World Wide Web (2020) 23:1319–1340
https://doi.org/10.1007/s11280-019-00693-x

ICFR: An effective incremental collaborative filtering based recommendation architecture for personalized websites

Yayuan Tang[1,2] · Kehua Guo[2] · Ruifang Zhang[2] · Tao Xu[2] · Jianhua Ma[3] · Tao Chi[4]

Received: 10 August 2018 / Revised: 4 April 2019 / Accepted: 6 May 2019
Published online: 21 May 2019

Abstract
To solve the problem that users' retrieval intentions are seldom considered by personalized websites, we propose an improved incremental collaborative filtering (CF)-based recommendation implementation method (ICFR) in this paper. The ICFR model uses one of the most popular recommendation algorithms – the collaborative filtering recommendation algorithm – for personalized websites. This paper first uses a CF algorithm to obtain the relationship between user preferences and recommended content. Second, the browsing behaviour information of users is extracted by analysing Web logs and is then converted into ratings. Finally, an incremental algorithm is designed to update historical user preference data. Based on this established model, we propose some cases for this architecture, which illustrate that the ICFR model is suitable for personalized website recommendations.

Keywords Recommendation system · User-based collaborative filtering · Personalized website · User preference · Incremental updating

This article belongs to the Topical Collection: *Special Issue on Smart Computing and Cyber Technology for Cyberization*
Guest Editors: Xiaokang Zhou, Flavia C. Delicato, Kevin Wang, and Runhe Huang

✉ Kehua Guo
guokehua@csu.edu.cn

[1] School of Electronics and Information Engineering, Hunan University of Science and Engineering, Yongzhou, China

[2] School of Computer Science and Engineering, Central South University, Changsha, China

[3] Faculty of Computer and Information Sciences, Hosei University, Tokyo, Japan

[4] Key Laboratory of Fisheries Information, Ministry of Agriculture, Shanghai Ocean University, Shanghai, China

1 Introduction

Recommendation systems have been an important research topic in the last twenty years [6] with its applications including tourism [23], virtual commodities [29], and friend analysis in social media [25, 34]. Large companies such as Amazon and Netflix have successfully employed recommendation systems in their websites [6]. However, a great majority of Internet sites are personalized websites (e.g., university websites, government websites and small and medium enterprise websites (SME websites)). Due to some restrictive factors, these personal websites pay more attention to the construction of the website content but ignore users' retrieval intentions, which results in poorer user experiences compared to those on large websites. Therefore, how to properly apply recommendation systems into various personalized websites to improve the quality of their results is a key point in the research on personalized website optimization.

At present, the research on recommendation systems is quite developed, and increasingly more recommendation methods have been proposed. Ordinarily, recommendation systems are divided into three categories [22, 23]: content-based recommendations [2, 8], knowledge-based recommendations [8] and collaborative filtering (CF) recommendations [9, 16, 32]. The CF algorithm has important applications in recommendation systems [7, 35]. The algorithm discovers the users' preferences by mining the users' historical behavioural data, divides all the users into groups based on different preferences, and recommends items that neighbour their preferences. CF has three main methods of implementation: user-based collaborative filtering (UBCF), item-based collaborative filtering (IBCF) and model-based collaborative filtering [10]. User-based collaborative filtering systems suggest interesting items to a user based on similar-minded people called neighbours [4]. Item-based collaborative filtering systems use the relationships between different items to indirectly calculate the recommendations for users [26]. Model-based collaborative filtering systems need to establish a summarized data model in advance [3].

Despite there being many studies on recommendation systems, they have not been widely used by personalized websites. One of the reasons is that there are no unified rules for web developers to directly use recommendation systems. That is, much work is required if some personalized websites want to implement predictive recommendations based on users' intent. Meanwhile, much repetitive work among these websites results in various wasted resources. Consequently, we analyse the structural features of personalized websites and propose an effective incremental CF-based recommendation architecture in which each accessible link on the website can be regarded as an item in the CF model. In this case, you can use the model regardless of whether it is for an electronic mall or a music website. In addition, we use the user-based CF algorithm because the number of users may be far less than the number of items and the change is relatively small for personalized websites. Finally, we provide an incremental updating method in the user-based CF recommendation system.

Compared with the traditional recommendation systems based on the CF algorithm, the ICFR model has some advantages. (a) It was a wider scope of application. The system redefines the elements (items and ratings) of the CF algorithm, thereby eliminating its limitation to situations where users are asked to explicitly rate items. (b) The incremental updating algorithm reduces the computational burden. The system can automatically update the user's rating data according to the user's browsing behaviour after each time the user logs out of the system or ends the current session. (c) The key functions of this system are modularized. The developers do not need to code many changes or write new code if they want to implement the ICFR model.

