BIG DATA MINING AND ANALYTICS
ISSN 2096-0654 02/06 pp282 – 293
Volume 5, Number 4, December 2022
DOI: 10.26599/BDMA.2022.9020012

Recommendation System with Biclustering

Jianjun Sun and Yu Zhang*

Abstract: The massive growth of online commercial data has raised the request for an automatic recommender system to benefit both users and merchants. One of the most frequently used recommendation methods is collaborative filtering, but its accuracy is limited by the sparsity of the rating dataset. Most existing collaborative filtering methods consider all features when calculating user/item similarity and ignore much local information. In collaborative filtering, selecting neighbors and determining users' similarities are the most important parts. For the selection of better neighbors, this study proposes a novel biclustering method based on modified fuzzy adaptive resonance theory. To reflect the similarity between users, a new measure that considers the effect of the number of users' common items is proposed. Specifically, the proposed novel biclustering method is first adopted to obtain local similarity and local prediction. Second, item-based collaborative filtering is used to generate global predictions. Finally, the two resultant predictions are fused to obtain a final one. Experiment results demonstrate that the proposed method outperforms state-of-the-art models in terms of several aspects on three benchmark datasets.

Key words: Recommendation System (RS); Collaborative Filtering (CF); local pattern; biclustering; similarity measure

1 Introduction

Every day, more and more commercial data are produced on the web, causing difficulties in extracting valuable information. For example, the Netflix website has over 18 000 movies; thus, the preferences of moviegoers should be investigated to help merchants gain more profits. To solve such an information overload problem[1], we propose a Recommender Systems (RS) that merchants can be adopted to complement their users' experience and to drive further profits.

RS methods can mainly be summarized into three categories: content-based, Collaborative Filtering (CF) based, and hybrid. Content-based[2] models depend on user features (age, gender, and occupation) and item features (such as the movie theme). In certain situations, the content-based method is difficult to implement. CF-based methods use knowledge of similar users' preferences to predict ratings. These methods are combined in the hybrid method[3]. In a review of literature, the CF-based method is found as the most frequently used RS technology. The objective of CF is to predict the rating scores of items that users have not rated, which allows for the recommendation of items with high rating scores to users. The mechanism behind CF is that if users A and B have similar ratings (preference) on several items, the two users may also have similar ratings on other items. CF first calculates the similarities of an active user with others and then selects the neighbors with high similarities. Finally, the ratings of neighbors are predicted based on their similarities. The neighbors and similarity measures have considerable effects on the performance of CF methods.

The CF performance is highly affected by the sparsity of historical rating data. In many cases, on average, each user can rate less than 5% among thousands of items. Many values in the rating matrix are missing, and sparsity severely affects the recommendation accuracy. This problem has been addressed by using clustering[4, 5],

• Jianjun Sun is with School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China. E-mail: jjsun@mail.nwpu.edu.cn.
• Yu Zhang is with Ningbo Institute of Northwestern Polytechnical University, Ningbo 315103, China, and is also with School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China. E-mail: zhangyu@nwpu.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2021-09-06; revised: 2022-03-02; accepted: 2022-05-05

mainly by one-way methods. Between active and historical users, calculating the similarity measure considers all the items, and ignores local information. However, missing local information may reduce the quality of neighbors[4]. In fact, like-minded users may differ globally but are alike locally. Local pattern-based (local similarity) recommendation methods can take advantage of the local information ignored by traditional one-dimension clustering[6–9]. Considering that two users have similar preferences on several items but may have different preferences on others, Bu et al.[6] proposed a multiclass co-clustering algorithm to capture subgroups and combined them with the traditional CF algorithm to improve prediction performance. With the assumption that the global user-item matrix can be represented as the weighted sum of local small low-rank matrices, novel local low-rank matrix approximation-based recommendation systems have been proposed[7, 8]. Bansal and Baliyan[9] proposed a novel memory-based evolutionary BIclustering-based Memetic Algorithm for Recommender System (BIMARS). First, BIMARS removes the ratings with values less than or equal to 2. Then, similar users are found by searching biclusters. Subsequently, similarity vectors are randomly initialized, and the memetic algorithm is used to optimize the similarity vector. Finally, the ratings are predicted by combining the average of the neighborhood. BIMARS removes the ratings with scores less than or equal to 2, thus reducing the number of ratings and increasing the sparsity of the datasets.

Figure 1 shows the difference between one-way clustering and biclustering. The latter can cluster the matrix from both the row and column dimensions simultaneously and thus find many local coherent patterns. The rows/columns in the bicluster have strong correlations. As a combination of row and column dimension clustering, biclustering has been successfully applied to gene expression data analysis[10], tumor classification[11], stock price prediction[12, 13], and community detection[14]. In this study, a novel biclustering based on two fuzzy ART modules is proposed, with one module for item clustering and the other module for user clustering. Item clustering and user clustering are connected via the bicluster test. Item clustering is done with standard fuzzy ART[15], which can quickly learn about novel input patterns without forgetting previous knowledge. The winning neuron is determined only by the vigilance test. For user clustering, the winning neuron is determined by using two steps. First, the input pattern must pass the vigilance test, and the second step is to pass the bicluster test. A neuron that passes the vigilance test is deemed as the potential winning neuron; a potential winning neuron that passes the bicluster test is set as the winning neuron.

CF can be divided into two steps; the first step is finding neighbors, and the second step is predicting the rating with the neighbors and similarity measure. With the proposed fuzzy ART-based biclustering, excellent neighbors can be found. In addition, an excellent similarity measure is needed to obtain better predictive values. Generally speaking, the more items co-rated by two users, the higher their similarity. However, in literature, the similarity measure is usually the cosine coefficient or Pearson correlation coefficient, which does not consider the number of co-rated items. Therefore, in this study, to better reflect the similarity between two users, we proposed a novel similarity measure, S_l, that considers the number of users' co-rated items.

The workflow of Biclustering-Based CF (BBCF) can be summarized as follows: (1) mine biclusters with the proposed biclustering method; (2) select the users in the bicluster as neighbors, and (3) obtain local prediction with the neighbors and the proposed similarity measure S_l. To achieve better prediction results, in this study, we combine the biclustering-based local prediction and Item-Based CF (IBCF) global prediction with linear regression[16], and name it CBI. In this fusion method, BBCF adopts user-based local similarity while IBCF[17]

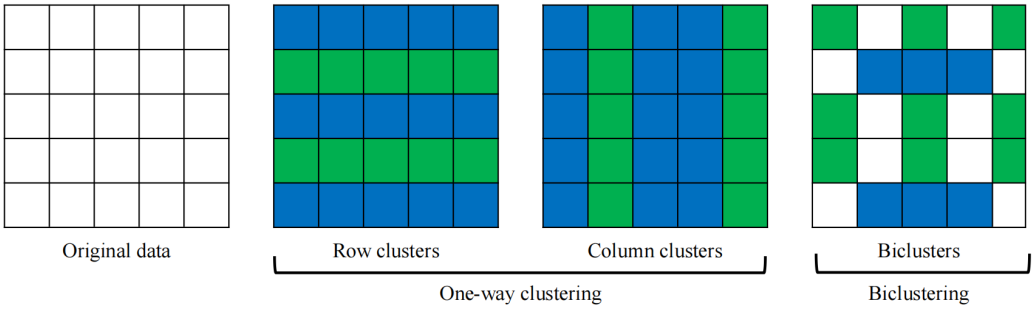


Fig. 1 Example for illustrating the difference between one-way clustering and biclustering.

