Postprint of Research on Collaborative Filtering Algorithm Based on User Ratings and Co-rated Items

Authors: Zhang Hong, Wang Hui

Date: 2018-05-20T00:00:00+00:00

Abstract

To address the problems existing in collaborative filtering algorithms, improvements are proposed to enhance the accuracy of rating prediction and recommendation results. Traditional similarity metrics only consider user ratings, which is overly simplistic. Building upon Pearson similarity, we incorporate user rating time and item popularity to weight user ratings, and perform a weighted combination with similarity calculations based on the scale of co-rated items, thereby yielding more accurate and realistic results. Experimental results show that the Mean Absolute Error (MAE) of the new algorithm' s rating predictions is significantly lower than that of Pearson similarity, achieving a reduction in MAE of over 10%, while also improving recommendation recall and coverage. Although validation is limited to experiments on movie rating datasets, the algorithm demonstrates the ability to enhance collaborative filtering accuracy and holds practical significance.

Full Text

Preamble

Research on Collaborative Filtering Algorithm Based on User Ratings and Common Rating Items

Zhang Hong†, Wang Hui (School of Economics & Management, Zhejiang Sci-Tech University, Hangzhou 310018, China)

Abstract: This paper proposes improvements to the collaborative filtering algorithm to enhance the accuracy of rating prediction and recommendation results. Traditional similarity measurement methods consider only user ratings, which is overly simplistic. Building upon Pearson correlation coefficient, we introduce user rating time and item popularity to weight user ratings, and combine this

with similarity calculation based on the scale of common rating items through weighted fusion. This approach yields more accurate and realistic results. Experimental results demonstrate that the new algorithm' s Mean Absolute Error (MAE) for rating prediction is significantly lower than that of Pearson correlation coefficient, reducing MAE by over 10% while improving recommendation recall and coverage. Although experiments were conducted only on movie rating datasets, which presents certain limitations, the algorithm effectively enhances collaborative filtering accuracy and holds practical significance.

Keywords: collaborative filtering; Pearson correlation coefficient; rating time; common rating items; item popularity

0 Introduction

The development of information technology and e-commerce has greatly facilitated people' s lives, but the accompanying surge in information volume has also created information overload, making it difficult for consumers to find useful information among massive amounts of data. While search engines can filter information, they cannot provide "personalized" services. The emergence of personalized recommendation technology effectively alleviates the 困扰 caused by information overload to consumers, as its core is to recommend relevant products to consumers based on their individual preferences.

Personalized recommendation technology encompasses many algorithms, among which collaborative filtering is one of the most widely applied. First proposed by Goldberg et al. in 1992, collaborative filtering divides into two types: model-based collaborative filtering and memory-based collaborative filtering. Model-based collaborative filtering learns relevant models from historical rating data and uses these models for rating prediction. Memory-based collaborative filtering includes user-based CF and item-based CF, whose main idea is to determine neighbor users (or items) based on similarity, then use neighbors' ratings on items to predict the target user' s rating on those items, thereby generating recommendation lists. Evidently, the accuracy of user similarity measurement significantly impacts recommendation effectiveness. However, traditional similarity measurement methods such as cosine similarity, adjusted cosine similarity, and Pearson correlation coefficient consider only co-rated items between users. Under conditions of sparse user rating matrices, the calculated similarity lacks accuracy.

To optimize collaborative filtering algorithms, many experts and scholars have researched improvements to similarity calculation, primarily in three directions:

a) **Improving traditional similarity measurement methods.** Hao et al. utilized balance factors to calculate item rating differences between users and incorporated them into traditional cosine similarity algorithms, with optimal balance factors obtained through experiments. Liu et al. con-

sidered not only user rating information but also user behavior information when calculating user similarity. Ester et al. proposed a new similarity measurement method based on the sigmoid function, which can reduce the impact of items with low common ratings between users on similarity calculation.

b) **Dimensionality reduction of rating matrices.** Gong combined Singular Value Decomposition (SVD) with user-based CF, using SVD results to fill missing values before generating recommendations with user-based CF, thereby improving system accuracy. Pirasteh generated recommendations by reducing matrix dimensionality and utilized additional information such as movie genre and director data for similarity calculation. Additionally, Principal Component Analysis (PCA) is also commonly used for dimensionality reduction.

c) **Integrating other techniques with collaborative filtering algorithms.** Tsai proposed a staged scoring prediction model that first clusters users and items using clustering methods on the initial rating matrix, then applies weighted non-negative matrix factorization to early item clustering results for prediction and recommendation. Wang et al. improved data grouping algorithms using Hamming distance to achieve higher clustering accuracy, followed by Slope One method for recommendations. Furthermore, social network structures, data mining, and machine learning methods have also been applied to similarity measurement.

