APMDI-CF: An Effective and Efficient Recommendation Algorithm for Online Users

Ya-Jun Leng*, Zhi Wang, Dan Peng, and Huan Zhang
College of Economics and Management, Shanghai University of Electric Power,
Shanghai 201306, China
[e-mail: huayi2001@163.com, wangzhi991379@163.com, pd270leo@163.com, zh17795706435@163.com]
*Corresponding author: Ya-Jun Leng

Received July 13, 2023; revised August 20, 2023; accepted November 14, 2023; published November 30, 2023

Abstract

Recommendation systems provide personalized products or services to online users by mining their past preferences. Collaborative filtering is a popular recommendation technique because it is easy to implement. However, with the rapid growth of the number of users in recommendation systems, collaborative filtering suffers from serious scalability and sparsity problems. To address these problems, a novel collaborative filtering recommendation algorithm is proposed. The proposed algorithm partitions the users using affinity propagation clustering, and searches for *k* nearest neighbors in the partition where active user belongs, which can reduce the range of searching and improve real-time performance. When predicting the ratings of active user's unrated items, mean deviation method is used to impute values for neighbors' missing ratings, thus the sparsity can be decreased and the recommendation quality can be ensured. Experiments based on two different datasets show that the proposed algorithm is excellent both in terms of real-time performance and recommendation quality.

Keywords: Recommendation system, Collaborative filtering, Affinity propagation clustering, Mean deviation method

1. Introduction

Nowadays, the problem of information overload is becoming more and more serious [1]. Recommendation systems [2] are effective ways to solve the problem of information overload. Many websites have applied recommendation systems to provide personalized products or services to online users, such as YouTube, Netflix, Amazon and Facebook, etc. Recommendation systems are usually divided into three categories [3]: content-based recommendations, collaborative recommendations and hybrid approaches. Collaborative filtering does not need to consider the content of the items being recommended and is easy to implement, which makes it a successful recommendation technique [4], [5]. Collaborative filtering finds neighbors with similar interests and preferences to the active user, and then recommends items (e.g., web pages, music, movies, books, products, etc.) which neighbors like to the active user [6]. Memory-based and model-based methods are two common methods in collaborative filtering. Memory-based method first calculates similarities between users, and generates recommendations based on the similarities and ratings of users. Model-based method adopts machine learning technique to construct a model from users' rating data, and uses the model to generate recommendations to the active user [7]. However, the number of users and items on websites is constantly increasing, and the sparsity and scalability problems become more and more serious, which directly affect the accuracy and real-time performance of collaborative filtering [1], [8].

In this paper, a novel collaborative filtering algorithm based on affinity propagation clustering and mean deviation is proposed. By affinity propagation clustering algorithm, users with similar interests are grouped into the same cluster. Similarities between the active user and cluster centers are calculated, and the cluster with the highest similarity is selected to search nearest neighbors for the active user, which can reduce the neighbor search scope and improve real-time performance of the collaborative filtering algorithm. Nearest neighbors are selected, and a dense rating matrix after offline filling is used to calculate the predictions of unrated items for the active user, thus the sparsity of the nearest neighbor ratings is reduced and the recommendation quality is improved. Experiments are conducted on the MovieLens100K and Jester datasets, the MAE, F1 and run-time of the collaborative filtering algorithms are compared. The experimental results show that the proposed algorithm is superior to existing ones. This study has the following contributions:

(1) Affinity propagation algorithm is adopted to cluster users. The nearest neighbors are searched in the active user's cluster, which reduces the neighbor search scope and improves the speed of the recommendation algorithm.

(2) The mean deviation method is proposed to alleviate the sparsity of user ratings, and the accuracy of the recommendation is improved.

Our article is organized as follows: In Section 2, related work is introduced. Section 3 describes the proposed collaborative filtering algorithm based on affinity propagation clustering and mean deviation. Section 4 gives the experimental evaluation, and the study is summarized in Section 5.

2. Related work

(1) Studies on the scalability problem. Mu et al. [1] used the preferences of virtual core users to make recommendations for target users, which can reduce the set of candidate neighbors

and alleviate the scalability problem. Chang et al. [9] used the convolutional autoencoder to reduce the computations when the amount of data is huge, thus the real-time performance of their recommendation method is excellent. Wu et al. [10] developed a recommendation system based on bi-clustering and moth flame optimization. Their recommendation system is highly scalable. Chen et al. [11] proposed a matrix factorization model with explicit feedback and implicit feedback. Two feedback matrices are decomposed into a shared subspace simultaneously, which addresses the scalability problem of the recommendation algorithm. **(2) Studies on the sparsity problem.** Feng et al. [12] proposed a multi-factor similarity measure, which can capture nonlinear and linear correlations between users, thus alleviating the sparsity problem in collaborative filtering. Belkhadir et al. [13] proposed an intelligent recommendation method based on social regularization and trust information, which has high recommendation quality. Moon et al. [14] proposed a collaborative filtering method for implicit datasets. The method can address the popularity bias and data sparsity problems. Duan et al. [15] proposed a method which combines matrix factorization and review-based collaborative filtering, thus the sparsity problem is addressed. **(3) Affinity propagation algorithm.** Leng et al. [16] applied affinity propagation algorithm to cluster black-start schemes, and designed a novel black-start evaluation method to restore the power system after a large-area blackout. In the study of Wang et al. [17], according to the climate characteristics of photovoltaic power stations, the photovoltaic output data are marked, and distributed photovoltaic power stations are clustered by affinity propagation algorithm, so that the meteorological data are consistent across all clusters. Wei et al. [18] proposed a novel intelligent fault diagnosis method for roller bearings based on affinity propagation algorithm and adaptive feature selection technique to better equip with a non-expert to carry out diagnosis operations. Geng et al. [19] used affinity propagation algorithm to improve data envelopment analysis model in the division of the efficiency value of decision making units. Through the affinity propagation algorithm, high influence input data of the energy efficiency can be obtained.