utilizes item-based global similarity. The formula of CBI is $P_f = w_l \times P_l + w_g \times P_g + b$, where P_l and P_g denote the prediction rating of BBCF and IBCF, respectively, P_f is the final fused prediction rating, w_l and w_g are the weights of two prediction ratings, and b is the bias. The technical contributions of the proposed CBI method can be summarized as follows:

- We propose a novel BBCF method with the modified fuzzy ART neural network. The biclustering is composed of two fuzzy ART neural networks; the first is a standard fuzzy ART that is used to cluster the item dimension, and the second is a modified fuzzy ART that is used to cluster the user dimension. The modification is that an additional bicluster test is added to user clustering.
- A novel similarity method that considers the number of users' co-rated items is proposed.
- Extensive experiments on three datasets are carried out, and the results demonstrate the excellent prediction performance of the proposed method.

The remaining sections of the paper are organized as follows. Related works are described in Section 2. Section 3 provides a detailed description of the proposed method. Section 4 presents the experiments. Conclusion and future works are given in Section 5.

2 Related Work

CF plays an important role in the recommendation system. Many CF methods have been proposed. Typical CF methods include baseline, model-based, and memory-based CF and their variants.

Considering that CF data exhibit large user and item effects, several users provide higher/lower rating values than others, whereas several items receive higher/lower rating values than others. Baseline[18] adjusts the data by considering user and item effects. The baseline limitation lies in ignoring local personalization consideration, and thus it cannot differentiate the users' personalized rating range from that of the system. Baseline evaluates by using the sum of global average, average user deviation, and average item deviation, as shown in the following:

$$\mu = \frac{\sum_{u \in U_i, i \in I_u} r_{u,i}}{n_r} \tag{1}$$

$$b_u = \frac{\sum_{i \in I_u} (r_{u,i} - \mu)}{|I_u|} \tag{2}$$

$$b_i = \frac{\sum_{u \in U_i} (r_{u,i} - \mu - b_u)}{|U_i|} \tag{3}$$

$$p_{u,i} = \mu + b_u + b_i \tag{4}$$

where μ denotes the global average, $r_{u,i}$ denotes the rating for user u on item i, n_r denotes the number of ratings, I_u denotes the items rated by user u, U_i denotes the users rated by item i, and b_u and b_i are the average user and item deviations, respectively. The model-based method builds a model with the rating data[19–25]. A typical method is matrix completion, which can recover the empty elements in the rating matrix by analyzing non-empty elements. Matrix completion assumes that a user's rating is affected by only a few factors and the rating matrix has a low rank. For example, Deep Learning based Matrix Completion (DLMC)[25] is based on a multiple-hidden-layers neural network. The original data are first compressed into low-dimensional latent data through nonlinear mapping, and then the low-dimensional latent data are mapped back to reconstruct the original data through multiple stages. The missing entries in the original data and the parameters in the deep learning structure can be optimized simultaneously by minimizing the reconstruction error. However, the limitation of DLMC is that in real applications, the rating matrix may not be low rank, violating the assumption of the rating matrix. The model-based method requires expensive model-building and a trade-off between prediction accuracy and scalability.

The memory-based CF method is easy to implement, scales well, and can easily handle incremental data; thus, it is frequently used. The representative works are IBCF[17] and User-Based CF (UBCF)[26]. The workflow of IBCF is as follows: first, the similarity between item i and the other items is calculated in the following:

$$s(i,j) = \frac{\sum\limits_{u \in U_i \cap U_j} (r_{u,i} - \overline{r}_u)(r_{u,j} - \overline{r}_u)}{\sqrt{\sum\limits_{u \in U_i \cap U_j} (r_{u,i} - \overline{r}_u)^2} \sqrt{\sum\limits_{u \in U_i \cap U_j} (r_{u,j} - \overline{r}_u)^2}} \tag{5}$$

where $s(i,j)$ is the similarity (Pearson correlation coefficient) of items i and j, $U_i \cap U_j$ denotes the users who rate both items i and j, and $\overline{r}_u$ denotes the average of the the ratings of user u.

Then, the similarities are ranked, and the top-n items are selected as the neighbor items. Finally, we predict $p_{u,i}$ in the follwing, which is the weighted average of neighbors' ratings,

$$p_{u,i} = \frac{\sum_{j \in N_i} s(i,j) \times r_{u,j}}{\sum_{j \in N_i} |s(i,j)|} \tag{6}$$

where N_i denotes neighbor items.

UBCF also shares a similar process with IBCF. First, the similarities between the active and whole users in the

entire rating matrix are calculated. Then the similarities are sorted, and the top-n users are selected as neighbors. Finally, the unknown rating is predicted in the following, which is the sum of the average rating of user u and the weighted average of deviations from the neighbors' mean,

$$s(u,u') = \frac{r_u^{\mathrm{T}} \times r_{u'}}{\|r_u\|_2 \, \|r_{u'}\|_2} \tag{7}$$

$$p_{u,i} = \bar{r}_u + \frac{\sum_{u' \in N_u} s(u,u')(r_{u',i} - \bar{r}_{u'})}{\sum_{u' \in N_u} |s(u,u')|} \tag{8}$$

where $s(u,u')$ is the similarity (cosine correlation coefficient) between users u and u', and N_u denotes the neighbor users.

In recent years, many variants of UBCF and IBCF have been proposed. Among these variants, new similarity measures proposed to improve the perception capability of sparse data are one hot topic. For example, Singh et al.[27] proposed a novel similarity measure (modified Bhattacharya coefficient) to improve prediction accuracy. Jiang et al.[28] proposed an information entropy-based similarity measure to compute the similarity between users. Except for the modification of the similarity metric, calculating the similarity with a genetic algorithm is another research direction[29,30], where the similarity is not calculated with the Pearson correlation coefficient or cosine coefficient. The similarity is randomly initialized and finally optimized with the genetic algorithm. Another variant is the modification of CF methods. Given that the neighbors have a considerable impact on the performance of CF systems, incorporating clustering to handle sparsity problems is another hot topic[4,31]. For example, Sarwar et al.[4] incorporated one-way clustering (k-means) into CF to partition the users into various clusters, similar to the rows in Fig. 1. The neighbors of the active user are selected by looking into its cluster. The users in the same cluster are considered similar, and the prediction is solely based on the clusters instead of the whole dataset. Incorporating clustering into CF can find better neighbors, improving prediction accuracy.

