Knowledge and Information Systems (2020) 62:3133–3154
https://doi.org/10.1007/s10115-020-01455-2

REGULAR PAPER

Improved covering-based collaborative filtering for new users' personalized recommendations

Zhipeng Zhang[1] · Yasuo Kudo[2] · Tetsuya Murai[3] · Yonggong Ren[1]

Received: 28 December 2017 / Revised: 12 February 2020 / Accepted: 26 February 2020 /
Published online: 13 March 2020

Abstract
User-based collaborative filtering (UBCF) is widely used in recommender systems (RSs) as one of the most successful approaches, but traditional UBCF cannot provide recommendations with satisfactory accuracy and diversity simultaneously. Covering-based collaborative filtering (CBCF) is a useful approach that we have proposed in our previous work, which greatly improves the traditional UBCF and could provide satisfactory recommendations to an active user which often has sufficient rating information. However, different from an active user, a new user in RSs often has special characteristics (e.g., fewer ratings or ratings concentrating on popular items), and the previous CBCF approach cannot provide satisfactory recommendations for a new user. In this paper, aiming to provide personalized recommendations for a new user, through a detailed analysis of the characteristics of new users, we reconstruct a decision class to improve the previous CBCF and utilize the covering reduction algorithm in covering-based rough sets to remove redundant candidate neighbors for a new user. Furthermore, unlike the previous CBCF, our improved CBCF could provide personalized recommendations without needing special additional information. Experimental results suggest that for the sparse datasets that often occur in real RSs, the improved CBCF significantly outperforms those of existing work and can provide personalized recommendations for a new user with satisfactory accuracy and diversity simultaneously.

✉ Yasuo Kudo
kudo@csse.muroran-it.ac.jp

Zhipeng Zhang
zhipengzhang@lnnu.edu.cn

Tetsuya Murai
t-murai@photon.chitose.ac.jp

Yonggong Ren
ryg@lnnu.edu.cn

1 School of Computer and Information Technology, Liaoning Normal University, Dalian 116029, China

2 College of Information and Systems, Muroran Institute of Technology, Muroran 050-8585, Japan

3 Faculty of Science and Technology, Chitose Institute of Science and Technology, Chitose 066-8655, Japan

Keywords Covering-based rough sets · User-based collaborative filtering · Covering reduction · Personalized recommendations · New user cold-start

1 Introduction

Rapid economic and technological development has led to people's requirements becoming more personalized. Recommender systems (RSs), which can recommend personalized objects (e.g., books, CDs, movies, and news), are widely used applications in daily life, for Web sites such as Amazon and Netflix. Their great commercial value and research potential have rendered RSs increasingly significant in recent years [3,4,16].

Currently, although most studies focus on developing new approaches to improve RS accuracy, it has been argued that using only an accuracy metric to evaluate RSs is not sufficient and that the diversity of recommendations must also be considered as an important evaluation measure [6,7,13,17,29,32]. Because in a real business environment, a company can use RSs to obtain more benefits by providing recommendations with higher diversity. For example, as there are many movies in the statistical long tail that have only a few ratings, it would be profitable for Netflix if RSs would encourage users to rent movies in the long tail, because these are less costly to license and acquire from distributors than new releases or highly popular movies. However, recent studies have shown that it is very difficult to obtain a reasonable trade-off between the accuracy and diversity of an RS [20,37], because increasing the diversity of recommendations is usually accompanied by a loss in accuracy [14].

Collaborative filtering (CF) approaches are popularly used in RSs owing to their satisfactory performance [28,35]. On the assumption that users who have similar preferences in the past will tend to have similar tastes in the future, user-based collaborative filtering (UBCF) has been proposed and applied in practice [12]. UBCF is simple and efficient, and it can provide satisfactory recommendations utilizing only the user's historical ratings, so it has been demonstrated remarkable success in RSs. However, the traditional UBCF usually cannot provide recommendations with satisfactory accuracy and diversity at the same time [6]. In addition, recent researches have concluded that improving recommendation diversity can frequently be accompanied by losing recommendation accuracy, making it difficult to select reasonable trade-off between accuracy and diversity [14,20,23,37]. Many studies have been conducted to increase the diversity of recommendations based on UBCF. Among these studies, some approaches can improve diversity significantly, but accompanied by losses in accuracy [1,2]. Although some methods can improve accuracy and diversity simultaneously, they require additional information that is often not available or incomplete [9,21]. Covering-based collaborative filtering (CBCF) is a useful approach, falling in the latter research line mentioned above, that we proposed in our previous work to improve accuracy and diversity of the traditional UBCF by utilizing covering-based rough sets [34,36]. However, all of these studies focus on providing satisfactory recommendations for an active user which often has sufficient rating information, but a new user in RSs differs in some aspects (e.g., fewer ratings or ratings concentrating on popular items), recommendation difficulty is increased [5,19,27,30,33]. Therefore, researchers face the difficult problem of how to utilize only easily obtained information to provide recommendations for a new user with satisfactory accuracy and diversity simultaneously.

In this paper, we aim to improve the previous CBCF to provide satisfactory accuracy and diversity of recommendations simultaneously for a new user in RSs. Because a new user often has few ratings, the previous CBCF cannot utilize the insufficient information to remove redundant candidate neighbors for a new user effectively. In our improved CBCF, in order to remove as many redundant candidate neighbors as possible for a new user, by analyzing the proportion and characteristic of new users' rating scores, we reconstruct the decision class by the niche items which have fewer ratings from users. In this way, different from the previous CBCF, our improved CBCF could remove redundant candidate neighbors efficiently without requiring any special additional information. Experimental results indicate that our improved CBCF can not only improve the accuracy metric, but also increase the diversity of recommendations for a new user for the sort of sparse datasets that often occur in connection with real RSs.

The remainder of this paper is organized as follows. In Sect. 2, we introduce the traditional UBCF approach and review some studies that attempt to improve UBCF to obtain better performance. In Sect. 3, we present our problem setting through an analysis of real-world datasets. In Sect. 4, we explain the motivation and detailed procedures of our improved CBCF and then make comparisons between the previous CBCF and improved CBCF. In Sect. 5, we describe our experiments and compare the results of our improved CBCF with other existing work. Finally, in Sect. 6, we draw conclusions and offer suggestions for future work.

2 Related work

2.1 New user cold-start problem in the traditional UBCF

UBCF was first proposed by Herlocker et al. [12], which is one of the most successful approaches popularly used in RSs. The traditional UBCF involves the following four procedures: similarity computation between users, neighborhood selection, rating prediction, and item recommendation [28]. Similarity between a pair of users is computed based on ratings of the co-rated items, which makes UBCF perform poorly on new users' recommendations, because a new user often has insufficient number of co-rated items in a given rating data. Currently, many approaches have been present to solve new users' personalized recommendation. Generally, these researches can be classified into two lines. In the first line, researches employ information from other sources, such as demographic information or user profiles. Kim et al. [15] utilized collaborative tagging to grasp and filter users' preference for items. Lika et al. [19] incorporated classification methods into traditional UBCF approach and used demographic data to identify users with similar behavior. Niu et al. [22] utilized the side information which was beyond user-item rating matrix from various online recommendation sites to compute the similarity of different users. However, approaches in the first line need the additional information which is often not available or incomplete. In the second line, researches aim to improve similarity or prediction computation methods. Gan and Jiang [9] proposed a network-based collaborative filtering approach to improve the diversity without lowing the accuracy of recommendations; however, the performance of this approach depended on the selected parameter, and the optimal value of parameter was still unknown. Adomavicius and Kwon [1] developed a sophisticated graph-theoretic approach to maximize the diversity of recommendations; however, the more items were selected as candidate items, the more diverse and the less accurate were in recommendations. They also improved the recommendation diversity by re-ranking the candidate items through a new re-rank technology. This approach could provide recommendations with good diversity, but it comes at the expense of accuracy [2]. Said et al. [26] decreased the impact of popular items and increased the impact of unpopular items. This approach could improve the diversity of recommendations for a new user with acceptable accuracy; however, it was unstable and performed differently for different selected datasets.