Most of the previous studies on collaborative filtering only solved one of the problems of sparsity and scalability. Our study focuses on both of the above problems. The use of clustering for user partitioning narrows down the search space and enhances efficiency. Additionally, incorporating the mean deviation method to impute missing ratings contributes to reducing sparsity and maintaining recommendation quality.

3. The proposed collaborative filtering algorithm

3.1 Clustering users by affinity propagation algorithm

Let $U=\{u_1, u_2, \ldots, u_m\}$ be the user space, $I=\{i_1, i_2, \ldots, i_n\}$ be the item space. The user-item rating matrix A can be expressed as Equation (1), where a_{ij} in A denotes the rating user u_i gives to item i_j. Because the rating matrix is very sparse, many ratings in A are absent, i.e. $a_{ij}=\bullet$.

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \tag{1}$$

Affinity propagation clustering (APC) [20] is used to cluster users in A. For any two users p and q, the similarity between them is computed by Euclidean distance: $s(p, q)=-\|p-q\|^2$. Then the similarity matrix $S=[s_{ij}]_{m\times m}$ is constructed, where s_{ij} is the similarity between the ith user

and the *j*th user. APC propagates responsibility $r(p, q)$ and availability $a(p, q)$ between users p and q. According to Equations (2) to (4) [20],[21], $r(p, q)$ and $a(p, q)$ are updated.

$$r^{(c)}(p,q)=(1-\lambda)\times\left(s(p,q)-\max_{q^{'}s.t.q^{'}\neq q}\{a^{(c-1)}(p,q^{'})+s(p,q^{'})\}\right)+\lambda\times r^{(c-1)}(p,q) \tag{2}$$

$$\text{If } p\neq q, a^{(c)}(p,q)=(1-\lambda)\times\left(\min\{0,r^{(c)}(q,q)+\sum_{p^{'}s.t.p^{'}\notin\{p,q\}}\max\{0,r^{(c)}(p^{'},q)\}\}\right)+\lambda\times a^{(c-1)}(p,q) \tag{3}$$

$$a^{(c)}(q,q)=(1-\lambda)\times\left(\sum_{p^{'}s.t.p^{'}\neq q}\max\{0,r^{(c)}(p^{'},q)\}\right)+\lambda\times a^{(c-1)}(q,q) \tag{4}$$

where c is the current number of iterations, and λ is the adjustment coefficient, which can eliminate numerical oscillations.

When APC terminates, the exemplar (user q) can be found for user p. User q satisfies the formula $q=\arg\max_{h}\{a(p,h)+r(p,h)\}$, where h is any user in user space U. The exemplars of all users are found by APC, and users in matrix A are clustered into t clusters, and the set of user clusters $C=\{c_1, c_2, \ldots, c_t\}$ is obtained, where $c_1\cup c_2\cup\cdots\cup c_t=U$, $c_i\cap c_j=\varnothing$ ($1\leq i\leq t$, $1\leq j\leq t$). And the set of cluster centers $R=\{r_1, r_2,\ldots, r_t\}$ is determined.

3.2 Imputing ratings by mean deviation method

Since too sparse data will affect the accuracy of recommendation [22],[23], a new data imputation method called mean deviation method is proposed in this study. When the nearest neighbors are used to generate recommendations to the active user, the mean deviation method is used to impute the absent ratings of the nearest neighbors, thus the sparsity issue can be alleviated.

Definition 1. Rating frequency weight. Let $I_u=\{i\in I|a_{u,i}\neq\bullet\}$ be the set of items that user u has rated. Take any item i in item set I, $U_i=\{u\in U\mid a_{u,i}\neq\bullet\}$ denotes the set of users who have rated i, then the rating frequency weight of any user u_j ($u_j\in U_i$) on item i is $w_{u_j}^{i}=\left|I_{u_j}\right|\Big/\sum_{u\in U_i}|I_u|$.

Rating frequency weight reflects a member's role in the group with the number of his/her ratings. The larger the proportion of the number of a member's ratings to the total number of ratings, the more representative his/her ratings of items are. That's because the more ratings a member has, the more items he/she has exposed to, thus he/she has a deeper sense of items. On the other hand, members with more ratings tend to interact with the system more often, and they contribute more to the development of the system, thus the system should give more consideration to their opinions as encouragement.

Take any item i in the item space I, then $R_i^s=\sum_{u\in U_i}w_u^i\times a_{u,i}$ is called the standard rating value of item i, and $R^s=(R_{i_1}^s, R_{i_2}^s,\cdots, R_{i_n}^s)$ is the standard rating vector of item space I. Take any user u_j in user space U, the mean deviation between the rating vector of u_j and the standard rating vector is $d_{u_j}=\frac{1}{\left|I_{u_j}\right|}\sum_{i\in I_{u_j}}\left|a_{u_j,i}-R_i^s\right|$. Let $I^{above}=\{i\in I_{u_j}\mid a_{u_j,i}-R_i^s>0\}$ be the set of items for which the ratings of u_j are higher than the standard ratings, $I^{below}=\{i\in I_{u_j}\mid a_{u_j,i}-R_i^s<0\}$ be the set of items for which the ratings of u_j are lower than the standard ratings. Then the rating of user u_j to unrated item i_p can be predicted as:

$$f_{u_j,i_p}=\begin{cases} R^s_{i_p}+d_{u_j}, & \left|I^{above}\right|\geq\left|I^{below}\right|, \\ R^s_{i_p}-d_{u_j}, & \left|I^{above}\right|<\left|I^{below}\right|. \end{cases} \tag{5}$$

where f_{u_j,i_p} is the predicted rating for user u_j on item i_p.