These methods have improved similarity calculation to varying degrees and achieved certain results. However, like traditional similarity calculation methods, they overlook the influence of user rating time, the scale of common rating items, and item popularity. This paper addresses the limitations of Pearson correlation coefficient by introducing user rating time and item popularity to weight user ratings, combined with a similarity algorithm based on the scale of common rating items to derive a new similarity measurement method called RJpearson, validated through experiments on the MovieLens dataset. Experimental results show that the proposed method effectively reduces rating prediction errors and improves similarity measurement accuracy.

---

1 Traditional Collaborative Filtering Algorithm and Its Problems

1.1 Traditional Collaborative Filtering Algorithm

Collaborative filtering algorithms divide into user-based collaborative filtering and item-based collaborative filtering, with similar main steps: similarity measurement based on historical ratings, neighbor list generation, rating prediction, and recommendation list generation based on predicted ratings. This paper focuses on user-based collaborative filtering, with the specific algorithm flow

shown in [Figure 1: see original paper].

Similarity measurement is the core of collaborative filtering algorithms. Commonly used methods include cosine similarity, adjusted cosine similarity, and Pearson correlation coefficient. Pearson correlation coefficient calculates the linear correlation between two variables using the following formula:

$$Pearson(i,j) = \frac{\sum_{k \in I_{ij}} (r_{ik} - \bar{r}_i)(r_{jk} - \bar{r}_j)}{\sqrt{\sum_{k \in I_{ij}} (r_{ik} - \bar{r}_i)^2} \sqrt{\sum_{k \in I_{ij}} (r_{jk} - \bar{r}_j)^2}}$$

where $Pearson(i,j)$ represents the similarity between users i and j, I_{ij} represents the set of items co-rated by users i and j, r_{ik} and r_{jk} represent user i' s and user j' s ratings on item k respectively, and $\bar{r}_i$ and $\bar{r}_j$ represent the average ratings of user i and user j respectively.

The rating prediction formula is defined as follows:

$$\hat{r}_{ik} = \bar{r}_i + \frac{\sum_{j \in N_i} Sim(i,j) \times (r_{jk} - \bar{r}_j)}{\sum_{j \in N_i} |Sim(i,j)|}$$

where $\bar{r}_i$ represents the average rating of target user i on all rated items, N_i represents the neighbor set of target user i, r_{jk} represents neighbor user j' s rating on item k, $\bar{r}_j$ represents neighbor user j' s average rating on rated items, and $Sim(i,j)$ represents the similarity between target user i and neighbor user j.

1.2 Problems in Similarity Measurement

1.2.1 Ignoring the Scale of Common Rating Items E-commerce platforms contain massive numbers of users and items, but each user purchases or rates only a limited number of items, resulting in extremely sparse user rating matrices. The scale of common rating items affects user similarity calculation. For example, consider the ratings of users user1 and user2 on purchased items as shown in .

Using Pearson similarity measurement, the similarity between U1 and U2 is calculated as 1. However, the rating matrix clearly shows that U1 prefers I1, I4, and I5, while U2 prefers I3 and I6—their preference behaviors are not similar. Traditional Pearson similarity measurement ignores the scale of common rating items, leading to inaccurate similarity calculations.

1.2.2 Ignoring the Influence of User Rating Time Time stamps are a crucial factor in recommendation systems as they reflect user interest migration. Users' interests and hobbies change significantly over different time periods. For instance, users show higher interest in cooling products during summer and

warming products during winter. However, traditional similarity measurement methods ignore this important factor and cannot reflect changes in user interests in similarity measurement, resulting in recommendations that may not align with users' current preferences.

1.2.3 Ignoring the Influence of Item Popularity Users often exhibit herd mentality when purchasing products, defined as the social psychological behavior of abandoning one' s own opinions to align with the majority under group influence. When shopping online, users habitually check product sales volume—higher sales increase users' tendency to purchase and give high ratings. Therefore, item popularity influences consumer purchasing behavior. Pearson similarity calculation ignores the impact of item popularity on similarity measurement, leading to insufficient similarity measurement accuracy and causing recommendation results to mostly consist of popular items while neglecting other items. True "personalized"recommendations should not only recommend popular items but also suggest items that reflect users' preference behaviors.

To address these problems and improve the accuracy of user similarity measurement and recommendation effectiveness, this paper proposes a new user similarity measurement method based on Pearson correlation coefficient, detailed in the following section.