Each CF method has superiority and limitations. The shortcomings are commonly overcome by fusing the prediction results of different methods[28,32,33]. With fusion, different methods can complement each other. Usually, the fusion result is the weighted sum of two prediction ratings. In literature, fusion's common mathematical formula is $P_f = w_1 \times P_1 + w_2 \times P_2$, where P_1 and P_2 denote the prediction ratings of two methods; P_f is the final fused prediction rating; and w_1 and w_2 are the weights of two prediction ratings. In some cases, the sum of w_1 and w_2 is limited to 1[28]. Apart from fusing the prediction ratings of two models, other kinds of fusions have also been proposed, for example the fusion in Ref. [34] is performed by combining different similarities instead of different predictions.

3 Method

In this section, each step of the proposed CBI method is described in detail. Figure 2 vividly describes the CBI flowchart. First, the whole dataset is divided into training and test subsets. Subsequently, fuzzy ART is adopted to mine biclusters from the training subset and is then utilized to obtain the local prediction vector P_{tr}^l. IBCF is used to obtain the global prediction vector P_{tr}^g. P_{tr}^l, P_{tr}^g, and R_{tr} (a vector composed of the whole non-empty ratings in the training subset) in the training subset are used to train the linear regression model (w^g, w^l, and b). Thus, three optimal parameters, w^{g*}, w^{l*}, and

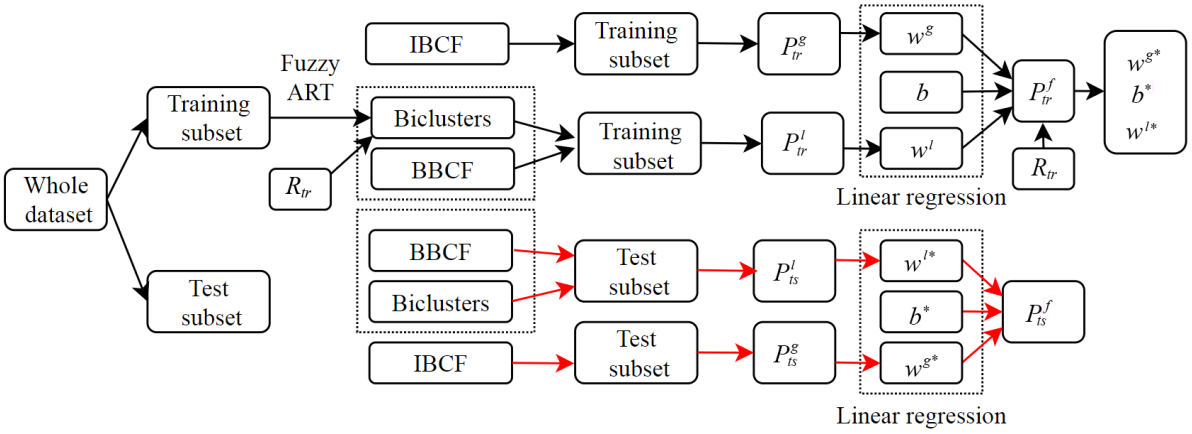


Fig. 2 Workflow of the proposed method.

b^* can be determined. In the training stage, the objective is to find the biclusters and obtain the optimal parameter value of the linear regression model. Finally, in the test stage, the biclusters obtained in the training stage are used to determine the local predictions vector P_{ts}^l. IBCF is then used to obtain the global prediction vector P_{ts}^g. Then with linear regression, P_{ts}^l and P_{ts}^g can be fused to obtain the final prediction vector P_{ts}^f.

3.1 Dataset separation

The whole dataset $M \in \mathbf{R}^{m\times n}$ (m and n denotes the number of rows and columns in M, respectively) with N_m rating scores beging divided into a training subset M_{tr} with N_{tr} non-empty ratings and a test subset M_{ts} with N_{ts} non-empty ratings. For example, if M has 1000 ratings, from which 800 are selected to compose M_{tr}, the remaining 200 ratings are used to construct M_{ts}. The separation is performed by randomly selecting 20% from the non-empty ratings of each user instead of randomly selecting from the whole users' ratings. Such separation ensures that in the training subset, no user has very few non-empty ratings. A user who has few non-empty ratings is nearly impossible to contain in any bicluster. Figure 3 shows an example of this separation. The above matrix M is the original complete dataset. The bottom left subfigure is the training subset, and the bottom right subfigure is the test subset. The rows represent users, and the columns represent items. In M, M_{ij} represents the rating value of the j-th item rated by the i-th user. $M_{ij} = 0$ indicates that the rating value is empty, and the j-th item has not been rated by the i-th user. The sizes of the training and test subsets are the same as those of the original complete dataset. In addition, the fuzzy ART module requires normalized data, and therefore M_{tr} is normalized in the following:

$$M_{tr}^n = \frac{M_{tr} - min(M)}{max(M) - min(M)} \tag{9}$$

where M_{tr}^n is the normalized result of M_{tr}, $min(M)$ and $max(M)$ are the minimum and maximum of M, respectively. After normalization, all elements in M_{tr}^n are in the range of [0, 1]. M_{ts} is normalized in same way.

3.2 Training stage

Having obtained the training and test subsets from the above stage, the next step is to mine biclusters in two steps, namely, item clustering and user clustering. Both steps are performed with Fuzzy ART (FA)[15,35].

3.2.1 Introduction of fuzzy ART

Figure 4 illustrates the FA which contains three parts, two neuron layers, and one orienting subsystem. F_1 is the feature layer, F_2 is the category layer, and their neurons are connected by weight W. The number of neurons in F_1 layer is the same as that of features in the input pattern A. Each neuron in F_2 layer represents one cluster (category). Initially, F_2 layer has only one uncommitted neuron, and neurons are incrementally added with the coming of new input patterns, while always maintaining one additional uncommitted neuron. The vigilance threshold ρ determines which category the input pattern belongs to. If the closeness between the input pattern and the current winning neuron (category) is smaller than ρ, then the orienting subsystem resets that neuron and turns to the next winning neuron. If

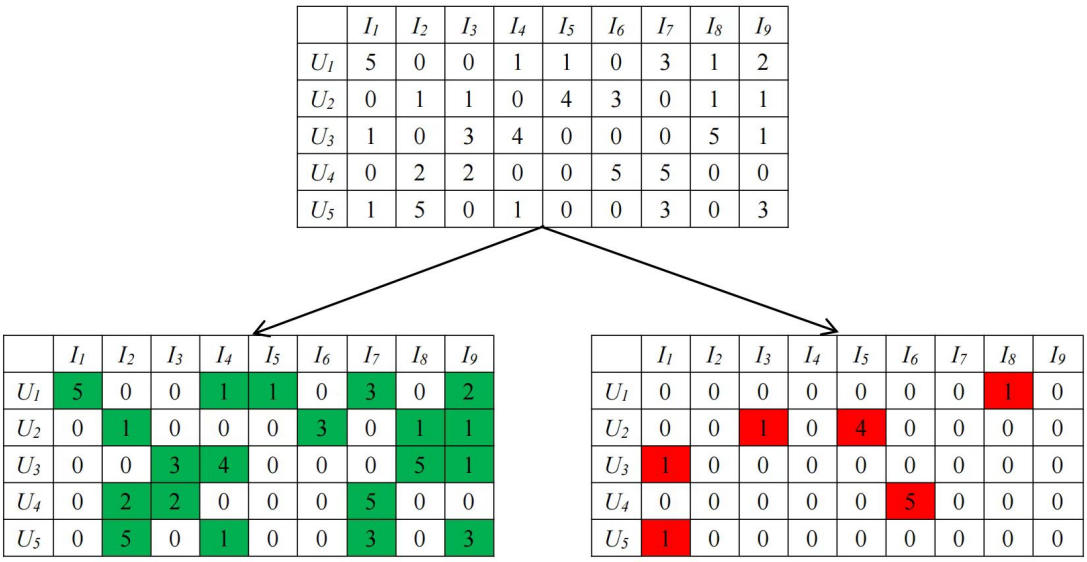


Fig. 3 Example for illustrating dataset separation.