Repeat Equation (5) to impute the ratings of unrated items of all users, and a dense rating matrix B is obtained:

$$B=\begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix} \tag{6}$$

where $b_{ij}=a_{ij}$ if $a_{ij}\neq\bullet$ and f_{ij} otherwise.

3.3 Recommendation generation

According to the cosine similarity equation [24],[25], the similarity of the active user u to each cluster center in $R=\{r_1, r_2,\ldots, r_t\}$ is calculated, and the most similar cluster c_{max} is selected as the cluster to which u belongs.

$$sim(u,r)=\frac{\sum_{i\in I_{ur}} a_{u,i}\times a_{r,i}}{\sqrt{\sum_{i\in I_u} a_{u,i}^2}\sqrt{\sum_{i\in I_r} a_{r,i}^2}} \tag{7}$$

where $sim(u, r)$ denotes the similarity between active user u and cluster center r, $a_{u,i}$ is the rating of user u on item i in matrix A.

Based on the matrix A, the similarity between u and each user in c_{max} is calculated using Equation (7), and k most similar users are selected as user u's nearest neighbors $U_n=\{u_1, u_2,\ldots, u_k\}$.

For any unrated item i of u (i.e. $R_{u,i}=\bullet$), based on the rating data in matrix B, Equation (8) [26], [27] is used to predict the missing rating values for u.

$$p_{u,i}=\overline{B_u}+\frac{\sum_{u_k\in U_n} sim(u,u_k)\times(b_{u_k,i}-\overline{B_{u_k}})}{\sum_{u_k\in U_n}(|sim(u,u_k)|)} \tag{8}$$

where $p_{u,i}$ denotes the prediction rating of item i for user u, $b_{u_k,i}$ denotes the rating of user u_k on item i in matrix B, $\overline{B_u}$ is the average rating of user u in matrix B.

Top-G recommendation set $I_r=\{i_1, i_2,\ldots, i_G\}$ (i.e. G items with the highest prediction ratings) is provided to the active user u.

3.4 Algorithm description

Our proposed algorithm based on affinity propagation clustering (APC) and mean deviation is divided into two stages: offline processing and online recommendation. In the offline stage, APC is firstly used to cluster users in the original rating matrix, and the cluster center of each cluster is obtained. So the neighbor search scope is reduced. Then the mean deviation method is used to impute the ratings of unrated items for users, which alleviate the sparsity of rating data. In the online stage, the similarities between the active user and all cluster centers are computed, and the most similar cluster is selected as the nearest neighbor search space. Then the nearest neighbors are determined and recommendations are made.

Algorithm 1. Collaborative filtering algorithm based on affinity propagation clustering and mean deviation (APMDI-CF).
Inputs: Rating matrix A, number of user clusters t, number of nearest neighbors k, number of recommended items G
Output: Set of recommended items I_r
Steps:

(a) According to the APC algorithm, users in matrix A are clustered into t clusters, and the set of user clusters $C=\{c_1, c_2, \ldots, c_t\}$ is obtained. And the set of cluster centers $R=\{r_1, r_2,\ldots, r_t\}$ is determined.

(b) For any user u_j, the mean deviation method is used to impute the ratings of unrated items of u_j. Repeat step (b) to impute the ratings of unrated items of all users, and a dense rating matrix B is obtained.

(c) Based on the rating data of matrix A, the similarity of the active user u to each cluster center in $R=\{r_1, r_2,\ldots, r_t\}$ is calculated by Equation (7), and the most similar cluster c_{max} is selected as the cluster to which u belongs.

(d) Based on the matrix A, the similarity between u and each user in c_{max} is calculated using Equation (7), and k most similar users are selected as user u's nearest neighbors $U_n=\{u_1, u_2,\ldots, u_k\}$.

(e) For unrated items of u, based on the rating data of matrix B, Equation (8) is used to predict the missing rating values for u.

(f) Top-G recommendation set $I_r=\{i_1, i_2,\ldots, i_G\}$ is provided to the active user u.

Steps (a)-(b) of the proposed algorithm are carried out offline, they do not affect the recommendation speed. The most time-consuming step of our algorithm APMDI-CF is to search for neighbors (i.e. Steps (c) and (d)). Step (c) calculate similarities between active user u and cluster centers, the computational complexity is $O(t\times n)$ (where t is the number of user clusters, n is the number of items). Step (d) searches for nearest neighbors in the cluster to which u belongs, the computational complexity is $O(m_a\times n)$ (where m_a denotes the total number of users in the cluster to which u belongs). So the total computational complexity of steps (c)-(d) is $O(t\times n)+O(m_a\times n)$. Since t and m_a are both much smaller than n, the total complexity of steps (c)-(d) is $O(n)$. The traditional memory-based collaborative filtering algorithms search for the nearest neighbors of u in the whole user space, the computational complexity is $O(m\times n)$ (where m is the number of users). Therefore, the computational complexity of the proposed algorithm APMDI-CF is lower than those of the memory-based collaborative filtering algorithms.