2.2 The previous CBCF approach

To increase the diversity of recommendations while maintaining comparable levels of recommendation accuracy, we developed in our previous work a CBCF approach to improve the traditional UBCF [34,36]. The rationale of CBCF is the covering reduction algorithm in covering-based rough sets. Rough set theory was first presented by Pawlak in the early 1980s [24]. Covering-based rough sets extend classical rough sets by using a covering of the domain rather than a partition [31], making it easier to use in real applications. Here, we define covering and covering approximation space, and more detailed explanations can be found in the literature [38,40].

Definition 1 Let T be the domain of discourse, and C a family of subsets of T. If none of the subsets in C is empty and $\cup C = T$, then C is called a covering of T.

Definition 2 Let T be a non-empty set, and C a covering of T. We refer to the ordered pair $\langle T, C \rangle$ as the covering approximation space.

Covering reduction is a significant concept in covering-based rough set theory. It can remove redundant elements to form the minimal covering that generates the same covering approximations and thus is also a key concept in reducing redundant information in data mining [39,40]. Different covering reduction algorithms correspond to different types of covering-based rough sets. Since the previous study [36], it has been clear that the algorithm provided by Zhu et al. [40] is the most appropriate means of removing redundant information in real applications. Detailed information regarding this algorithm is provided in Definition 3 and Algorithm 1.

Definition 3 Let C be a covering of domain T, and $K \in C$. If there exists another element K' of C such that $K \subset K'$, then K is a redundant element of covering C. When we remove all redundant elements from C, the set of all remaining elements is still a covering of T, and this new covering has no redundant element. We refer to this new covering as the reduct of C.

Algorithm 1 CRA (covering reduction algorithm)

```
Require:  A covering of a domain: $C$.
Ensure:  An irreducible covering of a domain: $reduct(C)$.
   $K_i$, $K_j$: Elements in the covering $C$.
1: set $reduct(C)$=$C$;
2: for $i = 1$ to $|C|$ do
3:   for $j = 1$ to $|C|$ do
4:     if $K_j \subset K_i$ then
5:       if $K_j \in reduct(C)$ then
6:         $reduct(C) = reduct(C) - \{K_j\}$;
7:       end if
8:     end if
9:   end for
10: end for
11: return $reduct(C)$;
```

Table 1 Proportion of items and ratings in the MovieLens dataset

	Item number	Item rate (%)	Rating number	Rating rate (%)
Ratings $\geq$ 10K	174	1.63	2,757,120	27.57
5K $\leq$ ratings $<$ 10K	296	2.77	2,112,854	21.12
1K $\leq$ ratings $<$ 5K	1564	14.64	3,562,589	35.62
Ratings $\leq$ 1K	8647	80.96	1,569,491	15.69

Table 2 Proportion of items and ratings in the Netflix dataset

	Item number	Item rate (%)	Rating number	Rating rate (%)
Ratings $\geq$ 50K	501	2.82	45,020,066	45.63
10K $\leq$ ratings $<$ 50K	1541	8.67	34,889,199	35.36
1K $\leq$ ratings $<$ 10K	5084	28.61	17,193,080	17.42
Ratings $\leq$ 1K	10,644	59.90	1,569,491	1.59

In previous CBCF, combining with the characteristics of redundant users in UBCF and redundant elements in covering-based rough sets, we inserted a neighbor selection procedure into the traditional UBCF that could remove redundant candidate neighbors by covering reduction algorithm. To remove as many redundant users as possible, according to the sufficient information from an active user, we first extracted relevant attributes of the active user, then constructed decision class by all items that fit the active user's relevant attributes, and reduced the domain from all items to decision class. Because the CBCF approach could select more appropriate users to comprise the neighborhood of an active user, CBCF was able to provide recommendations with satisfactory accuracy and diversity simultaneously for an active user. However, CBCF needs additional information (e.g., item attribute); besides that, its good performance is based on sufficient ratings which new users often do not own.

3 Analysis and problem setting

In this section, first, we analyze two popular datasets that are often used to evaluate RS approaches. Then, in accordance with the analysis result, we discuss the problem setting of this paper.

3.1 Data analysis

Here, we analyze two popular datasets that were collected from the real world and are often used to evaluate RSs. One is the MovieLens dataset [12], obtained from the Web site of the GroupLens lab. This dataset contains 71,567 users, 10,681 movies, and a total of 10,002,054 ratings on a scale of $\{0.5, 1, 1.5, \ldots, 5\}$. Each user has rated at least 20 movies, resulting in a sparsity of 95.81%. The other is the Netflix dataset, obtained from the Netflix Prize Web site (http://www.netflixprize.com). This dataset contains a total of 100 million ratings from 480,189 users over 17,770 movies (98.81% sparsity). The ratings are on a $\{1, 2, 3, 4, 5\}$ scale, and each user has rated a different number of movies. In this paper, considering the

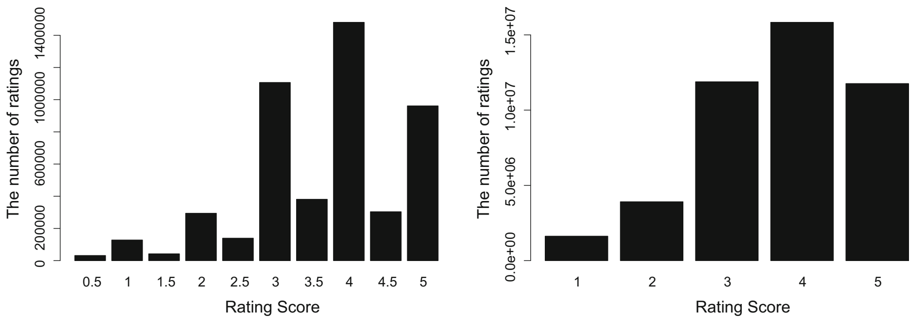


Fig. 1 Proportion of rating scores on popular items in the MovieLens (left) and Netflix (right)

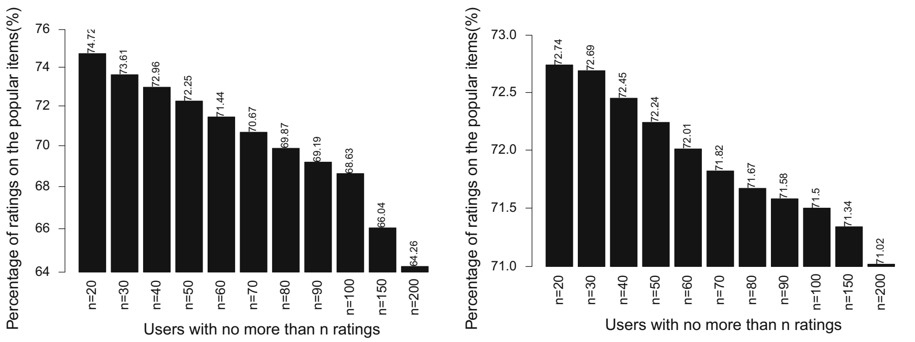


Fig. 2 Percentage of ratings on popular items by users with no more than n ratings in the MovieLens (left) and Netflix (right)

size of datasets and the number of experiments we conducted, we used the full datasets for analysis and smaller subsets for some of the experiments in Sect. 5.

First, we perform statistical analyses for these two datasets. Tables 1 and 2 show the number of items and the corresponding ratings, as well as their proportions according to the different number of ratings. As shown in the tables, for the MovieLens dataset, items that have more than 5K ratings account for only 4.40% of all items, but their corresponding ratings comprise 48.69% of all the ratings. In the Netflix dataset, this performance is more obvious, even though items that have more than 50K ratings comprise only a 2.82% proportion of all items, with ratings corresponding entirely to them accounting for 45.63% of all ratings. From the data analysis above, we can conclude that in real-world database, after sorting all items by descending order according to the number of ratings, the top fewer items usually correspond to a large proportion of the ratings. Therefore, in this paper, we call them popular items.