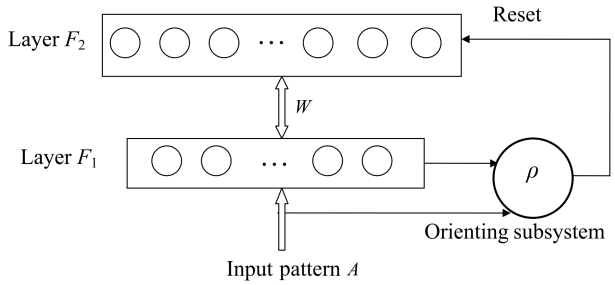


Fig. 4 Fuzzy ART[35].

any existing neuron (category) passes the vigilance test, then the input pattern is assigned to the corresponding neuron, and the weight of the corresponding neuron is updated for learning the information contained in the input pattern. If none of the committed neurons passes the vigilance test, then the weights of the uncommitted neuron are updated and thus become a newly committed neuron. Besides, a new uncommitted neuron is added to F_2 layer.

3.2.2 Biclusters searching

Biclustering can be considered as the combination of column and row clusterings. Item clustering is performed with the standard FA model FA_i[15]. The inputs are item vectors $M_{tr}^n(:, j)$ $(j = 1, 2, \ldots, n)$, vigilance parameter ρ_i, choice parameter α_i, learning parameter β_i, and maximal epochs ME_i. Finally, the item cluster set $CI = \{CI_1, CI_2, \ldots, CI_{n_1}\}$ composed of n_1 item clusters can be obtained.

User clustering is performed with FA_u, a modified FA. The pseudocode of user clustering is displayed in Algorithm 1. The modification lies in that when an input pattern (user vector) passes the vigilance test, subsequently, the bicluster test needs to be passed before assigning to the winning neuron (user cluster). The bicluster test is that if the correlation between the input pattern and the bicluster set is not smaller than a predetermined threshold T_c, then the input pattern is assigned to the winning neuron; otherwise, increase ρ_u (the vigilance parameter, and ρ_s is its increment) and rerun FA_u with the updated ρ_u. Given that high vigilance threshold leads to narrow generalization and the neuron (cluster) represents fewer input patterns, if no existing committed neuron (cluster) passes the bicluster test, ρ_u must be enlarged. With FA_u, user cluster set $CU = \{CU_1, CU_2, \ldots, CU_{n_2}\}$ composed of n_2 user clusters can be obtained. When an input pattern is presented to the F_1 layer of FA_u, the following steps are performed:

(1) The input U is complement-coded to avoid category proliferation. The complement coding is performed with $A = (U, U^c)$, where U_c is the complement of U, and $U_i^c = 1 - U_i, \forall i = 1, \ldots, n$.

Algorithm 1 FA_u algorithm

Require: M_{tr}^n, ρ_u, α_u, β_u, ME_u, T_c, and ρ_s
Ensure: CU
1: Initialize FA_u.
2: **while** W is not converged and ME_u is not reached **do**
3: **for** $i = 1; i = i + 1; i \leqslant m$ **do**
4: $U \leftarrow M_{tr}^n(i, :)$, $\rho \leftarrow \rho_u$, $A \leftarrow (U, U^c)$.
5: **loop**:
6: Present A to F_1 layer of FA_u.
7: Calculate the cluster choice function values.
8: Calculate the match function value.
9: Select the potential winning neuron n_{pw}.
10: **while** n_{pw} fails vigilance test **do**
11: Shut-off n_{pw}, select a new n_{pw}.
12: **end while**
13: **if** n_{pw} passes bicluster test **then**
14: n_{pw} becomes winning neuron n_w, assign U to the n_w, update weight of n_w.
15: **else**
16: $\rho_u \leftarrow \rho_u + \rho_s$, **goto loop**.
17: **end if**
18: **end for**
19: **end while**
20: **return** user cluster set CU.

(2) Each neuron in F_2 calculates a value for the cluster choice function, which selects the neuron (n_c) having maximal function value, as shown in the following:

$$T_j = \frac{|A \wedge W_j|}{|W_j| + \alpha_u} \tag{10}$$

where W_j is the weight of the j-th neuron, "$\wedge$" is the fuzzy MIN operation defined by $(A \wedge W_j)_k = \min(A_k, W_{j,k})$, $W_{j,k}$ is the value of the j-th neuron's k-th weight. "||" means taking the first-order norm, and α_u is a very small positive real number.

(3) Calculate the match function value of n_c as follows:

$$M_j = \frac{|W_j \wedge A|}{|A|} \tag{11}$$

if $M_j \geqslant \rho_u$, n_c passes the vigilance test and becomes the potential winning neuron n_{pw}. If $M_j \leqslant \rho_u$, then apparently n_c cannot be the potential winning neuron and is shut off by the orienting subsystem. The competition among the remaining neurons is rerun until the potential winning neuron n_{pw} is found.

(4) Combine CU_u (the user cluster that contains n_{pw}) and each item cluster in CI to construct biclusters. Then, calculate the correlation (AC) between the bicluster and input pattern as follows:

$$S_l(u,v) = \frac{\sum\limits_{i\in S}(r_{u,i}-\overline{r}_u)(r_{v,i}-\overline{r}_v)}{(1+\mathrm{e}^{-\frac{n}{2}})\sqrt{\sum\limits_{i\in S}(r_{u,i}-\overline{r}_u)^2}\sqrt{\sum\limits_{i\in S}(r_{u,j}-\overline{r}_v)^2}} \quad (12)$$

$$AC(B_k,U) = \frac{1}{|R|}\times\sum_{j=1}^{|R|} S_l(B_k(j,:),U) \quad (13)$$

where $S = I_u \cap I_v$ denotes the items co-rated by users u and v. $N_S = |S|$ is the cardinality of S. S_l is the local similarity measure of two users and is the modified Pearson correlation coefficient. The difference lies in that S_l considers the number of co-rated items n_s. The bigger n_s becomes, the more items two users rate simultaneously, and the higher their similarity becomes. B_k is the k-th bicluster, $B_k(j,:)$ represents the j-th row of B_k, and $|R|$ is the number of rows (users) in B_k.