In Step (a) of the proposed algorithm APMDI-CF, the rating matrix $A_1(m, n)$ needs to be stored to cluster users by APC, so the space complexity is $O(m\times n)$. For Step (b), the rating frequency weights of m users on n items are stored, the space complexity is $O(m\times n)$. In Step (c), the similarities between the active user u to t cluster centers are calculated and stored, the space complexity is $O(t)$. In Step (d), the similarities between u and m_a users in c_{max} are calculated, and in Steps (e)-(f), G items are recommended, we need to store m_a similarities and G items. Therefore, the space complexity of Step (d) is $O(m_a)$, and the space complexity of Steps (e)-(f) is $O(G)$. The overall space complexity of the proposed algorithm APMDI-CF is $O(m\times n)+O(m\times n)+O(t)+O(m_a)+O(G)=O(m\times n)$. For the traditional memory-based collaborative filtering algorithms, an $m\times n$ rating matrix needs to be stored to calculate similarities and recommend items to the active user u, the overall space complexity is $O(m\times n)$, too.

4. Experimental evaluation

4.1 Datasets

MovieLens100K dataset [28] and Jester dataset [29] were used to evaluate the proposed algorithm. The MovieLens100K dataset contains 100,000 ratings (rating values are integers from 1 to 5) of 1682 movies by 943 users. The Jester dataset contains more than 1,700,000 ratings (rating values are real numbers from −10 to 10) of 150 jokes by 63,974 users. In this paper, two subsets were randomly selected from MovieLens100K dataset and Jester dataset respectively for experiments. **Table 1** shows the datasets used in our study. The Jester dataset were rescaled on to [1, 5]. For the MovieLens100K dataset, 600 users were selected as the training set and the other 300 users as the test set. For the Jester dataset, 500 users were selected as the training set and the other 150 users as the test set.

Table 1. MovieLens100K and Jester datasets used in our study.

	Total number of users	Total number of items	Total number of ratings	Sparsity Level[1]
MovieLens100K	900	1,675	95,303	0.9368
Jester	650	140	48,676	0.4651

[1]Sparsity level = 1 − total existing ratings / number of rows times number of columns in the rating matrix

4.2 Evaluation metrics

Mean absolute error (MAE) [30],[31], F1 [32], *coverage* [32] and run-time [33] were used as the criteria for measuring the performance of the algorithm. MAE measures the difference between true and predicted values. The smaller the MAE value, the higher the prediction accuracy of a collaborative filtering algorithm. Assume H ratings (i.e. q_1, q_2,…, q_H) are hidden in the test set. The prediction values of the hidden ratings by a collaborative filtering algorithm are p_1, p_2,…, p_H, then the MAE of the algorithm is:

$$MAE = \frac{\sum_{i=1}^{H} | p_i - q_i |}{H} \tag{9}$$

F1 is often used to evaluate the recommendation accuracy of a collaborative filtering algorithm. For all items in the active user's test set, the ratings of these items are predicted, and G items with prediction values higher than the positive rating-threshold P_r[1] are recommended to the active user. Let D denote the number of relevant recommended items (the items of the top-G list that are rated higher than the positive rating-threshold P_r by the active user). The *precision* is computed by Equation (10):

$$precision = \frac{D}{G} \tag{10}$$

The *recall* is computed by Equation (11):

[1] In our experiments, we set the positive rating-threshold P_r to 2.

$$recall = \frac{D}{M} \tag{11}$$

where M is the number of items in the test set and rated higher than P_r by the active user.

The F1 is computed by Equation (12):

$$\mathrm{F1} = \frac{2 \times precision \times recall}{precision + recall} \tag{12}$$

Coverage measures the percentage of ratings for which a collaborative filtering algorithm can provide predictions. The *coverage* is computed by Equation (13):

$$coverage = \frac{Q}{H} \tag{13}$$

where H is the number of ratings in the test set, and Q is the number of ratings that a collaborative filtering algorithm can make predictions.

The run-time is used to measure the scalability of a collaborative filtering algorithm. The shorter the run-time for the algorithm to generate recommendations, the better the scalability.

4.3 Experimental results

4.3.1 MAE of each algorithm for different values of *k*

We compared the proposed algorithm APMDI-CF with affinity propagation clustering-based algorithm (AP-CF[2]), kmeans-based algorithm (KCLUST-CF) [34] and user-based algorithm (UBCF) [15]. First, 20% of the ratings in the test set were randomly hidden, and the above 4 collaborative filtering algorithms were implemented to predict these ratings. **Fig. 1** shows the MAE of each algorithm for different number of nearest neighbors *k*. The MAE values of all collaborative filtering algorithms decrease as we increase the value of *k*. But the MAE values of all algorithms decrease less and less, and eventually all the curves flatten out. In the case of MovieLens100K dataset, the MAE of the proposed algorithm APMDI-CF is lower than those of AP-CF, KCLUST-CF and UBCF at each value of *k*. Therefore, APMDI-CF performs better than the other 3 algorithms. For the other dataset Jester, the MAE of APMDI-CF is lower than those of AP-CF and KCLUST-CF. When the value of *k* is 15, 20, 25 or 30, the MAE of APMDI-CF is higher than that of UBCF. For the MovieLens100K dataset, APMDI-CF performs better than UBCF. However, for the Jester dataset, the performance of APMDI-CF is slightly worse than that of UBCF. The reason is that Jester is a dense dataset. When predicting an unrated item *i* for the active user, most of the nearest neighbors searched by UBCF have ratings on *i*, and thus the prediction accuracy is high. The MovieLens100K dataset is very sparse. The nearest neighbors searched by UBCF have few ratings on *i*, and there are many missing values, so the prediction accuracy is poor. Though the nearest neighbors searched by APMDI-CF are limited to the cluster to which the active user belongs, the missing values of the nearest neighbor ratings are effectively imputed by the mean deviation method. This allows every nearest neighbor to participate in the rating prediction, so the prediction accuracy of APMDI-CF is higher than that of UBCF. Datasets on actual websites are often extremely sparse, so APMDI-CF has stronger applicability.