---

2 Improved Algorithm Based on User Ratings and Common Rating Items

2.1 Similarity Calculation Based on User Ratings

To reflect users' current preferences and exclude the influence of item popularity on user ratings, we first incorporate a penalty factor and logistic time weight function to weight user ratings before using Formula (1) for similarity calculation.

2.1.1 Penalty Factor To address Pearson similarity' s limitation of ignoring item popularity, we introduce a penalty factor to weight user ratings. Item popularity is measured by N_k, representing the number of users who rated item k—a larger N_k indicates a more popular item. To reduce the impact of item popularity on user similarity, the penalty factor for item k is defined as:

$$penalty(k) = \frac{1}{\log(1 + N_k)}$$

The more ratings item k receives (i.e., the more popular the item), the greater the penalty, and the smaller its influence when calculating user similarity. Therefore, introducing the penalty factor can reduce errors caused by item popularity in user similarity calculation.

2.1.2 Logistic Time Weight Function To address the limitation that traditional algorithms cannot reflect changes in user interests over time, we incorporate a logistic time weight function when calculating user similarity. The logistic function weights user ratings temporally to increase the weight of recent ratings:

$$f(t_{ik}) = \frac{1}{1 + e^{-t_{ik}}}$$

where t_{ik} represents the time difference between when user i rated item k and the user's earliest rating time. The function value $f(t_{ik})$ grows nonlinearly with t_{ik}— the more recent the rating, the greater its impact on similarity calculation. This function' s value range is (0,1). Temporal weighting of user ratings addresses the limitation of traditional algorithms that rely solely on ratings, better selecting neighbor users with higher similarity for target users.

2.1.3 Similarity Calculation Based on User Ratings Based on item popularity and logistic time weight function, user ratings are weighted as:

$$R_{ik} = r_{ik} \times penalty(k) \times f(t_{ik})$$

where r_{ik} represents user i' s rating on item k, $penalty(k)$ represents the penalty factor for item k, and $f(t_{ik})$ represents the logistic time weight for user i' s rating on item k. After weighted processing, a new rating matrix is obtained, and similarity calculation based on user ratings is performed on this new matrix:

$$Rpearson(i,j) = \frac{\sum_{k \in I_{ij}} (R_{ik} - \bar{R}_i)(R_{jk} - \bar{R}_j)}{\sqrt{\sum_{k \in I_{ij}} (R_{ik} - \bar{R}_i)^2} \sqrt{\sum_{k \in I_{ij}} (R_{jk} - \bar{R}_j)^2}}$$

where $Rpearson(i,j)$ represents the similarity between users i and j, R_{ik} and R_{jk} represent the weighted ratings of users i and j on item k, and $\bar{R}_i$ and $\bar{R}_j$ represent the average weighted ratings of users i and j respectively.

2.2 Similarity Calculation Based on Common Rating Items

The scale of common rating items is generally measured using Jaccard similarity:

$$Jaccard(i,j) = \frac{|I_i \cap I_j|}{|I_i \cup I_j|}$$

where I_i and I_j are the rating sets of users i and j respectively. Jaccard similarity measures the similarity between two users using the ratio of the intersection (i.e., number of common rating items) to the union of the two users'rating sets. In two users' rating sets, more common rating items indicate higher similarity between

the two users, meaning their interests are more similar, and vice versa. Jaccard similarity effectively reflects user similarity based on common rating items.

2.3 Algorithm Improvement Based on User Ratings and Common Rating Items

We fuse similarity based on user ratings ($Rpearson$) and similarity based on common rating items ($Jaccard$) to obtain $RJpearson$ similarity:

$$RJpearson(i,j) = \lambda \times Rpearson(i,j) + (1-\lambda) \times Jaccard(i,j)$$

where λ is a balance parameter with a value range of (0,1). Different λ values yield different similarities. The improved algorithm flow is shown in [Figure 2: see original paper].

Therefore, when conducting experiments on different datasets, we use different $RJpearson$ calculation formulas based on the computed λ values.

---

3 Experiments and Results

3.1 Experimental Dataset

This experiment uses the MovieLens-100k dataset provided by GroupLens. The dataset employs a 5-point rating scale and contains 943 users and 1,682 movies, with each user rating at least 20 movies. We randomly selected 80% of the sample data as the training set and the remaining 20% as the test set. Similarity calculations were performed on the training set, and ratings for movies in the test set were predicted and compared with actual ratings to obtain experimental results.

The comparison algorithms in this experiment are: traditional user-based CF algorithm and recommendation algorithm based on Jaccard similarity.