If at least one of these passes the bicluster test ($AC \geqslant T_c$), then set n_{pw} as the final winning neuron n_w. Otherwise, increase the vigilance with $\rho_u = \rho_u + \rho_s$ and jump to the second step to rerun.

(5) Update the weight of the winning neuron n_w for learning the information contained in the input pattern as follows:

$$W_j = \beta_u(W_j \wedge A) + (1-\beta_u)W_j \quad (14)$$

where $\beta_u \in [0,1]$ is the learning rate.

With the above item and user clustering, the n_1 item clusters and n_2 user clusters are generated, $n_1 \times n_2$ biclusters can be obtained by pairwise coupling CU and CI. The next step is to obtain local predictions with the mined biclusters and BBCF from M_{tr}^n.

3.2.3 Local prediction based on BBCF

With biclusters obtained from the M_{tr}^n, the next step is to mine local predictions from M_{tr}^n. To predict $p_{u,i}^l$ which is the local prediction of user u on item i, we must first find where the user cluster CU_u user u belongs to and the item cluster CI_i item i belongs to. Second, combine CU_u and CI_i to construct a bicluster B. Then, the normalized prediction $p_{u,i}^n$ can be calculated,

$$p_{u,i}^n = \overline{r}_u + \frac{\sum_{u'\in U_{\overline{B}}} S_l(u,u')(r_{u',i}-\overline{r}_{u'})}{\sum_{u'\in U_{\overline{B}}}\left|S_l(u,u')\right|} \quad (15)$$

where $U_{\overline{B}}$ denotes the neighbor users (whole users excluding u in B); S_l is calculated as Eq. (12) and is the local similarity between users u and u'; and $\overline{r}_{u'}$ shares similar definition.

If $p_{u,i}^n \leqslant 0$, then it is reset as 0. If $p_{u,i}^n \geqslant 1$, then it is reset as 1. Finally, the final unnormalized prediction can be calculated,

$$p_{u,i}^l = p_{u,i}^n \times (\max(M)-\min(M)) + \min(M) \quad (16)$$

where $p_{u,i}^n$ is in the range of [0, 1]. With Eq. (16), $p_{u,i}^n$ can be transformed to be unnormalized local prediction $p_{u,i}^l$ with range $[\min(M), \max(M)]$.

3.2.4 Global prediction based on IBCF

Having obtained the user-based local prediction, the next step is to obtain an item-based global prediction with IBCF[17] from M_{tr}. The global prediction $p_{u,i}^g$ of user u on item i is calculated by Eq. (6).

3.2.5 Combination of local and global predictions

To boost the prediction accuracy, we combine the local prediction $p_{u,i}^l$ and global prediction $p_{u,i}^g$ to overcome each of their shortcomings. The combination can be deemed as a quadratic linear regression model,

$$p_{u,i} = w^l \times p_{u,i}^l + w^g \times p_{u,i}^g + b \quad (17)$$

where b is a constant bias; w^l and w^g are the weights of both predictions and are constrained in the range of [0, 1]; and b is constrained in the range of $[\min(M) - \max(M), \max(M) - \min(M)]$.

Figure 2 shows that all $p_{u,i}^l$ in the training subset can be gathered to construct a vector P_{tr}^l, and all $p_{u,i}^g$ in the training subset can be gathered to construct a vector P_{tr}^g. The optimal values of three linear regression parameters, w^{l*}, w^{g*}, and b^* can be determined by minimizing the absolute errors between the prediction vector $(w^l \times P_{u,i}^l + w^g \times P_{u,i}^g + b)$ and true rating vector R_{tr} in the training dataset. The three optimal values can be applied in the test stage to obtain the predictions of the test subset.

3.3 Test stage

With the obtained biclusters and the optimal parameter values of linear regression in the training stage, the next step is to predict each non-empty rating in the test subset M_{ts}. The local prediction vector P_{ts}^l can be calculated with the biclusters mined from the training subset and BBCF, while the global prediction vector P_{ts}^g can be calculated with IBCF. Finally, with linear regression, the test subset final prediction vector $P_{ts}^f = w^{l*} \times P_{ts}^l + w^{g*} \times P_{ts}^g + b^*$ can be obtained.

4 Experiment

4.1 Experimental settings

Three frequently used datasets are used for evaluating the performance of the proposed method. Table 1 shows the detailed description of the two GroupLens MovieLens

Table 1 Details of datasets.

Dataset	N_u	N_i	N_r	Scale
ml-100k	943	1682	100 000	[1, 2, 3, 4, 5]
ml-small	610	9724	100 836	[0.5, 1, . . . , 5]
Kaggle	671	9066	100 004	[0.5, 1, . . . , 5]

datasets† and one Kaggle movie dataset‡. N_u denotes the number of users; N_i denotes the number of items; N_r represents the number of non-empty ratings; and scale is the range of ratings. The sparsity levels of the three datasets are all over 90%.

Evaluating the performance of the proposed CBI method commonly uses three measures, namely, Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). The three measures can reflect the difference between the true and the predicted ratings. The definitions of MAE, MSE, and RMSE are shown in the following:

$$\text{MAE} = \frac{\sum_{i=1}^{n_r} |P_i - R_i|}{n_r} \tag{18}$$

$$\text{MSE} = \frac{\sum_{i=1}^{n_r} (P_i - R_i)^2}{n_r} \tag{19}$$

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n_r} (P_i - R_i)^2}{n_r}} \tag{20}$$

where P_i and R_i are the predicted and true ratings, respectively.

To better validate the performance of proposed framework, we compare CBI with six other CF algorithms (baseline[18], IBCF[17], UBCF[26], DLMC[25], Clust[4], and BIMARS[9]). The main principles of the comparison methods are introduced in Sections 1 and 2.

All the methods are implemented with MATLAB programming language. The release name of the MATLAB software is R2015a, and it is run on a workstation configured with Ubuntu 16.04.5 operating system, 4 TB size disk, and 256 GB memory. The workstation has 12 CPU cores whose type is Intel(R) Xeon(R) Gold 5118 @ 2.30 GHz. Table 2 shows the optimal parameter values of all methods determined with grid search. Notably, the baseline method has no parameter and therefore is not listed in Table 2.

4.2 Experimental result and analysis

4.2.1 Cross validation

First, a 5-fold cross validation experiment is performed, and the dataset partition is carried out as previously described in Section 3. The training subset is considered as the known history rating information and used to mine biclusters and determine the linear regression parameter values. The test subset is used to obtain prediction errors.

Table 2 Parameters for all methods.