[2]AP-CF means that in the collaborative filtering algorithm, only APC is used to cluster the users, the original rating matrix is not imputed by mean deviation method, and the original rating matrix is used to predict the missing ratings.

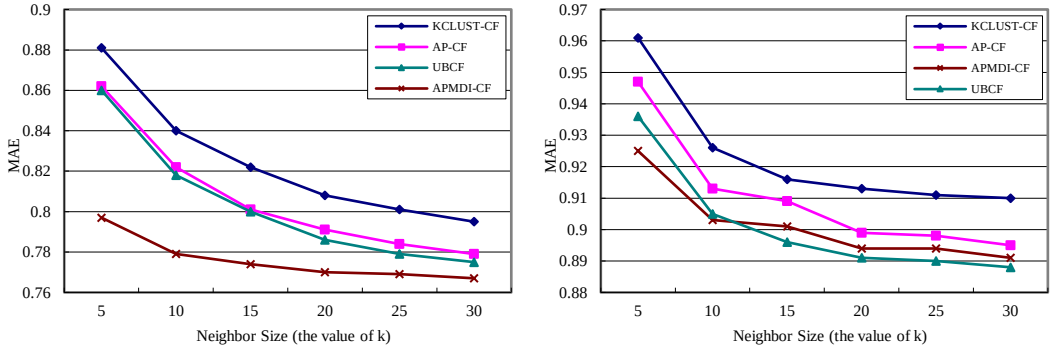

Fig. 1. Comparison of MAE on different *k*: MovieLens100K dataset (left), Jester dataset (right).

4.3.2 MAE of each algorithm for different density levels

In this section, the number of ratings of the active user is changed, and the recommendation quality of each algorithm under different density levels *s* is verified. The value of *k* (number of nearest neighbors) for each algorithm is set to 30. We retained 20%, 40%, 50%, 60% and 80% of the ratings in test set to obtain 5 different density levels. **Fig. 2** shows MAE of each collaborative filtering algorithm for different density levels *s*. The MAE values of the algorithms gradually decrease as we increase the value of *s*. The more ratings in the test set, the higher the recommendation quality of an algorithm. For MovieLens100K, the MAE of APMDI-CF is lower than those of AP-CF, KCLUST-CF and UBCF at each value of *s*. Therefore, APMDI-CF performs better than the other 3 algorithms. For Jester, the MAE of APMDI-CF is lower than those of AP-CF and KCLUST-CF. When the value of *s* is 20%, 40%, 50% or 60%, the MAE of APMDI-CF is lower than that of UBCF. When the value of *s* is 80%, the MAE of APMDI-CF is slightly higher than that of UBCF. This is consistent with the experimental results in Section 4.3.1. When the ratings are dense, most of the nearest neighbors searched by UBCF have ratings on the target item *i*, thus the prediction accuracy of UBCF is high. On the other hand, dense ratings weaken the imputation effect of the mean deviation method in APMDI-CF. As a result, UBCF performs better than APMDI-CF when *s* is 80%.

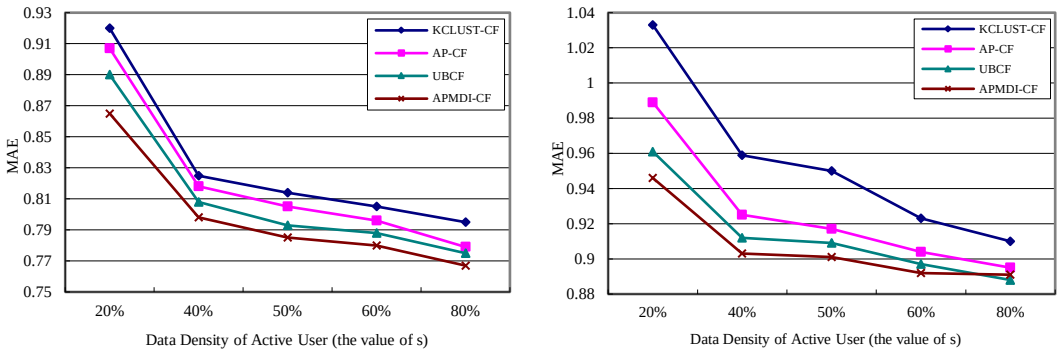

Fig. 2. Comparison of MAE on different s: MovieLens100K dataset (left), Jester dataset (right).

4.3.3. Comparison with popular algorithms

In order to compare the F1 and *coverage* of APMDI-CF with those of popular collaborative filtering algorithms, we implemented the mean squared differences and Jaccard-based algorithm (MJ-CF) [35], the kmeans-based algorithm (KCLUST-CF) [34], the potential model-based algorithm (PM-CF) [36], the user-based algorithm (UBCF) [15], the affinity propagation clustering-based algorithm (AP-CF), and the preference similarity-based algorithm (PS-CF) [37]. **Fig. 3** and **Table 2** show the F1 values. In **Fig. 3**, the neighbor size varies from 10 to 30. For both Movielens100K and Jester datasets, the F1 values of APMDI-CF are larger than those of the other six algorithms. **Table 2** gives the average F1 values of all collaborative filtering algorithms. The F1 is the average of 3 values of each algorithm that the neighbor size k varies from 10 to 30. As shown, for Movielens100K and Jester datasets, the average F1 values of APMDI-CF are 0.867 and 0.773, respectively. They are all larger than those of the other algorithms. It can be concluded that the proposed algorithm APMDI-CF has the highest recommendation accuracy.