Next, we consider the proportion of rating scores on popular items. Figure 1 shows the results. As found in the figure, in the MovieLens dataset, the rating range is from 0.5 to 5 with half-star increments, but most of the rating scores are concentrated on $\{3, 4, 5\}$. In the Netflix dataset, although the rating range is just from 1 to 5, most of the rating scores are also included in $\{3, 4, 5\}$. These results indicate that users' rating scores on popular items are relatively concentrated, with the difference between these rating scores not being very large. Because the similarity computation is based on the rating scores of co-rated items, the closer the rating scores, the more similar the two users. Therefore, if most of the co-rated items between two users are popular items, then the rating scores between them will be similar, and they will achieve higher similarity. From the above, we can conclude that if co-rated items between two users concentrate on popular items, the similarity between them will be higher.

Finally, we discuss the percentage of ratings on popular items by users with ratings no more than $\{20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200\}$ in the two datasets. Figure 2 shows the results. As shown in the figure, the two datasets have the same performance in that the percentage of ratings on popular items decreases as the number of ratings by users increases. Users with having no more than 20 ratings have the highest percentage, almost 74.72% in the MovieLens dataset and 72.74% in the Netflix dataset. In general, new users often have fewer ratings (i.e., no more than 20 ratings). Therefore, we can conclude that most ratings of new users concentrate on popular items.

3.2 Problem setting

According to our conclusions obtained above, if co-rated items between two users concentrate on popular items, then they will have higher similarity. Because most ratings of a new user are on popular items, if there are other users whose ratings also concentrate on popular items, the similarity between them will be very high, and these users can easily be selected into the neighborhood of the new user. Therefore, in traditional UBCF, a new user's neighborhood is usually comprised of users whose ratings concentrate on popular items.

However, neighborhoods comprised of users whose ratings concentrate on popular items can make predictions only for fewer types of items, perhaps even only popular items. Hence, in traditional UBCF, candidate items with high predicted scores are the most popular items, resulting in a low diversity of recommendations for a new user. In addition, these users can predict accurate rating scores only for popular items rather than for all types of items. Thus, the accuracy of recommendations for a new user will also be unsatisfactory. Here, we define users whose ratings concentrate on popular items as redundant users of a new user. In the traditional UBCF approach, as the neighborhood of a new user often contains many redundant users, recommendations they produce might concentrate on popular items. A new user's acceptance of these recommendations will substantially increase the percentage of ratings on popular items and further improve the similarity between the new user and redundant users. Under these circumstances, redundant users are easier to select into a neighborhood. Finally, as a consequence, a vicious circle is established, and a new user might be able to obtain only recommendations determined by popular items.

4 Improved CBCF for new users' personalized recommendations

To introduce our improved CBCF approach, we first present some RS-related notations and terminologies. Given an RS, let $U = \{u_1, u_2, \ldots, u_m\}$ and $I = \{i_1, i_2, \ldots, i_n\}$ be finite sets of users and items, respectively, $R \cup \{\star\}$ the set of possible item rating scores, and RM the user-item rating matrix. Absence of a rating is indicated by an asterisk ($\star$). The rating score of user u for item i is denoted by $r_{u,i} \in R \cup \{\star\}$, and the average of the valid ratings of user u is denoted by $\bar{r}_u$. $I_u = \{i \in I | r_{u,i} \neq \star\}$ is the set of all items rated by user u, and I_u^c is the complementary set of I_u, indicating items that have not yet been rated by user u.

4.1 Motivation of improved CBCF

In order to provide recommendations with satisfactory accuracy and diversity simultaneously for a new user, our improved CBCF aims to remove as many redundant users as possible and utilizes the remaining more appropriate users to comprise the neighborhood of a new user.

The target of previous CBCF is to provide satisfactory recommendations for an active user. Because an active user has rated many items, there is sufficient information that could be utilized. Therefore, in the previous CBCF, the decision class consists of items that fit the active user's relevant attributes, and relevant attributes can be obtained from sufficient rating information. However, for a new user, ratings are usually very few, and it is unreliable to extract relevant attributes according to a new user's rating information. Moreover, in the previous CBCF, the item attribute matrix had to be inputted as the indispensable condition, even though some datasets do not have this information. Therefore, for a new user's personalized recommendations, in our improved CBCF approach, we must make full use of the characteristic of a new user (e.g., fewer ratings or ratings concentrating on popular items) and reconstruct the decision class while ensuring as far as possible that the new approach requires no special additional information.

Note that, generally speaking, new user cold-start problems in RSs include two types [30]:

1. Complete new user cold-start where no rating records are available;
2. Incomplete new user cold-start where only a small number of ratings are available.

For complete new user cold-start (the first type), to provide personalized recommendations, some other special additional information (e.g., user profiles or demographic) should be utilized, but they are often unavailable or difficult to obtain. Our improved CBCF only needs few rating information; therefore, the main purpose of our improved CBCF is to provide personalized recommendations for incomplete new users (the second type) by only utilizing the easily available rating information.

4.2 Reconstruction of decision class for a new user

In accordance with the discussion in Sect. 4.1, we reconstruct the decision class for the new user as the set of niche items in the dataset used for recommendation. As we discussed in Sect. 3.1, in real-world database, after sorting all items by descending order according to the number of ratings, the top fewer items correspond to a large population of the ratings, and we called them popular items. We then define niche items as items that are not popular in the dataset.

There are the following three reasons why we reconstruct the decision class for the new user as the set of niche items:

1. Redundant candidate neighbors for a new user are able to be removed as many as possible;
2. The decision class as the set of niche items is easily constructed from the user-item matrix;
3. Computation of the decision class can be performed off-line.

The first reason is that we can remove redundant candidate neighbors for a new user as many as possible. Generally, in candidate neighbors of a new user, items rated by a redundant user u_1 are the most popular items and the set of niche items rated by u_1 is very small or even empty. Under these circumstances, it is very easy to find another user u_2 whose rated niche items' set includes u_1's. In other words, in the entire set of niche items, u_2 can not only make predictions for items as u_1 does, but also predict ratings for other types of niche items.

Algorithm 2 Decision class construction algorithm for new user

```
Input: User-item rating matrix $RM$, ratio threshold $rt$
Output: Decision class $D_{nu}$ for new user
1: for all $i \in I$ do
2:     $n_i \leftarrow$ Count the number of users $u \in U$ such that $r_{u,i} \neq \star$.
3: end for
4: $D_{nu} = I$.
5: while $\frac{|D_{nu}|}{|I|} \geq rt$ do
6:     $i \leftarrow$ Select an item with highest value $n_i$ in $D_{nu}$.
7:     $D_{nu} \leftarrow D_{nu} \setminus \{i\}$.
8: end while
9: return $D_{nu}$
```

Therefore, u_2 might be more appropriate for being selected into the neighborhood than u_1, even though the similarity of u_1 might be a little higher. When reducing the domain from item set I to the decision class comprised by the set of niche items, some redundant users who have not rated niche items will be removed first. Then because a redundant element in a covering is also included in other elements, which has the same characteristics as the set of items rated by redundant users, we can utilize covering reduction to remove redundant users. Even using the covering reduction algorithm cannot remove all the redundant users, but it can remove most of them, and our experiments in Sect. 5 confirm this.

The second reason is that, by utilizing the set of niche items, decision class can be constructed easily without requiring any other special additional information. It is because the niche items could be extracted easily from the user-item rating matrix which can be obtained from almost all types of RSs.