The remainder of this paper is organized as follows. Section 2 gives a simple review of the related work. Section 3 describes the system model, client information acquisition,

recommendation algorithm and incremental updating computations. Section 4 provides and evaluates the experimental results. Section 5 presents the conclusions and describes future work.

2 Related works

The recommendation algorithm based on user-based CF is one of the most concise and praiseworthy methods [13]. In recent years, the user-based CF recommendation algorithm has been applied in many recommendation systems [21], and there are many studies that aimed to improve the CF recommendation algorithm. Reference [17] develops a user-based tourist attraction recommendation system. In it, the CF algorithm is effectively applied to tourism. Reference [20] introduces a user-based collaborative filtering technology of an e-commerce personalized recommendation system into a logistics platform to better match service and customers. In addition, there are many other specific applications of the user-based CF algorithm. Some studies have also been carried out to improve the shortcomings of the CF algorithm. Reference [33] implements a user-based CF algorithm on a cloud computing platform to overcome the scalability problem of the CF algorithm. Reference [28] introduces Apache Mahout to construct a customized recommendation system from a selection of machine learning algorithms and focuses on addressing the challenges in the CF, such as scalability and data sparsity. In addition, the recommended algorithm is also used to optimize the energy efficiency of wireless sensor networks [30, 31].

The main objective of CF is to use people with similar preferences to recommend information [6]. The kernel of the CF algorithm is computing the similarity [1, 17]. Several similarity measures have been proposed in the literature with the intention of identifying users with similar inclinations [1], and there are three traditional methods of computing the similarity that are commonly applied in CF recommendation systems: the Pearson correlation coefficient (PCC) [5], the cosine similarity [17] and the modified cosine similarity [12]. Among these, the PCC is the most widely used, and it serves as a benchmark for CF [33]. Due to some defects in the PPC, a number of researchers have tried to overcome its weaknesses and proposed many improved schemes. Reference [27] presents the constrained Pearson correlation coefficient (CPCC) that considers the impact of positive and negative ratings. Reference [14] proposes the weighted Pearson correlation coefficient (WPCC). The researchers increase the influence of items with high variances in ratings and decrease the influence of items with low variances by incorporating a variance weight term. The sigmoid function-based Pearson correlation coefficient (SPCC), which only considers items with positive correlation, is proposed in reference [15].

The majority of the computations in a CF recommendation system are from the similarity computing process. The traditional CF algorithms are best when using static off-line settings [31]. Because the data are constantly changing and updating, the system needs to recalculate the similarity between the current user and the other users if it intends to obtain the optimal similar neighbours, which is quite a waste of time. Thus, reference [11] designs incremental and parallel versions of the co-clustering algorithm and uses them to build an efficient real-time CF framework. Their empirical evaluation demonstrates that this approach is accurate and it has much lower computational costs. Reference [24] proposes a method for addressing the scalability problem based on incremental updates of the user-to-user similarities, which is not based on any approximation method and has the potential to provide high-quality recommendations. Additionally, reference [19] proposes an evolutionary co-clustering method that improves the predictive performance, while maintaining the scalability of the co-clustering in the online phase.

3 System model and problem description

The section presents the system model for generating suitable recommendation results from the server and the incremental updating method for the ICFR model. There are three main elements in the user-based CF algorithm: (a) users, (b) items and (c) ratings [18]. In our method, the user rating is replaced by normalized user behavioural data rather than a direct rating of the items by users. We assume that the server has already collected some rating data and recorded them in a matrix, which guarantees that the server can return better recommendation results for fewer test users.

3.1 System model

Figure 1 shows the structure of the architecture and work processes of the server that is recommending suitable item lists to the client and incrementally updating the data according to users' access behaviour. The process mainly consists of 3 parts: (1) user behavioural information acquisition and normalization, (2) the item recommendation process, (3) and the incremental updating computations.

In the ICFR, we first fetch the client information (e.g., account info, client physical address, and additional requirements) from the client to identify the current user. In addition, the user's browsing behaviours (e.g., user browsing time and favourites) are recorded in the Web logs to indirectly measure the user's preferences on the Web page and to build a user-item matrix. (Figure 1(a)). Then, the server starts the recommendation process that is implemented by the CF algorithm. We can get a user-user similarity matrix from the user-item matrix by computing the similarity. The current user's preferences can be predicted based on those of other users who are similar to the current user according to the K-nearest Neighbour algorithm. As a result, we return the recommendation results that are probably close to the user's preferences to the client. (Figure 1(b)). When the current user logs out of the website or closes the current session with the server, the system will analyse the Web logs and normalize the extracted data as user ratings for items. Then, the incremental updating algorithm updates the history data. (Figure 1(c)).