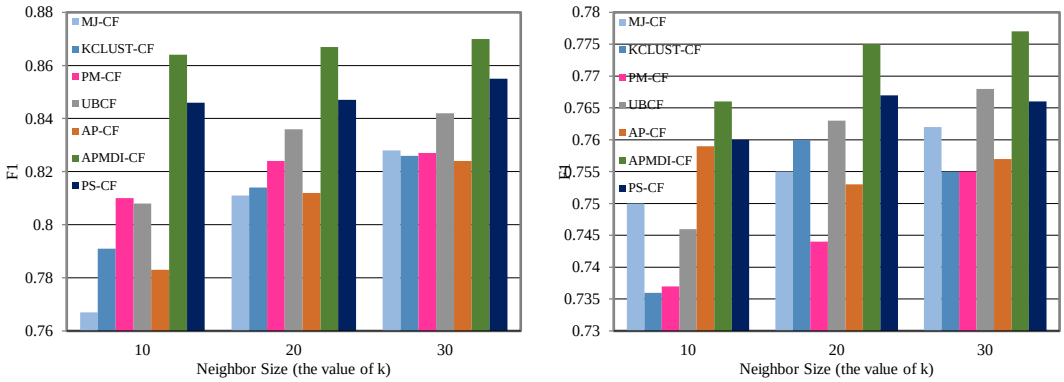

Fig. 3. Comparison of F1: MovieLens100K dataset (above), Jester dataset (below).

Table 2. Average F1 of each algorithm.

	MJ-CF	KCLUST-CF	PM-CF	UBCF	AP-CF	APMDI-CF	PS-CF
MovieLens100K	0.802	0.810	0.820	0.829	0.806	**0.867**	0.849
Jester	0.756	0.750	0.745	0.759	0.756	**0.773**	0.764

Table 3 shows the *coverage* of each algorithm. The number of nearest neighbors for each algorithm is set to 5. For Movielens100K and Jester datasets, the *coverage* values of APMDI-CF are 1.000 and 1.000, respectively. They are all larger than those of the other algorithms. Therefore, the proposed algorithm APMDI-CF performs better than the other 6 algorithms in *coverage*.

Table 3. *Coverage* of each algorithm.

	MJ-CF	KCLUST-CF	PM-CF	UBCF	AP-CF	APMDI-CF	PS-CF
Movielens100K	0.743	0.804	0.890	0.821	0.795	**1.000**	0.949
Jester	0.959	0.983	0.994	0.995	0.978	**1.000**	0.999

4.3.4 Comparison of run-time and complexity

We compared the running speed of APMDI-CF with that of UBCF. We implemented APMDI-CF and UBCF on Movielens100K dataset. For APMDI-CF, the number of user clusters t is set to 8, and the number of nearest neighbors k is set to 30. For UBCF, the value of k is also set to 30. The response times of APMDI-CF and UBCF are shown in **Fig. 4**. The response time of APMDI-CF is 1,660 milliseconds, and the response time of UBCF is 1,736 milliseconds. APMDI-CF performs better than UBCF. APMDI-CF searches for nearest neighbors in the cluster to which the active user belongs, which reduces the range of searching neighbors, and thus APMDI-CF runs faster.

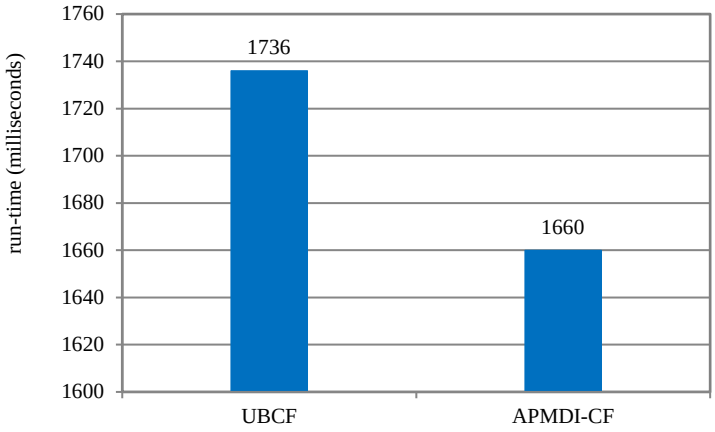

Fig. 4. Comparison of run-time.

Table 4 shows the computational complexity (CC) and space complexity (SC) of each algorithm. The computational complexity of the proposed algorithm APMDI-CF is $O(n)$, which is the lowest among all the 7 algorithms. The space complexity of each algorithm is $O(m\times n)$.

Table 4. Computational complexity and space complexity of each algorithm.

	MJ-CF	KCLUST-CF	PM-CF	UBCF	AP-CF	APMDI-CF	PS-CF
CC	$O(m\times n)$	$O(n)$	$O(m^2\times n)$	$O(m\times n)$	$O(n)$	$O(n)$	$O(m\times n)$
SC	$O(m\times n)$	$O(m\times n)$	$O(m\times n)$	$O(m\times n)$	$O(m\times n)$	$O(m\times n)$	$O(m\times n)$

5. Conclusions and future work

Nowadays, recommendation systems are used more and more widely in e-commerce, social network, video/music on demand, etc. Collaborative filtering is a successful technology for providing recommendations to online users, but it faces serious scalability and sparsity problems. In this paper, we propose a novel collaborative filtering recommendation algorithm based on affinity propagation clustering and mean deviation. Affinity propagation clustering is used to cluster users, which can reduce the range of searching neighbors and improve real-time performance of the collaborative filtering algorithm. For the missing values of nearest neighbors, the mean deviation method is proposed and used to impute these values, which alleviates the sparsity of the ratings and ensures the recommendation quality of the algorithm.