Method	Parameter setting
CBI	$\rho_i = 0.05, \alpha_i = 1 \times 10^{-5}; \beta_i = 1; ME_i = 1000;$ $\rho_u = 0.09, \alpha_u = 1 \times 10^{-5}; \beta_u = 1; ME_u = 1000;$ $T_c = 0.02; \rho_s = 0.01;$
UBCF	ml-100k: $n_u = 5$; ml-small: $n_u = 1$; Kaggle: $n_u = 1$
IBCF	$n_i = 200$
DLMC	$wdp = 0.01, n_2 = 2, n_u = \{50, 20\},$ Activition functions: tangent and sigmoid
Clust	$k = 60$
BIMARS	$n_p = 200$

Figure 5 shows the cross validation experiment results (average of three measures) on three datasets. The results show that CBI outperforms the other six methods in terms of all measures on all datasets. The average *MAE* of CBI on three datasets is nearly identical, approximately 0.62, improving approximately 0.07 compared with the best method, BIMARS. The finding that CBI is better than IBCF and UBCF can be explained by its use of both local and global information, while the latter two only use global information. The result that CBI outperforms Clust is due to two reasons. The first

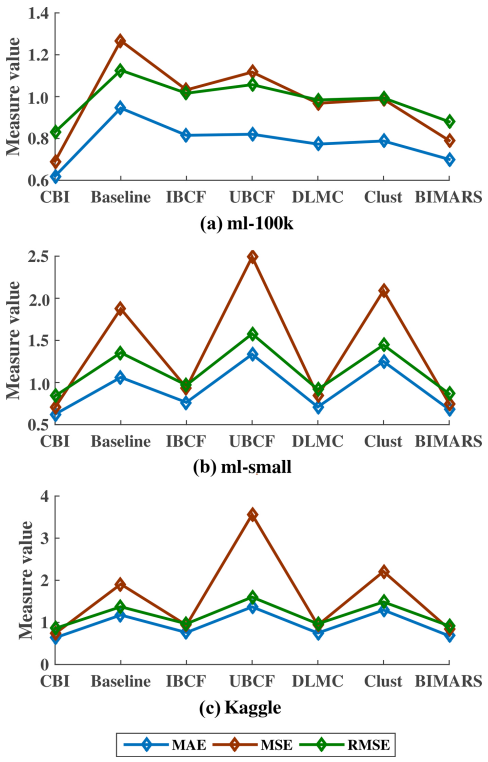


Fig. 5 Cross validation results comparison.

† https://grouplens.org/data sets/movielens/

‡ https://www.kaggle.com/rounakbanik/the–movies–data set

reason is that the similarity measure of Clust does not consider the effect of the number of co-rated items. The second reason is that Clust clusters only user dimensions and find worse neighbors than CBI. DLMC assumes that the rating matrix is low rank. However, this may not be the case in real applications, thus affecting the prediction accuracy of DLMC. As for BIMARS, which removes the ratings, much information is missing, and similar users found may be incorrect. In addition, BIMARS uses a memetic algorithm to optimize the similarity vector. Given the randomness of the memetic algorithm, BIMARS may fail to find an optimal similarity vector. The baseline is simply the sum of the global average, user bias, and item bias. Treating all users' ratings equally, its main limitation is the neglect of local personalization. The baseline cannot differentiate users' personalized rating range from the system's rating range and thus produces irrational predictive values.

In investigating the contribution of IBCF-based prediction and BBCF-based prediction, the values of w^{l*} and w^{g*} need to be analyzed. As expected, for the proposed CBI method, w^{l*} is much bigger than w^{g*} on all datasets. This result indicates that BBCF-based prediction plays a more important role than IBCF-based prediction. For example, in the ml-100k dataset, the average optimal values of w^{l*}, w^{g*}, and b^* are 1, 0.12, and -0.35, respectively.

The Absolute Error (AE) statistics is also investigated to know more details about the difference between the predicted and true ratings of the test subset. Figure 6 shows the histogram of the AE of all methods on the ml-100k dataset. Notably, given that in BIMARS, the trivial ratings are deleted, the test dataset size is smaller than that of other methods. Thus, comparing BIMARS with other methods in terms of AE statistics may be inappropriate and is therefore not displayed in Fig. 6. Given that the range of the ratings in the ml-100k dataset is [1, 5], the AE is divided into five ranges, namely, [0, 1), [1, 2), [2, 3), [3, 4), and [4, 5]. The proposed CBI produced the maximal number of [0–1) AEs and the minimal number of [1, 2), [2, 3), [3, 4), [4, 5] AEs, obtaining the smallest error. For the other two datasets, the AE histogram shares similar trends.

4.2.2 Sparsity test

Given that CF is limited by the sparsity of the dataset, testing the performance of CBI under different sparsity levels is necessary. Table 3 shows that nine pairs of training (S_{tr}) and test subset ratios (S_{ts}) are split.

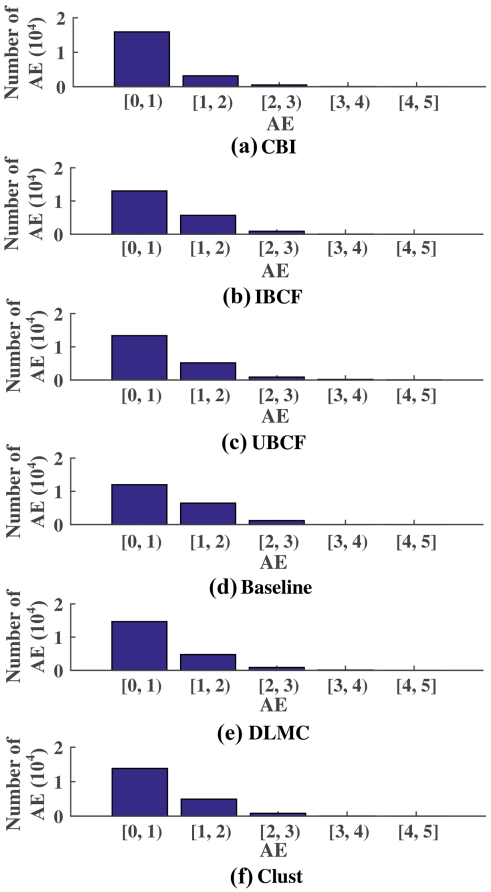


Fig. 6 AE histogram comparison on ml-100k dataset.

Table 3 Different sparsity levels.

Level	S_{tr}	S_{ts}	Level	S_{tr}	S_{ts}
1	0.9	0.1	6	0.4	0.6
2	0.8	0.2	7	0.3	0.7
3	0.7	0.3	8	0.2	0.8
4	0.6	0.4	9	0.1	0.9
5	0.5	0.5			

Different training subset sizes correspond to varying sparsity levels, and a high training ratio means a low sparsity level. The sparsity test results on three datasets are displayed in Fig. 7. CBI obtains lower MAE, MSE, and RMSE than all comparison methods on all sparsity levels on all datasets. On the whole, MAE, MSE, and RMSE have similar trends with the change of sparsity levels on the same dataset. With the increase in sparsity level, most methods obtain bigger prediction errors. The reason is that a higher sparsity level means fewer non-empty ratings in the training subset, and less information about users' preferences can be mined, thereby reducing prediction accuracy. Certain methods on some datasets (such as Clust on ml-small and Kaggle) show no huge fluctuations on all sparsity levels, possibly due to the