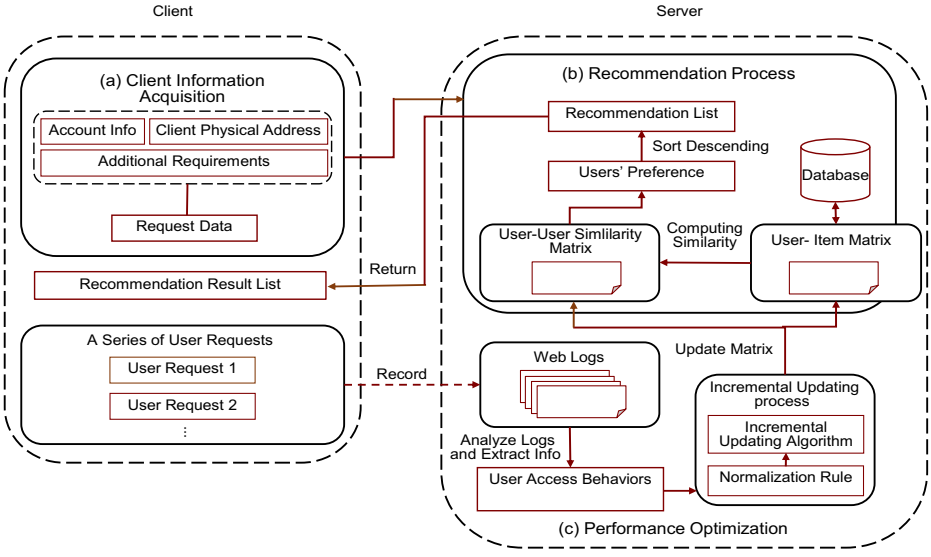


Figure 1 Architecture and work process of the ICFR

In CF algorithm, data incremental updating computation is a difficult problem because the amount of calculation is large and the computing process is complex. For the sake of not affecting the real-time performance of the system and in consideration of the relative stability of user preference in the short term, we analyze and normalize user access behaviour information and update data after user logouts or the current session is closed.

3.2 Client information acquisition

The ICFR architecture mainly adopts the CF mechanism to predict user preferences. In the CF algorithm, users, items and ratings are the three key elements of the user-item matrix. Traditionally, we simply regard one certain thing that is easy to distinguish as the "item" and the "rating" is directly obtained from user ratings. For example, users need to tell the system whether they are interested in or prefer an item by scoring the item. However, in the ICFR architecture, the "item" is more broadly defined, and we use implicit ratings to measure the degree to which users are interested in items.

Universally, the definition of the "item" element in the CF algorithm is atomic and relatively pure. For example, the item can represent all the movies on a video site or all the goods on an E-commerce platform. Hypothetically, on a news website, we regard each piece of news as an item. Under these circumstances, if a user often searches for "Chinese education"-related news on a website, then the recommendation results would contain this kind of news, but it would not include the "education" column for the user. Therefore, we redefine the "item" term in the CF algorithm by defining all accessible links on a website as the "item" element and ignoring whether it links to the homepage, a column or content page.

Figure 2 shows the structure of a common Web page. The column and the page are not at the same level, and a column is not atomic, and it may link to other pages. Nevertheless, we define all of these links as items.

Definition 3.1. Item set (*IS*): Given a website $W_{pw}$with W accessible links, the $IS = \{n_1, n_2, \ldots, n_{|W|}\}$ satisfies the following:

$$n_i = (n\_identity, link) \tag{1}$$

, where:

1) *link* is an accessible link on the website W where $1 \le i \le W$, and
2) $n\_identity$ is the identity of the link *link*.