The third reason is that the decision class can be constructed off-line by the set of niche items, because computation of the decision class is independent from any user. In this way, the user reduction procedure can also be computed off-line. Therefore, our improved CBCF has better computational complexity than the traditional UBCF and the previous CBCF, and we also discuss this in Sect. 4.3.

Algorithm 2 constructs a decision class D_{nu} for new user from the user-item rating matrix RM and the ratio threshold rt $(0 < rt < 1)$. In this algorithm, the set of popular items is regarded as the top $(1 - rt) \times 100\%$ items which have the largest number of ratings in I and the decision class D_{nu} for new user is constructed by removing popular items from the set of all items I.

4.3 Improved CBCF approach and computational complexity analysis

Algorithm 3 shows detailed procedures of the improved CBCF. Comparing with the previous CBCF, our improved CBCF has three main differences:

1. The target user is different. The previous CBCF aims to provide satisfactory recommendations for an active user who often has sufficient ratings, but it cannot perform well on a new user who usually has insufficient rating information. However, our improved CBCF can only utilize few rating information to provide personalized recommendations for a new user with satisfactory accuracy and diversity simultaneously.
2. Input information is different. To generate satisfactory recommendations, the previous CBCF needs both user-item rating matrix and item attribute matrix, but item attribute information in some RSs is unavailable. However, our improved CBCF only needs to input the user-item rating matrix rather than any other special information.

3. Decision class is different. In the previous CBCF, decision class is constructed by all items that fit active users' relevant attributes, but relevant attributes should be extracted through sufficient rating information. However, our improved CBCF constructs the decision class by niche items which could be extracted easily from the user-item rating matrix.

Algorithm 3 Improved CBCF approach

```
Input:  User-item rating matrix $RM$ and a new user $nu$.
Output:  The top $N$ recommended items for the new user $nu$.
    $k$ : Number of users in the neighborhood $N^r_{nu}(k)$ of the new user $nu$.
    $N$ : Number of items recommended to the new user $nu$.
    $D_{nu}$ : Decision class of the new user $nu$.
    $U^r$ : Users after user reduction, reduct-users.
    $I^c_{nu}$ : Items that have not yet been rated by the new user $nu$.
    $CN^r_{nu}$ : Candidate neighbors of the new user $nu$ after user reduction.
    $p_{nu,i}$ : Rating prediction of item $i$ for the new user $nu$.
1: for each user $u \in U$ do
2:     $C_u = I_u \cap D_{nu}$.
3: end for
4: Let $C^* = D_{nu} - \cup C_u$. Then, $C = \{C_1, C_2 ... C_{|U|}, C^*\} - \{\emptyset\}$ is a covering for the new user $nu$ in domain
   $D_{nu}$.
5: $reduct(C) = CRA(C)$.
6: Reduct-user $U^r = \{u \in U | C_u \in reduct(C)\}$.
7: $CN^r_{nu} = U^r$, compute the similarity between the new user $nu$ and each user $u \in CN^r_{nu}$.
8: for each item $i \in I^c_{nu}$ do
9:     Find the $k$ most similar users in $CN^r_{nu}$ to comprise neighborhood $N^r_{nu}(k)$;
10:    Predict rating score $p_{nu,i}$ for item $i$ by neighborhood $N^r_{nu}(k)$.
11: end for
12: Recommend to the new user $nu$ the top $N$ items having the highest $p_{nu,i}$.
```

Next, we analyze the computational complexities of the traditional UBCF, previous CBCF, and improved CBCF according to the number of users m, the number of reduct-users m^r, the number of similar users k, and the number of recommended items N. In all of three approaches, we assume the situation that a new user receives recommendations. This situation consists of the following four steps:

1. Computation of similarity between the new user and other users;
2. Sorting of other users by descending order of the similarity score (this corresponds to select k nearest users);
3. Computation of rating score for each item that the new user has no rating score, and
4. Sorting of items by descending order of the predicted score (this corresponds to select top N items).

In the traditional UBCF, first, the computational complexity of similarity computation is $O(mn)$ because at most n items are searched for computation for each user. Next, the computational complexity of sorting of m users by descending order is $O(m \log m)$ by using some fast sorting algorithm, e.g., quick sort. In general, the number k of similar users is very small relative to the number m of all users, i.e., $k \ll m$, the number k does not affect the computational complexity of step 2. Moreover, to predict the rating score for each item, it is required to search at most k similar users to use their similarity score and rating score, and because the new user has few rating score, rating scores of almost n items are predicted. It concludes that the computational complexity of step 3 is $O(n)$. Note that, similar to the case of step 2, the number k is very small relative to the number n, and k does not affect the computational complexity of step 3. Finally, the computational complexity of sorting of almost n items is $O(n \log n)$. Totally, the computational complexity of recommendation to the new user by the traditional UBCF is $O(mn + m \log m + n + n \log n) \cong O(mn)$.

Then, we consider the computational complexities of recommendation to the new user by the previous CBCF and improved CBCF. The following three processes are used in both previous CBCF and improved CBCF before recommendation: construction of a decision class, construction of a covering in the decision class, and covering reduction. In the previous CBCF, the decision class is extracted from the new user's relevant attributes. Because different users have different relevant attributes, the decision class should be extracted online, which will result in the previous CBCF requiring more computational complexity than the traditional UBCF. However, in our improved CBCF, we do not need to consider the computational complexity of these three processes: because making of decision class by Algorithm 2, making of a covering of the decision class (steps 1–4 in Algorithm 3), and covering reduction by Algorithm 1 are independent from any user and are able to compute off-line before using RS. Hence, for our improved CBCF, similar to the case of the traditional UBCF, we concentrate on the computational complexity of the above 4 steps.

To consider the computational complexity of recommendation by the improved CBCF, we can use the similar discussion for the case of the traditional UBCF. The main difference in this discussion is the number m^r of the reduct-users, and we obtain that the computational complexity of recommendation by the improved CBCF is $O(m^r n)$. In general, the number m^r is smaller than m, and practically, the improved CBCF has better computational complexity than the traditional UBCF based on using off-line computation of the decision class, covering, and covering reduction.

5 Experiments and evaluations

In this section, we introduce the evaluation dataset and metrics, and compare the performance of the improved CBCF approach with other work using different datasets.

5.1 Experimental setup and evaluation metrics

In our experiments, we used the MovieLens and Netflix datasets to evaluate our improved CBCF approach. We also used the Jester dataset [10], obtained from the online joke Web site http://www.ieor.berkeley.edu/~goldberg/jester-data, because it has characteristics different from the former two datasets. This dataset contains ratings of 100 jokes from 24,983 users (27.53% sparsity). Each user has rated 36 or more jokes. The value range of rating scores is -10 to 10. A value of 99 represents an absent rating. From the information of these three datasets, we find that MovieLens and Netflix are the same type of dataset, each containing a huge number of items; however, each user has rated fewer items, with the number of rated items being substantially smaller than the number of unrated items. Therefore, the two datasets are very sparse, and popular items can be found easily. In contrast, the Jester dataset contains only 100 items, each user has rated a sufficient number of items relative to all items, and unrated items are fewer than rated ones; hence, this dataset is not sparse. In addition, in the Jester dataset, every item has been rated by many users, with the result that it is difficult to distinguish whether an item is popular. In fact, we can even say that each item is popular.