3.2 Evaluation Metrics

Various criteria exist for measuring recommendation system quality. This paper selects the commonly used Mean Absolute Error (MAE) and precision-recall methods.

MAE is the most frequently used evaluation standard for comparing errors between actual and predicted values, calculated as:

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

where p_i and q_i represent the predicted rating and actual rating for a user respectively. The formula shows that closer predicted ratings to actual ratings yield smaller absolute errors and smaller MAE values, indicating better prediction performance.

Precision and recall generally measure the effectiveness of recommendation lists returned by algorithms:

$$Precision = \frac{n}{R_n}$$

$$Recall = \frac{n}{N}$$

where n represents the number of correctly recommended items (i.e., system-recommended items that appear in users' actual purchases), N represents the number of items users truly need (i.e., actual purchases), and R_n represents the number of items recommended by the system.

Precision and recall are often contradictory—when precision increases, recall tends to decrease, and vice versa. Therefore, to comprehensively consider both metrics, this paper adopts $Fusion$ as the recommendation effectiveness measure, calculated as:

$$Fusion = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

3.3 Experimental Results

3.3.1 Selecting Optimal λ Value We tested λ values of 0, 0.1, 0.2, 0.3, ···, 1, calculating corresponding MAE values for each. On the MovieLens-100k dataset, MAE reached its minimum value of 0.70 when $\lambda = 0.8$. Therefore, the $RJpearson$ calculation formula is:

$$RJpearson(i,j) = 0.8 \times Rpearson(i,j) + 0.2 \times Jaccard(i,j)$$

On the ml-latest-small dataset, the optimal λ value was 0.6.

3.3.2 Impact of Neighbor Quantity on Prediction Accuracy To examine how neighbor quantity affects rating prediction accuracy, we selected two different rating datasets: MovieLens-100k and ml-latest-small (generated on October 17, 2016, containing 100,004 ratings from 671 users on 9,125 movies).

1) MovieLens-100k Dataset

Neighbor quantity impacts prediction accuracy. We selected neighbor quantities of 5, 10, 20, 30, 50, and 100, calculating MAE values under three algorithms. Results are shown in , with comparison chart in [Figure 4: see original paper].

Table 2: MAE Values for Three Algorithms

Neighbors	Pearson	Jaccard	RJpearson
5	0.751	0.793	0.698
10	0.754	0.796	0.701
20	0.757	0.801	0.704
30	0.759	0.804	0.706
50	0.761	0.807	0.708
100	0.763	0.809	0.710

Figure 4: MAE Values for Different Algorithms

Experimental results show that *RJpearson* similarity yields significantly lower MAE values than Pearson and Jaccard similarities, with improvement percentages shown in .

Table 3: MAE Improvement Percentage of RJpearson Similarity

Neighbors	vs Pearson	vs Jaccard
5	6.94%	12.55%
10	10.59%	11.51%
20	12.32%	10.71%
30	12.58%	10.24%
50	12.87%	9.71%
100	12.04%	9.16%

The *RJpearson* similarity clearly reduces user-based CF rating prediction errors and improves prediction accuracy.

2) ml-latest-small Dataset

Similarly, with neighbor quantities of 5, 10, 20, 30, 50, and 100, we calculated MAE values under three algorithms. The comparison chart is shown in [Figure 5: see original paper].

Table 4: MAE Values for Three Algorithms

Neighbors	Pearson	Jaccard	RJpearson
5	0.698	0.734	0.623
10	0.701	0.738	0.627
20	0.704	0.741	0.631
30	0.706	0.744	0.634
50	0.708	0.746	0.637
100	0.710	0.748	0.640

Both experimental results show that *RJpearson* similarity yields significantly lower MAE values than Pearson and Jaccard similarities, with improvements exceeding 10%. On the ml-latest-small dataset, the algorithm demonstrates significant MAE reduction and improved prediction accuracy.

3.3.3 Impact of Recommendation Quantity on Recommendation Results Based on Section 3.3.2 results, both Pearson and *RJpearson* achieve optimal performance with lowest MAE at 5 neighbors. Therefore, with neighbor quantity fixed at 5, we selected recommendation quantities of 5, 10, 15, 20, and 25 to analyze their impact on recommendation results. We generated recommendation lists and calculated *Fusion* values, with comparison chart shown in [Figure 6: see original paper].

