute error of the algorithm is inversely proportional to the α proposed in this paper, and the larger the α, the smaller the average absolute error and the higher the accuracy of the recommendation for the user, as shown by the experimental results. As a result, the method can improve existing algorithms that have data sparsity or user interest issues.

4 Conclusion

In this paper, we propose a balance factor and time weighting function for the problems of data sparsity and user interest in the similarity calculation of traditional Pearson similarity and cosine similarity, and apply them in combination with item-based collaborative filtering. The experimental results show that the algorithm can improve the prediction accuracy of the recommendation system and provide more value to the traditional algorithm by dynamically adjusting the value of α in the balance factor. However, the algorithm uses the experimental method to arrive at the optimal value of α and the complexity of the algorithm increases after combining the balance factor with the time weighting function. Therefore, the next step will focus on how to adaptively balance the factor and reduce the complexity of the algorithm.

References

1. Wulam, A., Wang, Y., Zhang, D., et al.: A recommendation system based on fusing boosting model and DNN model. CMC-Computers Mater. Continua **60**(3), 1003–1013 (2019)
2. Subramaniyaswamy, V., Logesh, R., Chandrashekhar, M., et al.: A personalised movie recommendation system based on collaborative filtering. Int. J. High Perform. Comput. Networking **10**(1–2), 54–63 (2017)
3. Huynh, H.X., Phan, N.Q., Pham, N.M., et al.: Context-similarity collaborative filtering recommendation. IEEE. Access **8**, 33342–33351 (2020)

4. Thakkar, P., Varma, K., Ukani, V., et al.: Combining user-based and item-based collaborative filtering using machine learning. In: Information and Communication Technology for Intelligent Systems. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-1747-7_17
5. Ninan, A.M., Rajan, J.E.: An item based collaborative filtering on recommendation of travel route. Int. Res. J. Eng. Technol. **6**(5) (2019)
6. Singh, P.K., Sinha, M., Das, S., Choudhury, P.: Enhancing recommendation accuracy of item-based collaborative filtering using Bhattacharyya coefficient and most similar item. Appl. Intell. **50**(12), 4708–4731 (2020). https://doi.org/10.1007/s10489-020-01775-4
7. Hasan, M., Roy, F.: An item–item collaborative filtering recommender system using trust and genre to address the cold-start problem. Big Data Cogn. Comput. **3**(3), 39 (2019)
8. Li, D., Wang, C., Li, L., Zheng, Z.: Collaborative filtering algorithm with social information and dynamic time windows. Appl. Intell. **52**(5), 5261–5272 (2021). https://doi.org/10.1007/s10489-021-02519-8
9. Suganeshwari, G., Ibrahim, S.P.S.: Rule-based effective collaborative recommendation using unfavorable preference. IEEE Access **8**, 128116–128123 (2020)
10. Feng, J., Fengs, X., Zhang, N., et al.: An improved collaborative filtering method based on similarity. PLoS ONE **13**(9), e0204003 (2018)
11. Ajaegbu, C.: An optimized item-based collaborative filtering algorithm. J. Ambient. Intell. Humaniz. Comput. **12**(12), 10629–10636 (2021). https://doi.org/10.1007/s12652-020-02876-1
12. Kim, S.-C., Sung, K.-J., Park, C.-S., Kim, S.K.: Improvement of collaborative filtering using rating normalization. Multimed. Tools Appl. **75**(9), 4957–4968 (2013). https://doi.org/10.1007/s11042-013-1814-0
13. Jin, Q., Zhang, Y., Cai, W., et al.: A new similarity computing model of collaborative filtering. IEEE Access **8**, 17594–17604 (2020)
14. Ar, Y., Bostanci, E.: A genetic algorithm solution to the collaborative filtering problem. Expert Syst. Appl. **61**, 122–128 (2016)
15. Jiang, W., Chen, J., Jiang, Y., et al.: A new time-aware collaborative filtering intelligent recommendation system. CMC-Comput. Mater. Continua **61**(2), 849–859 (2019)
16. Kurdija, A.S., Silic, M., Vladimir, K., et al.: Efficient global correlation measures for a collaborative filtering dataset. Knowl.-Based Syst. **147**, 36–42 (2018)