To evaluate the performance of our improved CBCF approach, we utilized stratified sampling method to extract 1000 users and 1000 items from MovieLens, Netflix, and Jester

Table 3 Experimental items versus original data in the MovieLens dataset

	Experimental items	Original items	Item rate (%)
Ratings $\geq$ 10K	16	174	1.63
5K $\leq$ ratings $<$ 10K	28	296	2.77
1K $\leq$ ratings $<$ 5K	146	1564	14.64
Ratings $\leq$ 1K	810	8647	80.96
	1000	10,681	100

Table 4 Experimental items versus original data in the Netflix dataset

	Experimental items	Original items	Item rate (%)
Ratings $\geq$ 50K	28	501	2.82
10K $\leq$ ratings $<$ 50K	87	1541	8.67
1K $\leq$ ratings $<$ 10K	286	5084	28.61
Ratings $\leq$ 1K	599	10,644	59.90
	1000	17,770	100%

datasets. It is because utilizing subset of original datasets is very popular to evaluate the performance of recommendation approaches [8,18,25]. First, we select 1000 items based on the item and rating structure of the original dataset by stratified sampling (detailed information is given in Tables 3 and 4). Next, because we only select 1000 experimental users from the whole users (71,567 users in the MovieLens dataset, 480,189 users in the Netflix dataset), the sample size is very small relative to the original data. In order to ensure experimental users as similar with the original datasets as possible, we extract users that satisfy the following two conditions as candidate users:

1. The user has at least one rated item in the selected 1000 items;
2. Percentage of popular items in the user's rated items is no less than the minimum value shown in Fig. 2.

For example, in the case of the MovieLens dataset, a user is extracted as a candidate user if the user has at least one item in the selected 1000 items and the percentage of popular items in this user's rated items is no less than 64.26%. Similarly, for the Netflix dataset, the percentage of popular items in a candidate user's rated items is no less than 71.02%. In this way, users who have quite different rating proportion with original datasets will not be extracted, so that every experimental user selected from candidate users could have a rating proportion as closely with the original datasets as possible. Then, we select 200 test users and 800 training users from the candidate users. First, we randomly select 200 users who have ratings numbering no less than five and no more than 25 as the test users, and randomly mask 20% of the ratings in each test user. We regard every test user as a new user, and each new user has at most 20 ratings as training ratings by the masking of ratings and at most 5 ratings as test ratings in our experiments. Finally, 800 users are randomly selected from candidate users as the training users. Note that the selection of the ratio threshold rt directly affects the efficiency of user reduction.

– If the ratio threshold rt is too small, the size of decision class will be too small, and some users' rated items set will include decision class easily. In this case, most of users will be removed as redundant users, so it will lose the meaning of user reduction;

– If the ratio threshold rt is too big, the size of decision class will be too large. In this case, only fewer redundant users can be removed, so it will greatly reduce the efficiency of user reduction.

In this paper, based on statistical results obtained from MovieLens and Netflix in Tables 1 and 2, we have concluded that after sorting all items by descending order according to the number of ratings, the top 5% items correspond to about 50% of the ratings. So in our experiment, we set the ratio threshold $rt = 0.95$; it means the top 5% of items that have the most ratings are treated as popular items, with the remaining 95% items being considered as niche items.

In contrast, for the Jester dataset, because there are only 100 items, we treat the top 50 items that have the most ratings as popular items and the remaining items as niche items in the experiments. Since each user has rated 36 or more jokes in Jester dataset, here we randomly select 200 test users and remove some of their ratings to make them as new users, and 800 users are selected randomly as training users. To avoid the impact of accidental phenomena, we repeat the experiments 20 times for each dataset and compute the average values as our results. After selecting our experimental items and users from original datasets, the average sparsities of selected datasets from MovieLens, Netflix, and Jester are 98.90%, 88.83%, and 36.42%, respectively. Although the sparsities are a little different with original datasets, we can also call selected datasets from the MovieLens and Netflix which are sparse, and selected dataset from the Jester is not sparse.

To measure the performance of the improved CBCF approach, we used the mean absolute error (MAE) and root-mean-square error (RMSE) to represent the predictive accuracy of recommendations. Precision and recall metrics were used to evaluate the classification accuracy of recommendations. In addition, we used coverage, mean personality (MP), and mean novelty (MN) to evaluate the diversity of recommendations. In accordance with Herlocker's research [11], to maintain real-time performance, we selected different sized k neighborhoods from candidate neighbors, $k \in \{20, 25, 30, \ldots, 60\}$. Furthermore, to calculate the precision and recall values, we treated items rated no less than 3 as relevant items, and the number of recommendations was set to 2, 4, 6, 8, 10, and 12.

The MAE and RMSE metrics demonstrate the average error between predictions and real values; therefore, the lower these values, the better the RS accuracy.

$$\mathrm{MAE} = \frac{1}{|U|} \sum_{u \in U} \left(\frac{1}{|O_u|} \sum_{i \in O_u} |p_{u,i} - r_{u,i}| \right), \tag{1}$$

$$\mathrm{RMSE} = \frac{1}{|U|} \sum_{u \in U} \sqrt{\frac{1}{|O_u|} \sum_{i \in O_u} (p_{u,i} - r_{u,i})^2}, \tag{2}$$

where $O_u = \{i \in I | p_{u,i} \neq \star \wedge r_{u,i} \neq \star\}$ indicates the set of items rated by user u that have prediction values.

The precision metric refers to the proportion of relevant recommended items from the total number of recommended items for the new user, and recall metric indicates the proportion of relevant recommended items from all relevant items for the new user. Here, higher values of two metrics indicate better performance. Assume that N_s is the number of recommended item for the new user, and N_r denotes the number of items preferred by the new user, N_{rs} denotes the amount of the new user's relevant items that appear in the recommended list. The precision and recall metrics are defined as follows:

$$\text{Precision} = \frac{N_{rs}}{N_s}, \quad \text{Recall} = \frac{N_{rs}}{N_r}. \tag{3}$$

The coverage metric can be interpreted and defined differently in different research areas. We define the coverage metric as calculating the percentage of situations in which at least one k-nearest neighbor of a new user can rate an item that has not yet been rated by that new user. Here, let $S_{u,i}$ be the set of user u's neighbors who have rated item i, and define $Z_u = \{i \in I | S_{u,i} \neq \emptyset\}$.

$$\text{Coverage} = \frac{1}{|U|} \sum_{u \in U} \left(100 \times \frac{|I_u^c \cap Z_u|}{|I_u^c|} \right). \tag{4}$$

MP indicates the average degree of overlap between every two users' recommendations. For example, for two users u_i and u_j, we count the number of recommendations of the corresponding top N items, $\text{Rec}_i(N)$ and $\text{Rec}_j(N)$ and further normalize this number by the threshold value N to obtain the degree of overlap between two sets of recommendations. It is clear that an approach of higher recommendation diversity will have a larger MP. As discussed by Gan and Jiang [14], we use $N = 20$ in our calculation of this metric.

$$\text{MP}(N) = 1 - \frac{1}{N} \frac{2}{|U|(|U| - 1)} \sum_{1 \leq i < j \leq |U|} |\text{Rec}_i(N) \cap \text{Rec}_j(N)|. \tag{5}$$

MN indicates the novelty of recommendations provided to users. First, it calculates the fraction of users who have ever rated each recommendation and then computes the sum over all recommendations in $\text{Rec}_m(N)$ to obtain the novelty for user U_m. Finally, we calculate the average novelty over all users.

$$\text{MN}(N) = -\frac{1}{|U|} \sum_{1 \leq m \leq |U|} \sum_{n \in \text{Rec}_m(N)} \log_2 f_n, \tag{6}$$

where f_n indicates the fraction of users who rated the nth item. We also set $N = 20$ in the calculation of this metric, and an approach will have a larger MN if it can make newer recommendations.

In addition, the reduction rate is defined as an evaluation metric, which measures the effectiveness of removing redundant users from among all users. The reduction rate is given as follows:

$$\text{ReductionRate} = \frac{1}{|U|} \sum_{u \in U} \frac{|\text{CN}_u - \text{CN}_u^r|}{|\text{CN}_u|}, \tag{7}$$

where CN_u indicates the candidate neighbors of user u and CN_u^r represents user u's candidate neighbors after user reduction.