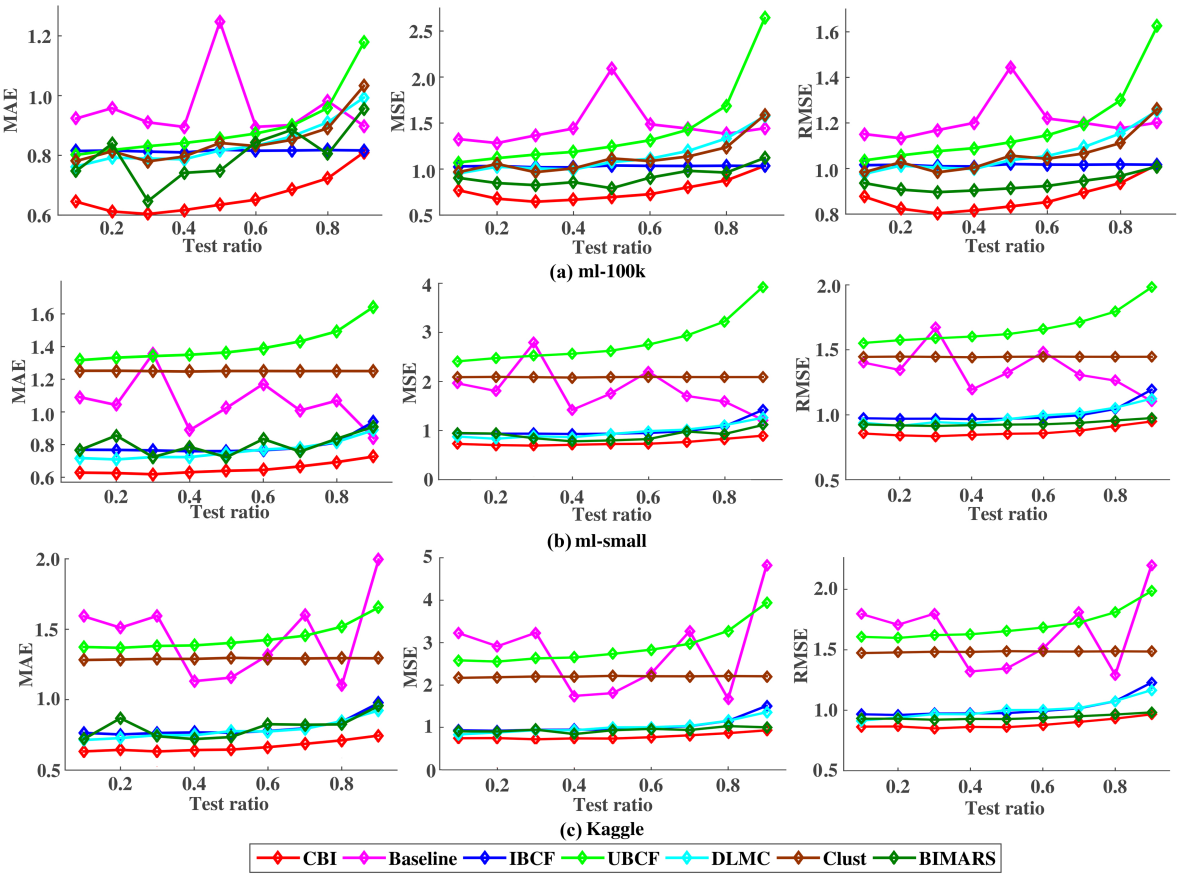


Fig. 7 Sparsity test results comparison.

effect of the dataset. The difference between Clust and UBCF is only the incorporation of selecting neighbors with k-means. Clust is better than UBCF in all cases, demonstrating the superiority of clustering. From the nine subfigures, we can summarize that CBI prediction performance is robust and excellent with changes in sparsity level. The above experiment results show that the proposed CBI outperforms all the comparison methods in all aspects on the three data sets, demonstrating the robustness, effectiveness, and superiority of CBI. The excellent performance of CBI is mainly contributed by the biclustering-based local prediction and demonstrates its effectiveness.

5 Conclusion and Future Work

A recommender system is a powerful tool for extracting customer preferences from historical data. In this study, to solve the sparsity problem of recommender systems, we propose a novel CF method named CBI because of the combined BBCF and IBCF. Neighbors and similarity measures are two important parts of CF, and therefore a novel biclustering method based on modified fuzzy ART and a novel similarity measure is proposed. Biclustering can mine local information that is ignored by global prediction methods. Biclustering-based CF is proposed to obtain local predictions, which are then fused with the global prediction to obtain a final fused prediction. Experiment results on three datasets show that the proposed CBI greatly outperforms all comparison methods in many aspects, demonstrating its effectiveness and superiority.

For future works, considering that users' preferences may change over time[36], the newly found bicluster must be more important than the previously found bicluster. Temporal information can be included to boost prediction accuracy. In addition, new rating data is always generated, and the proposed method can be improved for application to incremental cases. Furthermore, BBCF is only used to solve the data sparsity problem in this study. In the future, BBCF can be investigated for other problems, such as cold-start in the recommender system.

Acknowledgment

This work was supported by Ningbo Natural Science Foundation (No. 202003N4057) and the National Natural

Science Foundation of China (Nos. 62172336 and 62032018).

References

[1] J. Lu, D. S. Wu, M. S. Mao, W. Wang, and G. Q. Zhang, Recommender system application developments: A survey, *Decis. Support Syst.*, vol. 74, pp. 12–32, 2015.

[2] M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro, Semantics-aware content-based recommender systems, in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, eds. Boston, MA, USA: Springer, 2015, pp. 119–159.

[3] R. Burke, Hybrid recommender systems: Survey and experiments, *User Model. User-Adap Inter.*, vol. 12, no. 4, pp. 331–370, 2002.

[4] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, Recommender systems for large-scale E-commerce: Scalable neighborhood formation using clustering, *Communications*, vol. 50, no. 12, pp. 158–167, 2002.

[5] Y. C. Lu, K. Tozuka, G. Chakraborty, and M. Matsuhara, A novel item cluster-based collaborative filtering recommendation system, *Rev. Socionetwork Strateg.*, vol. 15, no. 2, pp. 327–346, 2021.

[6] J. J. Bu, X. Shen, B. Xu, C. Chen, X. F. He, and D. Cai, Improving collaborative recommendation via user-item subgroups, *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2363–2375, 2016.

[7] J. Lee, S. Bengio, S. Kim, G. Lebanon, and Y. Singer, Local collaborative ranking, in *Proc. 23rd Int. Conf. on World Wide Web*, Seoul, Republic of Korea, 2014, pp. 85–96.

[8] J. Lee, S. Kim, G. Lebanon, and Y. Singer, Local low- rank matrix approximation, in *Proc. 30th Int. Conf. on Machine Learning*, Atlanta, GA, USA, 2013, pp. 82–90.

[9] S. Bansal and N. Baliyan, Bi-MARS: A bi-clustering based memetic algorithm for recommender systems, *Appl. Soft Comput.*, vol. 97, p. 106785, 2020.

[10] Q. H. Huang, X. H. Huang, Z. F. Kong, X. L. Li, and D. C. Tao, Bi-phase evolutionary searching for Biclusters in gene expression data, *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 803–814, 2019.