Experiments were carried out on the movie and joke datasets. The results show that our algorithm APMDI-CF has high recommendation quality, and runs faster. When using APC to cluster users, the size of the bias parameter needs to be constantly adjusted to obtain the required number of clusters. How to efficiently determine the bias parameter value will be the focus of future work.

Acknowledgement

We would like to express our acknowledgements to the providers of datasets of MovieLens and Jester. This work was supported by the Youth Foundation of Humanities and Social Sciences of Ministry of Education of China under the Grant No. 22YJCZH073.

References

[1] C. Mu, W. Chen, Y. Liu, D. Lei and R. Liu, "Virtual information core optimization for collaborative filtering recommendation based on clustering and evolutionary algorithms," *Applied Soft Computing*, vol. 116, pp. 108355, 2022. Article (CrossRef Link)

[2] A. Gazdar and L. Hidri, "A new similarity measure for collaborative filtering based recommender systems," *Knowledge-Based Systems*, vol. 188, pp. 105058, 2020. Article (CrossRef Link)

[3] R. Bagher, H. Hassanpour and H. Mashayekhi, "User trends modeling for a content-based recommender system," *Expert Systems with Applications*, vol. 87, pp. 209-219, 2017. Article (CrossRef Link)

[4] R. Kuo, C. Chen and S. Keng, "Application of hybrid metaheuristic with perturbation-based K-nearest neighbors algorithm and densest imputation to collaborative filtering in recommender systems," *Information Sciences*, vol. 575, pp. 90-115, 2021. Article (CrossRef Link)

[5] S. Wan and Z. Niu, "A hybrid E-learning recommendation approach based on learners' influence propagation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 5, pp. 827-840, 2020. Article (CrossRef Link)

[6] M. Nguyen, J. Yu, T. Nguyen and S. Yongchareon, "High-order autoencoder with data augmentation for collaborative filtering," *Knowledge-Based Systems*, vol. 240, pp. 10773, 2022. Article (CrossRef Link)

[7] B. Hammou, A. Lahcen and S. Mouline, "FRAIPA version 2: A fast recommendation approach based on self-adaptation and multi-thresholding," *Expert Systems with Applications*, vol. 118, pp. 209-216, 2019. Article (CrossRef Link)

[8] M. Najafabadi, A. Mohamed and C. Onn, "An impact of time and item influencer in collaborative filtering recommendations using graph-based model," *Information Processing & Management*, vol. 56, no. 3, pp. 526-540, 2019. Article (CrossRef Link)

[9] C. Chang and H. Chang, "Strategies of collaborative filtering recommendation mechanism using a deep learning approach," in *Proc. of IEEE 3rd Eurasia Conference on IOT, Communication and Engineering*, pp. 381-385, 2021. Article (CrossRef Link)

[10] H. Wu, G. Wu, Y. Wang and Y. Chang, "Prediction on recommender system based on bi-clustering and moth flame optimization," *Applied Soft Computing*, vol. 120, pp. 108626, 2022. Article (CrossRef Link)

[11] S. Chen and Y. Peng, "Matrix factorization for recommendation with explicit and implicit feedback," *Knowledge-Based Systems*, vol. 158, pp. 109-117, 2018. Article (CrossRef Link)

[12] C. Feng, J. Liang, P. Song and Z. Wang, "A fusion collaborative filtering method for sparse data in recommender systems," *Information Sciences*, vol. 521, pp. 365-379, 2020. Article (CrossRef Link)

[13] I. Belkhadir, E. Omar and J. Boumhidi, "An intelligent recommender system using social trust path for recommendations in web-based social networks," *Procedia computer science*, vol. 148, pp. 181-190, 2019. Article (CrossRef Link)