5.2 Performance of the improved CBCF approach

To show the performance of the improved CBCF, we compared it with traditional UBCF and previous CBCF approaches. Comparisons were also made with the linear collaborative filtering (LINCF) and inverse user frequency collaborative filtering (IUFCF) presented by Said et al. [26]. For convenience, we refer to the improved CBCF approach as ICBCF in the rest of this subsection. In all of the experiments, we used the Pearson correlation coefficient

Table 5 Number of candidate neighbors for the traditional UBCF and ICBCF approaches

	UBCF	ICBCF	Reduction rate
MovieLens	800	331	0.586
Netflix	800	369	0.539
Jester	800	588	0.265

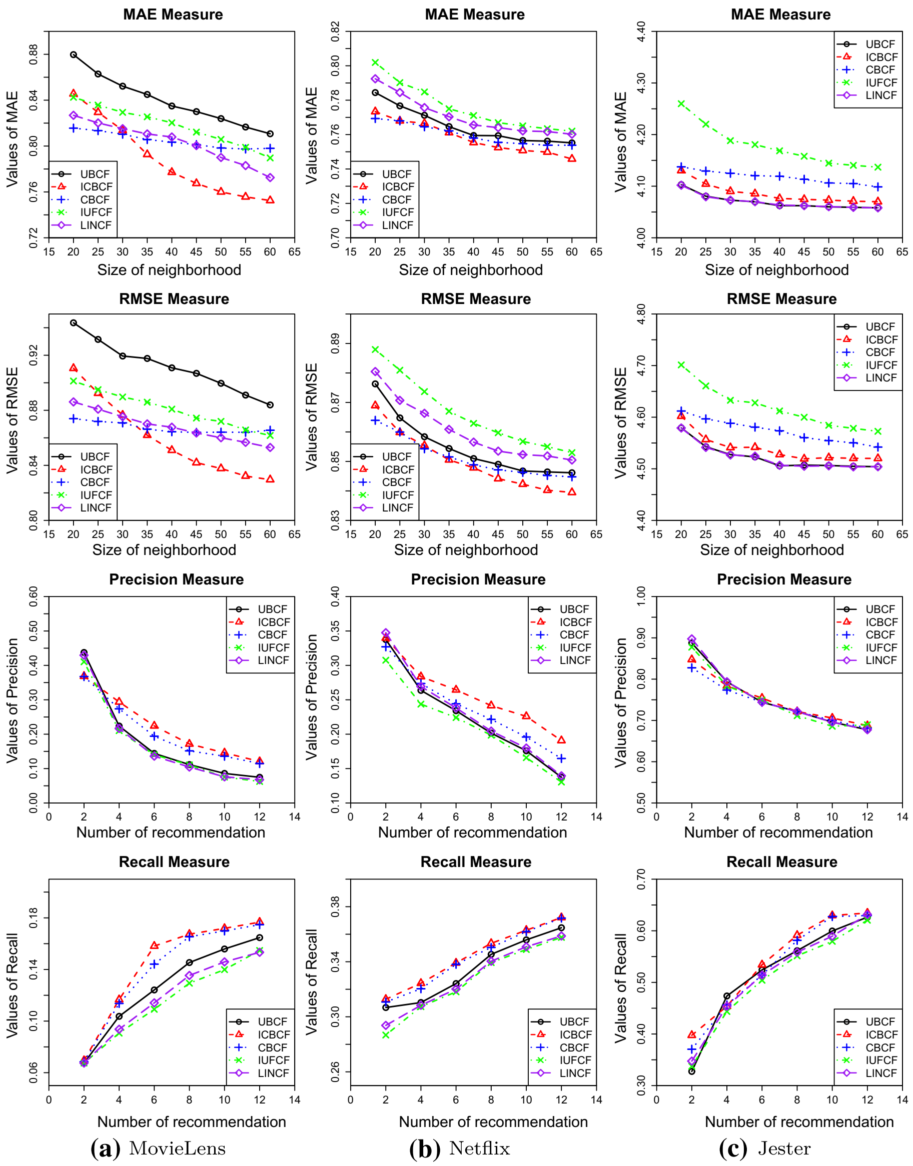


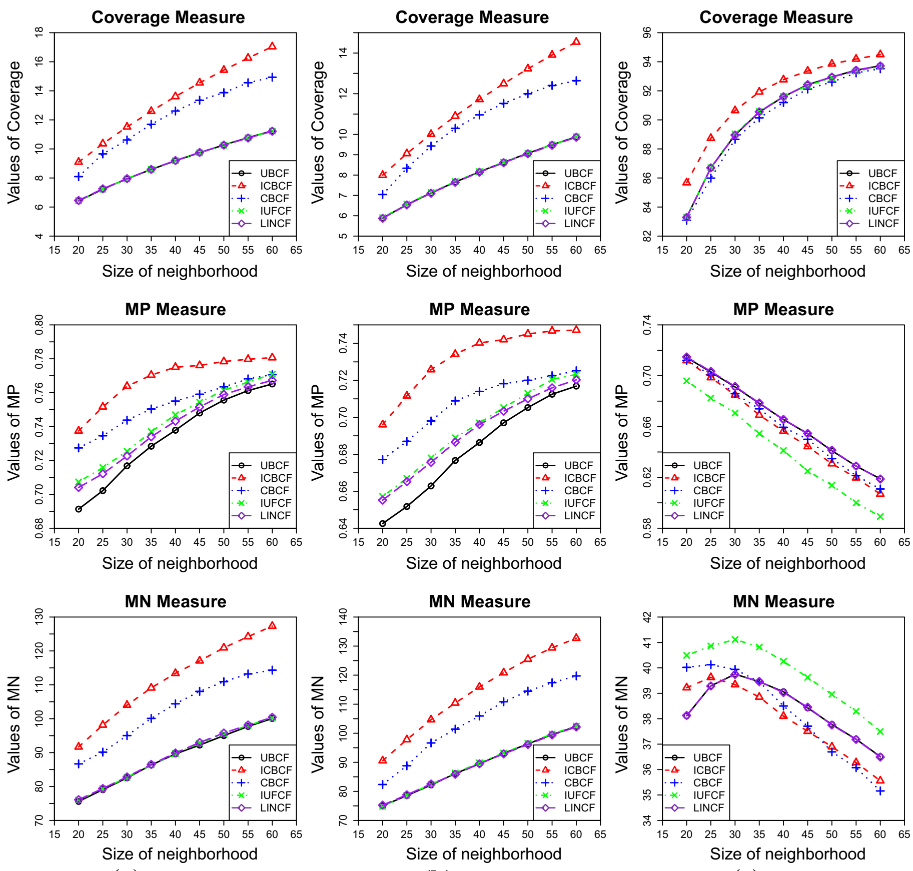
Fig. 3 Results of accuracy measures (MAE, RMSE, precision, and recall) on three datasets

as the similarity measure, and the weighted sum to predict the rating scores. Table 5 shows the results for the number of candidate neighbors for the traditional UBCF and ICBCF approaches on the MovieLens, Netflix, and Jester datasets. As can be seen, in the MovieLens and Netflix datasets, after user reduction, on average, more than half of the users are removed as redundant users. In the Jester dataset, the reduction rate is slightly lower, which means that approximately 26.5% of the users are removed as redundant users on average, with the result that in the ICBCF approach, the average number of candidate neighbors is 588.