[11] Q. H. Huang, Y. D. Chen, L. Z. Liu, D. C. Tao, and X. L. Li, On combining biclustering mining and AdaBoost for breast tumor classification, *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 4, pp. 728–738, 2020.

[12] Q. H. Huang, J. Yang, X. F. Feng, A. W. C. Liew, and X. L. Li, Automated trading point forecasting based on bicluster mining and fuzzy inference, *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 2, pp. 259–272, 2020.

[13] J. J. Sun, Q. H. Huang, and X. L. Li, Determination of temporal stock investment styles via biclustering trading patterns, *Cogn. Comput.*, vol. 11, no. 6, pp. 799–808, 2019.

[14] J. M. Cotelo, F. J. Ortega, J. A. Troyano, F. Enríquez, and F. L. Cruz, Known by who we follow: A biclustering application to community detection, *IEEE Access*, vol. 8, pp. 192218–192228, 2020.

[15] G. A. Carpenter, S. Grossberg, and D. B. Rosen, Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system, *Neural Netw.*, vol. 4, no. 6, pp. 759–771, 1991.

[16] P. Thakkar, K. Varma, V. Ukani, S. Mankad, and S. Tanwar, Combining user-based and item-based collaborative filtering using machine learning, in *Information and Communication Technology for Intelligent Systems*, S. Satapathy and A. Joshi, eds. Singapore: Springer, 2019, pp. 173–180.

[17] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, Item-based collaborative filtering recommendation algorithms, in *Proc. 10th Int. Conf. on World Wide Web*, Hong Kong, China, 2001, pp. 285–295.

[18] Y. Koren, Factor in the neighbors: Scalable and accurate collaborative filtering, *ACM Trans. Knowl. Discov. Data*, vol. 4, no. 1, p. 1, 2010.

[19] T. L. Huang, R. J. Zhao, L. Q. Bi, D. F. Zhang, and C. Lu, Neural embedding singular value decomposition for collaborative filtering, *IEEE Trans. Neural Netw. Learn. Syst.*, doi: 10.1109/TNNLS.2021.3070853.

[20] Z. Wang, H. L. Chen, Z. Li, K. Lin, N. Jiang, and F. Xia, VRConvMF: Visual recurrent convolutional matrix factorization for movie recommendation, *IEEE Trans. Emerg. Top. Comput. Intell.*, doi: 10.1109/TETCI.2021.3102619.

[21] A. Ramlatchan, M. Y. Yang, Q. Liu, M. Li, J. X. Wang, and Y. H. Li, A survey of matrix completion methods for recommendation systems, *Big Data Mining and Analytics*, vol. 1, no. 4, pp. 308–323, 2018.

[22] X. Wang, H. Y. Jin, A. Zhang, X. N. He, T. Xu, and T. S. Chua, Disentangled graph collaborative filtering, in *Proc. 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Virtual Event, China, 2020, pp. 1001–1010.

[23] M. G. Vozalis and K. G. Margaritis, A recommender system using principal component analysis, in *Proc. 11th Panhellenic Conf. in Informatics*, Patra, Greece, 2007, pp. 271–283.

[24] J. C. Fan, L. J. Ding, Y. D. Chen, and M. Udell, Factor group-sparse regularization for efficient low-rank matrix recovery, in *Proc. 33rd Int. Conf. on Neural Information Processing Systems*, Vancouver, Canada, 2019, pp. 5104–5114.

[25] J. C. Fan and T. Chow, Deep learning based matrix completion, *Neurocomputing*, vol. 266, pp. 540–549, 2017.

[26] Z. D. Zhao and M. S. Shang, User-based collaborative-filtering recommendation algorithms on Hadoop, in *Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining*, Phuket, Thailand, 2010, pp. 478–481.

[27] P. K. Singh, P. K. D. Pramanik, and P. Choudhury, Mitigating sparsity using Bhattacharyya coefficient and items' categorical attributes: Improving the performance of collaborative filtering based recommendation systems, *Appl. Intell.*, vol. 52, no. 5, pp. 5513–5536, 2022.

[28] M. Y. Jiang, Z. F. Zhang, J. Q. Jiang, Q. H. Wang, and Z. L. Pei, A collaborative filtering recommendation algorithm based on information theory and bi-clustering, *Neural Comput. Appl.*, vol. 31, no. 12, pp. 8279–8287, 2019.

[29] D. Anand, Feature extraction for collaborative filtering: A genetic programming approach, *Int. J. Comput. Sci. Issues*, vol. 9, no. 1, pp. 348–354, 2012.

[30] M. Duma and B. Twala, Sparseness reduction in collaborative filtering using a nearest neighbour artificial immune system with genetic algorithms, *Expert Syst. Appl.*, vol. 132, pp. 110–125, 2019.

[31] C. G. Huang and J. Yin, Effective association clusters filtering to cold-start recommendations, in *Proc. 7th Int. Conf. on Fuzzy Systems and Knowledge Discovery*, Yantai, China, 2010, pp. 2461–2464.

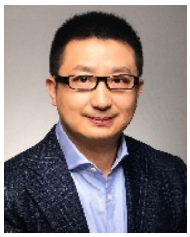
[32] S. Kant and T. Mahara, Merging user and item based collaborative filtering to alleviate data sparsity, *Int. J. Syst. Assur. Eng. Manag.*, vol. 9, no. 1 pp. 173–1797, 2018.

[33] K. S. Zhang and H. F. Li, Fusion-based recommender system, in *Proc. 13th Int. Conf. on Information Fusion*, Edinburgh, UK, 2010, pp. 1–7.

[34] Z. J. Yang, D. H. Xia, J. Liu, C. Zheng, Y. Z. Qu, Y. D. Chen, and C. J. Zhang, Fusion of internal similarity to improve the accuracy of recommendation algorithm, *J. Internet Things*, vol. 3, no. 2 pp. 65–76, 2021.

[35] X. Rui and D. C. Wunsch II, BARTMAP: A viable structure for biclustering, *Neural Netw.*, vol. 24, no. 7, pp. 709–716, 2011.

[36] C. X. Zhang, M. Yang, J. Lv, and W. Q. Yang, An improved hybrid collaborative filtering algorithm based on tags and time factor, *Big Data Mining and Analytics*, vol. 1, no. 2, pp. 128–136, 2018.



Yu Zhang received the double PhD degrees in computer science from Northwestern Polytechnical University, Xi'an, China in 2011, and RMIT University, Melbourne, Australia in 2020, respectively. He is currently an associate professor at the School of Computer Science, Northwestern Polytechnical University. His research interests include protocol design, wireless sensor networks, mobile computing, wearable health sensing, Internet of Things, and human-CPS resource management. He is a member of ACM and IEEE.

Jianjun Sun received the MEng degree in circuits and systems from South China University of Technology, Guangzhou, China in 2016. He is currently a PhD candidate at the School of Computer Science, Northwestern Polytechnic University, Xi'an, China. He has published several journal and conference papers. His research interests include data mining and artificial intelligence.