Figure 6: Fusion Values for Different Algorithms

As shown in [Figure 6: see original paper], *Fusion* values decrease as recommendation quantity increases, indicating that recommendation accuracy declines with more recommendations. Therefore, selecting an appropriate recommendation quantity is crucial. Compared with Pearson similarity, *RJpearson* similarity yields higher *Fusion* values, demonstrating improved recommendation accuracy. At a recommendation quantity of 5, *Fusion* value improves by 0.5%. Although *RJpearson* similarity' s improvement in *Fusion* is modest, the new algorithm provides some enhancement to user-based CF.

---

4 Conclusion

This paper addresses the limitations of traditional collaborative filtering algorithms by introducing penalty factors and logistic time functions to weight user ratings based on Pearson similarity, then combining with Jaccard similarity to obtain *RJpearson* similarity. Experimental validation on MovieLens datasets demonstrates that *RJpearson* similarity achieves higher accuracy and reduces rating prediction errors, thereby improving personalized recommendation service quality. However, this paper also has limitations: although rating prediction errors are significantly reduced, precision and recall of recommendation results do not improve substantially, possibly due to data sparsity. Future work will further optimize the algorithm, conduct experimental validation on multiple datasets, and explore combining text mining or complex network technologies for personalized recommendation research.

References

[1] Symeonidis P, Nanopoulos A, Papadopouls A, et al. Collaborative filtering based on user trends [J]. Advances in Data Analysis, 2006, 4425: 375-382.

[2] Adomavicius G, Tuzhilin A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions [J]. IEEE Trans on Knowledge and Data Engineering, 2005, 17(6): 734-749.

[3] Zhang Kai, Feng Zhiyong, Chen Shichan, et al. A framework for passengers demand prediction and recommendation [C]// Proc of IEEE International Conference on Services Computing. 2016: 340-347.

[4] Resnick P, Iacovou N, Suchak M, et al. GroupLens: an open architecture for collaborative filtering of net news [C]// Proc of ACM Conference on Computer Supported Cooperative Work. 1994: 175-186.

[5] Chen Hao, Li Zhongkun, Hu Wei. An improved collaborative recommendation algorithm based on optimized user similarity [J]. Journal of Supercomputing, 2016, 72(7): 2565-2578.

[6] Liu Haifeng, Hu Zheng, Mian A, et al. A new user similarity model to improve the accuracy of collaborative filtering [J]. Knowledge-Based Systems, 2014(56): 156-166.

[7] Jamali M, Ester M. A random walk model for combining trust based and item-based recommendation [C]// Proc of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2009.

[8] Ye HongWu, Dai Yae, Gong SongJie. Combining singular value decomposition and item-based recommender in collaborative filtering [C]// Proc of International Workshop on Knowledge Discovery and Data Mining. 2009: 769-772.

[9] Sarwar M B, Karypis G, A. Konstan J, et al. Application of dimensionality reduction in recommender system-a case study [C]// Proc of ACM WebKDD Workshop, 2000.

[10] Pirasteh P, Jung J J, Hwang D. Item-based collaborative filtering with attribute correlation: a case study on movie recommendation. Intelligent information and database systems [M]. Berlin: Springer International Publishing, 2014: 245-252.

[11] Tsai C, Hung C. Cluster ensembles in collaborative filtering recommendation [J]. Applied Soft Computing, 2012, 12(4): 1417-1425.

[12] Shaohua W, Zhengde Z, Xin H. The Research on collaborative filtering recommendation algorithm based in improved clustering processing [C]// Proc of IEEE International Conference on Computer and Information Technology. 2015: 1012-1015.

[13] Lu L, Medo M, Chi H Y, et al. Recommender systems [J]. Physics Reports, 2012, 519(1): 1-49.

[14] Menaouer B, Atmani B, Matta N. Dynamic knowledge mapping guided by data mining: application on healthcare [J]. Journal of Information Processing Systems, 2013, 9(1): 1-30.

[15] Motavaselalhagh F, Safi Esfahani F, Arabnia R. Knowledge-based adaptable scheduler for SaaS providers in cloud computing [J]. Human-centric Computing and Information Sciences, 2015, 5(1): 1-19.

[16] Ren Kankan, Qian XueZhong. Research on user similarity measure method in collaborative filtering algorithm [J]. Computer Engineering, 2015, 41(8).

[17] Su H, Lin X, Yan B, et al. The collaborative filtering algorithm with time weight based on MapReduce [C]// Proc of International Conference on Big Data Computing and Communications. 2015: 386-395.

[18] Yang L, Hu Y. An improved collaborative filtering algorithm based on the constraint model of confidence [J]. Journal of Computational Information Systems, 2015, 11(8): 3001-3009.

Note: Figure translations are in progress. See original paper for figures.