Figure 3 shows the results for MAE, RMSE, precision, and recall measures on the MovieLens, Netflix, and Jester datasets, respectively. As shown in the figure, in the MovieLens dataset, both the MAE and RMSE values of the UBCF approach are higher than in the other four approaches. The CBCF has lowest values when the size of neighborhood is no more than 30, but it decreases more slowly than other approaches. Furthermore, although the MAE and RMSE values of ICBCF are higher than those of CBCF, IUFCF, and LINCF in the beginning, ICBCF decreases faster than the other approaches as the neighborhood size increases. In the Netflix dataset, the MAE and RMSE values of UBCF are lower than those of LINCF and IUFCF over all sizes of neighborhood, indicating that the accuracy of UBCF outperforms that of LINCF and IUFCF; however, the values of UBCF are also higher than those of CBCF and ICBCF, demonstrating CBCF and ICBCF have improved the accuracy of traditional UBCF. Comparing CBCF with ICBCF, it can be found that values of CBCF are lower than ICBCF when sizes of neighborhood are 20 and 30, but ICBCF has lower values than CBCF as the size of neighborhood increases. On the other hand, in both MovieLens and Netflix, precision values decrease as the number of recommendations increases; on the contrary, recall values increase when increasing the number of recommendations. Values of precision and recall for CBCF are better than UBCF, IUFCF, and LINCF; however, they are lower than ICBCF because higher values of precision and recall indicate better classification accuracy, indicating that ICBCF outperforms other related approaches in terms of classification accuracy. In contrast, experimental approaches have different performances on the Jester dataset. MAE and RMSE of UBCF have lowest values, indicating that the predictive accuracy of UBCF is highest among all approaches. Although values of CBCF are lower than IUFCF, they are higher than ICBCF obviously. Values of precision and recall have almost same performances as the number of recommendation increases, and thus, ICBCF and CBCF cannot improve the classification accuracy of the traditional UBCF.

Experimental results in Fig. 3 indicate that in sparse datasets (e.g., MovieLens and Netflix), the predictive and classification accuracy of the ICBCF approach outperform that of the other approaches. However, in non-sparse datasets (e.g., Jester), the ICBCF cannot provide recommendations with better predictive and classification accuracy than other related approaches.

Figure 4 shows the results for coverage, MP, and MN on the MovieLens, Netflix, and Jester datasets, respectively. As shown in the figure, the values of the coverage for all approaches increase as the neighborhood size increases. Furthermore, the coverage of ICBCF is significantly higher than that of the other approaches, especially for the MovieLens and Netflix datasets. CBCF has second-best performance in MovieLens and Netflix, but it has lowest coverage values than other approaches over all sizes of neighborhood in Jester dataset. For the MP measure, in both the MovieLens and Netflix datasets, the IUFCF and LINCF approaches have improved the MP of the traditional UBCF slightly. CBCF and ICBCF increase the MP of UBCF clearly, and values of ICBCF are higher than CBCF greatly. In contrast, for the Jester dataset, the MP decreases as the neighborhood size increases, and the MP of traditional UBCF has the highest values. For the MN measure, in both the MovieLens and Netflix datasets, the MN values for UBCF, IUFCF, and LINCF are nearly the same; however, CBCF



(a) MovieLens **(b)** Netflix **(c)** Jester

Fig. 4 Results of diversity measures (coverage, MP, and MN) on three datasets

and ICBCF obviously have higher MN values than the other approaches, showing that CBCF and ICBCF can improve MN significantly, and values of ICBCF are higher than CBCF as the size of neighborhood increases. On the other hand, for the Jester dataset, although the MN values of CBCF and ICBCF are higher than that of UBCF at first, they decrease faster as the neighborhood size increases.

Experimental results in Fig. 4 demonstrate that in sparse datasets (e.g., MovieLens and Netflix), the recommendation diversity of the proposed ICBCF approach outperforms that of the other approaches. However, in non-sparse datasets (e.g., Jester), the ICBCF cannot always obtain better recommendation diversity than other related approaches.

5.3 Analysis and discussion

Our experimental results indicate that the ICBCF approach shows different performances with different datasets. For the MovieLens and Netflix datasets, there are huge numbers of items; however, for each user, the number of rated items is substantially smaller than the number of unrated items. Therefore, the two datasets are very sparse. Under these circumstances, because of users' different behaviors, there might exist some users whose ratings concentrate only on popular items (users whom we have defined as redundant users). As we have confirmed that a new user's ratings also concentrate on popular items, the similarity between redundant users and the new user might be very high, and the neighborhood of the new user might consist almost entirely of redundant users, causing recommendations from these redundant users to concentrate on fewer types of items, perhaps even only popular items. The ICBCF approach used the covering reduction algorithm to remove as many as these redundant users as possible. After user reduction, the neighborhood of a new user in the ICBCF approach consists of users who have rated diverse items, with the result that the diversity of ICBCF greatly outperforms that of the other existing approaches. In addition, because ratings of redundant users concentrate on popular items, they have no ability to make accurate predictions for niche items, resulting in lower recommendation accuracy in the traditional UBCF approach. In the ICBCF approach, redundant users in a neighborhood are mostly removed, with the result that a neighborhood in the ICBCF approach can make predictions for many types of items rather than only popular items, thereby also improving accuracy.

In contrast, for the Jester dataset, the total number of items is 100, and each user has rated at least 36 items. Hence, each user has rated sufficiently many items relative to all 100 items, and each item can be considered as a popular item. Therefore, this dataset is not sparse. In this case, in the ICBCF approach, each user can be considered as a redundant user, with the result that reduction loses its significance. Recommendations from ICBCF might also concentrate on popular items, with the result that the diversity of the ICBCF approach is inferior to those of traditional UBCF. However, because user reduction can select users who have rated more types of items, the coverage of ICBCF is still higher than that of the other approaches. In addition, co-rated items between two users are sufficient, so neighbors with higher similarity can ensure the prediction of more accurate ratings; however, some neighbors with higher similarity might be considered as redundant users to be removed, with the result that the accuracy does not improve but decreases.

Generally, in practical applications, RSs must handle large data that include huge numbers of users and items. Thus, for each user, only a small number of items have been rated compared with the huge number of unrated items. Therefore, most RSs have sparse datasets, such as the MovieLens and Netflix datasets. However, for a sparse dataset, the ICBCF approach can remove redundant users to create more appropriate neighborhoods than the UBCF approach and provide recommendations for a new user with more satisfactory accuracy and diversity values than in existing work. Thus, the ICBCF approach is significant for RSs.

6 Conclusions and future work

In this paper, we have improved CBCF to achieve personalized recommendations for a new user. The improved CBCF approach reconstructs the decision class to account for the set of niche items and uses covering reduction in covering-based rough sets to remove redundant users from candidate neighbors. By removing redundant users who have high similarity with the new user but can make predictions for only a few types of items, improved CBCF makes great improvements in both the accuracy and diversity metrics while utilizing only the user-item rating matrix with no other special information. Our experiments also show superiority of our approach by comparing it with traditional UBCF and other existing work. Although the improved CBCF is inferior to the traditional UBCF in non-sparse datasets (e.g., the Jester dataset), it greatly outperforms other relevant work in sparse datasets (e.g., the MovieLens and Netflix datasets), which occur more often in the real world. Therefore, our approach could be applied to provide satisfactory recommendations for a new user in real-world RSs.

Because the CBCF belongs to CF approaches, so although the CBCF approach is proposed to focus on improving UBCF, the principles of our approach can also be incorporated into item-based collaborative filtering; however, how to define the redundant items requires further consideration. On the other hand, the CBCF approach aims to improve the diversity of recommendations for both new and active users, with the result that we can summarize it to propose a CBCF framework for RSs. Furthermore, our improved CBCF aims to solve incomplete new user cold-start problem where rating information is provided, but cannot work on the complete new user cold-start problem where no ratings are available. We will pursue these goals in our future work.