[14] J. Moon, Y. Jeong, D. Chae, J. Choi, H. Shim and J. Lee, "CoMix: Collaborative filtering with mixup for implicit datasets," *Information Sciences*, vol. 628, pp. 254-268, 2023. Article (CrossRef Link)
[15] R. Duan, C. Jiang, and H. Jain, "Combining review-based collaborative filtering and matrix factorization: A solution to rating's sparsity problem," *Decision Support Systems*, vol. 156, pp. 113748, 2022. Article (CrossRef Link)
[16] Y. Leng, Z. Wu, W. Zhao, Z. Li, "Assessment method of black start scheme based on affinity propagation clustering weight," *Automation of Electric Power Systems*, vol. 44, no. 13, pp. 73-80, 2020. Article (CrossRef Link)
[17] X. Wang, M. Yu, Z. Huo, D. Yang, "Short-term power prediction of distributed photovoltaic station cluster based on affinity propagation clustering and LSTNet," *Automation of Electric Power Systems*, vol. 47, no. 6, pp. 133-141, 2023. Article (CrossRef Link)
[18] Z. Wei, Y. Wang, S. He, J. Bao, "A novel intelligent method for bearing fault diagnosis based on affinity propagation clustering and adaptive feature selection," *Knowledge-Based Systems*, vol. 116, pp. 1-12, 2017. Article (CrossRef Link)
[19] Z. Geng, R. Zeng, Y. Han, Y. Zhong, H. Fu, "Energy efficiency evaluation and energy saving based on DEA integrated affinity propagation clustering: Case study of complex petrochemical industries," *Energy*, vol. 179, pp. 863-875, 2019. Article (CrossRef Link)
[20] B. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007. Article (CrossRef Link)
[21] Y. Liu, J. Liu, Y. Jin, F. Li and T. Zheng, "An affinity propagation clustering based particle swarm optimizer for dynamic optimization," *Knowledge-Based Systems*, vol. 195, pp. 105711, 2020. Article (CrossRef Link)
[22] S. Natarajan, S. Vairavasundaram, S. Natarajan and A. Gandomi, "Resolving data sparsity and cold start problem in collaborative filtering recommender system using linked open data," *Expert Systems with Applications*, vol. 149, pp. 113248, 2020. Article (CrossRef Link)
[23] N. Alharbe, M. A. Rakrouki and A. Aljohani, "A collaborative filtering recommendation algorithm based on embedding representation," *Expert Systems with Applications*, vol. 215, pp. 119380, 2023. Article (CrossRef Link)
[24] Y. J. Leng, Z. Y. Wu, Q. Lu and S. P. Zhao, "Collaborative filtering based on multiple attribute decision making," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 34, no. 3, pp. 387-397, 2022. Article (CrossRef Link)
[25] F. Fkih, "Enhancing item-based collaborative filtering by users' similarities injection and low-quality data handling," *Data & Knowledge Engineering*, vol. 144, pp. 102126, 2023. Article (CrossRef Link)
[26] C. Zhang, X. Duan, F. Liu, X. Li and S. Liu, "Three-way Naive Bayesian collaborative filtering recommendation model for smart city," *Sustainable Cities and Society*, vol. 76, pp. 103373, 2022. Article (CrossRef Link)
[27] H. N. Kim, A. El-Saddik and G. S. Jo, "Collaborative error-reflected models for cold-start recommender systems," *Decision Support Systems*, vol. 51, no. 3, pp. 519-531, 2011. Article (CrossRef Link)
[28] B. Wang, H. Xu, C. Li, Y. Li and M. Wang, "TKGAT: Graph attention network for knowledge-enhanced tag-aware recommendation system," *Knowledge-Based Systems*, vol. 257, pp. 109903, 2022. Article (CrossRef Link)
[29] C. Xu, X. Wu, M. Wang, F. Qiu, Y. Liu and J. Ren, "Improving dynamic gesture recognition in untrimmed videos by an online lightweight framework and a new gesture dataset ZJUGesture," *Neurocomputing*, vol. 523, pp. 58-68, 2023. Article (CrossRef Link)
[30] L. Frías-Paredes, F. Mallor, M. Gastón-Romeo and L. Teresa, "Dynamic mean absolute error as new measure for assessing forecasting errors," *Energy Conversion and Management*, vol. 162, pp. 176-188, 2018. Article (CrossRef Link)
[31] D. Karunasingha, "Root mean square error or mean absolute error? Use their ratio as well," *Information Sciences*, vol. 585, pp. 609-629, 2022. Article (CrossRef Link)

[32] G. Adomavicius, A. 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, pp. 734-749, 2005. Article (CrossRef Link)
[33] Y. Cho and J. Kim, "Application of web usage mining and product taxonomy to collaborative recommendations in e-commerce," *Expert System with Applications*, vol. 26, no. 2, pp. 233-246, 2004. Article (CrossRef Link)
[34] A. Ikotun, A. Ezugwu, L. Abualigah, B. Abuhaija and J. Heming, "K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data," *Information Sciences*, vol. 622, pp. 178-210, 2023. Article (CrossRef Link)
[35] J. Bobadilla, F. Serradilla, J. Bernal, "A new collaborative filtering metric that improves the behavior of recommender systems," *Knowledge-Based Systems*, vol. 23, no. 6, pp. 520-528, 2010. Article (CrossRef Link)
[36] Y. Leng, Q. Lu, C. Liang, "A collaborative filtering similarity measure based on potential field," *Kybernetes*, vol. 45, no. 3, pp. 434-445, 2016. Article (CrossRef Link)
[37] Y. J. Leng, Z. Y. Wu, Q. Lu, S. Zhao, "Collaborative filtering based on multiple attribute decision making," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 34, no. 3, pp. 387-397, 2022. Article (CrossRef Link)

Ya-Jun Leng received the B.S. degree in Business Administration from Hefei University of Technology, Hefei, China, in 2008, and received the Ph.D. degree in Management Science and Engineering from Hefei University of Technology, Hefei, China, in 2013. Currently he is an Associate Professor in the College of Economics and Management at Shanghai University of Electric Power. His research interests include recommender system and data mining.

Zhi Wang received the B.S. degree in 2020 from the College of Software Engineering, Jiangxi University of Science and Technology, Jiangxi, China. She is currently working toward the M.S. degree in the College of Economics and Management, Shanghai University of Electric Power, Shanghai, China. Her main research interest is recommender system.

Dan Peng received the B.S. degree in 2018 from the Department of Electrical Engineering, Qilu University of Technology, Jinan, China. She is currently working toward the M.S. degree at Shanghai University of Electric Power. Her main research interest is recommender system.

Huan Zhang received the B.S. degree in 2019 from the School of Management, Hefei University of Technology, Xuancheng, China. She is currently working toward the M.S. degree at Shanghai University of Electric Power. Her main research interest is data mining.