References

1. Adomavicius G, Kwon YO (2011) Maximizing aggregate recommendation diversity: a graph-theoretic approach. In: Proceedings of the 1st international workshop on novelty and diversity in recommender systems. ACM, pp 3–10
2. Adomavicius G, Kwon YO (2012) Improving aggregate recommendation diversity using ranking-based techniques. IEEE Trans Knowl Data Eng 24(5):896–911
3. Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17:734–749
4. Bobadilla J, Ortega F (2013) Recommender system survey. Knowl Based Syst 46:109–132
5. Bobadilla J, Ortega F, Hernando A, Bernal J (2012) A collaborative filtering approach to mitigate the new user cold start problem. Knowl Based Syst 26:225–238
6. Boim R, Milo T, Novgorodov S (2011) Diversification and refinement in collaborative filtering recommender. In: Proceedings of the 20th international conference on knowledge management. ACM, pp 739–744
7. Clarke C, Kolla M, Cormack G, Vechtomova O, Ashkan A, Büttcher S, MacKinnon I (2008) Novelty and diversity in information retrieval evaluation. In: Proceedings of the 31st international ACM SIGIR conference on research and development in information retrieval. ACM, pp 659–666
8. Formoso V, Fernandez D, Cacheda F, Carneito V (2013) Using profile expansion techniques to alleviate the new user problem. Inf Process Manag 49(3):659–672
9. Gan MX, Jiang R (2013) Constructing a user similarity network to remove adverse influence of popular objects for personalized recommendations. Expert Syst Appl 40:4044–4053
10. Goldberg K, Roeder T, Gupta D, Perkins C (2001) Eigenstate: a constant time collaborative filtering algorithm. Inf Retrieval 4(2):133–151
11. Herlocker JL, Konstan JA (2002) An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Inf Retrieval 5:287–310
12. Herlocker JL, Konstan JA, Borchers A, Riedl J (1999) An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd annual symposium on research and development in information retrieval. ACM, pp 230–237
13. Hu R, Pu P (2011) Helping users perceive recommendation diversity. In: Proceedings of the 1st international workshop on novelty and diversity in recommender system. ACM, pp 43–50
14. Javari A, Jalili M (2015) A probabilistic model to resolve diversity–accuracy challenge of recommendation systems. Knowl Inf Syst 44:609–627

15. Kim HN, Ji AT, Ha I, Jo GS (2010) Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation. Electron Commer Res Appl 9:73–83
16. Kotkov D, Wang S, Veijalainen J (2016) A survey of serendipity in recommender systems. Knowl Based Syst 111:180–192
17. Kunaver M, Požrl T (2017) Diversity in recommender systems: a survey. Knowl Based Syst 123:154–162
18. Li DS, Chen C, Lv Q, Shang L, Zhao YY, Lu T, Gu N (2016) An algorithm for efficient privacy-preserving item-based collaborative filtering. Future Gener Comput Syst 55:311–320
19. Lika B, Kolomvatsos K, Hadjiefthymiades S (2014) Facing the cold start problem in recommender system. Expert Syst Appl 41:2065–2073
20. Liu J, Shi K, Guo Q (2012) Solving the accuracy–diversity dilemma via directed random walks. Phys Rev E 85:016118
21. Niemann K, Wolpers M (2013) A new collaborative filtering approach for increasing the aggregate diversity of recommender systems. In: Proceedings of the 19th symposium on knowledge discovery and data mining. ACM, pp 955–963
22. Niu J, Wang L, Liu X, Yu S (2016) FUIR: fusing user and item information to deal with data sparsity by using side information in recommendation systems. J Netw Comput Appl 70:41–50
23. Noia TD, Ostuni VC, Rosati J, Tomeo P, Sciascio ED (2014) An analysis of users' propensity toward diversity in recommendations. In: Proceedings of the 8th international conference on recommender system. ACM, pp 285–288
24. Pawlak Z (1982) Rough sets. Int J Comput Inform Sci 11:341–356
25. Patra BK, Launonen R, Ollikainen V, Nandi S (2015) A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data. Knowl Based Syst 82:163–177
26. Said A, Jain BJ, Albayrak S (2012) Analyzing weighting schemes in collaborative filtering: cold start, post cold start and power users. In: Proceedings of the 27th annual symposium on applied computing. ACM, pp 2035–2040
27. Son L (2016) Dealing with the new user cold-start problem in recommender systems: a comparative review. Inf Syst 58:87–104
28. Symeonidis P, Nanopoulos A, Papadopoulos AN, Manolopoulos Y (2008) Collaborative recommender systems: combining effectiveness and efficiency. Expert Syst Appl 34:2995–3013
29. Vargas S (2011) New approaches to diversity and novelty in recommender systems. In: Proceedings of the 4th BCS-IRSG conference on future directions information access. British Computer Society, pp 8–13
30. Wei J, He J, Chen K et al (2017) Collaborative filtering and deep learning based recommendation system for cold start items. Expert Syst Appl 69:29–39
31. Zakowski W (1983) Approximations in the space (u, π). Demonstr Math 16:761–769
32. Zhang Z, Kudo Y, Murai T, Ren Y (2019) Addressing complete new item cold-start recommendation: a niche item-based collaborative filtering via interrelationship mining. Appl Sci Basel 9:1894
33. Zhang Z, Dong M, Ota K, Kudo Y (2020) Alleviating new user cold-start in user-based collaborative filtering via bipartite network. IEEE Trans Comput Soc Syst. https://doi.org/10.1109/TCSS.2020.2971942
34. Zhang Z, Kudo Y, Murai T (2015) Applying covering-based rough set theory to user-based collaborative filtering to enhance the quality of recommendations. In: Proceedings of the 4th symposium on integrated uncertainty in knowledge modeling and decision making. Springer, pp 279–289
35. Zhang Z, Kudo Y, Murai T, Ren Y (2019) Enhancing recommendation accuracy of item-based collaborative filtering via item-variance weighting. Appl Sci Basel 9:1928
36. Zhang Z, Kudo Y, Murai T (2017) Neighbor selection for user-based collaborative filtering using covering-based rough sets. Ann Oper Res 256:359–374
37. Zhou T, Kuscsik Z, Liu J, Medo M, Wakeling JR, Zhang Y (2010) Solving the apparent diversity–accuracy dilemma of recommender systems. Proc Natl Acad Sci USA 107:4511–4515
38. Zhu W (2009) Relationship among basic concepts in covering-based rough sets. Inf Sci 179:2478–2486
39. Zhu W, Wang F (2003) Reduction and maximization of covering generalized rough sets. Inf Sci 152:217–230
40. Zhu W, Wang F (2007) On three types of covering-based rough sets. IEEE Trans Knowl Data Eng 19:1131–1144

Zhipeng Zhang received his Ph.D. in the Department of Information and Electronic Engineering from Muroran Institute of Technology, Japan in 2017. He is currently a Lecture in the School of Computer and Information Technology at the Liaoning Normal University, China. From April 2013 to March 2014, he was an exchange student sponsored by the government at Muroran Institute of Technology, Japan. Dr. Zhang serves as a PC member for ISKE2019, as well as a reviewer for Knowledge-based Systems, Knowledge and Information Systems, and Information Retrieval Journal. His research interests include Recommender systems, data mining, and artificial intelligence.

Yasuo Kudo received the Ph.D. degree in systems and information engineering from Hokkaido University, Sapporo, Hokkaido, Japan, in 2000. He is currently a Professor with the College of Information and Systems, Muroran Institute of Technology, Muroran, Hokkaido, Japan. His research interests include rough set theory, non-classical logic, data mining, and recommender systems.

Tetsuya Murai received the Ph.D. degree in information engineering from Hokkaido University, Sapporo, Hokkaido, Japan, in 1995. He is currently a Professor with Chitose Institute of Science and Technology, Chitose, Hokkaido, Japan. His research interests include rough set theory, granular computing, and modal logic.

Yonggong Ren received his Ph.D. in the College of Computer Science and Engineering from Northeastern University, China, in 2006 and his B.S. and M.S. degrees in Computer Science and Engineering from Liaoning Normal University, China, in 1995 and 2003, respectively. He is currently a Professor and Dean in the School of Computer and Information Technology at the Liaoning Normal University, China. From September 2006 to August 2007, he was a visiting scholar at the University of Oslo, Norway, supported by the Chinese government. His research interests include database, data mining, and artificial intelligence.