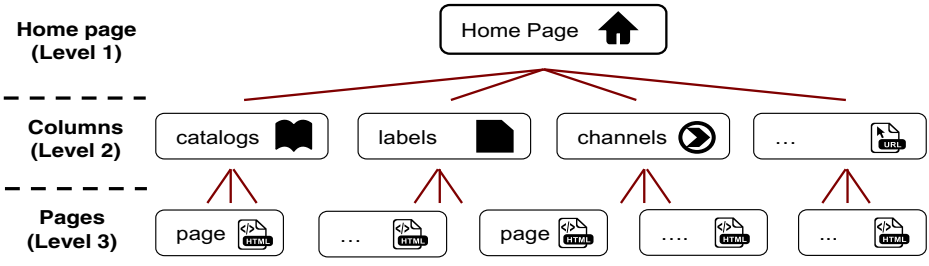


Figure 2 Structure of common Web pages

The acquisition method for the "rating" is another important part of the ICFR architecture, in addition to the definition of items. At present, dominant scoring and implicit scoring are the two main scoring models in the CF algorithm. Considering that most personalized websites such as university websites and government websites do not need user grades for items, and users' behaviour when browsing the websites has great relevance to their interests, we implement implicit scoring by gathering users' behavioural data from web logs and quantifying the data to a certain value as the user ratings for items.

Generally, user's browsing behaviour includes browsing times, the time spent on a page, bookmarks, saves and so on. After extracting users' browsing behaviours from Web logs, we need to quantify them to certain numerical values. Obviously, we can divide the above behavioural information into two kinds: Boolean and numerical values. For instance, if a page's data can be seen as a 0–1 problem and the user browsing times are concrete values, all these data can be normalized to a certain range. In the ICFR model, we can provide previous regular user behaviour lists to the system. We assume that there are N_{ub} kinds of user behaviours that are prestored in a set $BS = \{M_1, M_2, \ldots, M_{N_{ub}}\}$. $\forall M_i \in BS$ and $M_i = (m\_identity, value\_type, b\_value)$, where *identity* is the only behavioural variable, *value _ type* represents the data type (Boolean or numerical) and *b _ value* is an empty value for the moment. In addition, the value of N_{ub} is determined by the developer according to their needs. If N_{ub} is too large, we have to perform extensive analysis and extraction work with the Web logs to build the set *BS*, while enough data can alleviate the sparsity problem of collaborative filtering to some extent. However, the accuracy of the predicted rating can be affected if N_{ub} is too small.

In the ICFR model, the user set *US* is defined as $US = \{u_1, u_2, \ldots, u_N\}$ where $u_i = (u\_identity, client)$ (where *u _ identity* is the unique sign of each user and *client* represents the user account or other symbolic information).

Suppose that a user *U* makes requests to the server over a period of time. Then, we can analyse the Web logs and extract a set of user behaviours from it. Next, we define the user behaviour set.

Definition 3.2. User behaviour set: Given $\Gamma = \{H_1, H_2, \ldots, H_S\}$, a user behaviour set satisfies the following:

$$H_i = \{(u\_identity, n\_identity), BS\}$$

where

1) the *b _ value* of M_i in the set *BS* is extracted from Web logs,
2) *S* represents the number of items that the user has visited, and
3) *u _ identity* and *n _ identity* respectively represent the identity of the user (or the client) and the item.

Each user should provide only one rating for an item. Therefore, the *b _ value*(*s*) in the user behaviour set Γ must be normalized to one value. In the normalization algorithm, we respectively deal with the two data types of user behaviours. The normalization algorithm is presented as Algorithm 1.

Each user should only grade for an item by one rating. So *b _ value*(*s*) in the user behaviour set Γ must be normalized to one value. In the normalization algorithm, we respectively deal with the two data types of user behaviors. The normalization algorithm is presented as Algorithm 1.

Algorithm 1.
Normalization Algorithm

Input: $\Gamma = \{H_1, H_2, \ldots, H_S\}$;

Output: $\gamma = \{T_1, T_2, \ldots, T_S | 1 \le i \le S, T_i = (u\_identity, n\_identity, rating)\}$;

//user rating

for $i = 1$ to S do

$rating = 0$;

for $j = 1$ to N_{ub}do

if $value\_type$ is $Boolean$ then

if b_value is $true$then

$rating = rating + 1$;

end if;

else

$min = \text{Min}(\{M_j.b\_value | M_j \in BS, M_j.value\_type = numerical\ value\})$;

$max = \text{Max}(\{M_j.b\_value | M_j \in BS, M_j.value\_type = numerical\ value\})$;

$X = (M_j.b\_value - min)/(max - min)$;

$rating = rating + X$;

end if;

end for;

$T_i = (u\_identity, i\_identity, rating)$;

$\gamma.\text{add}(T_i)$;

end for;

return γ;

3.3 Recommendation algorithm

UBCF and IBCF are two main algorithms of the CF. They have similar ideas, but they are fit for diverse application sceneries. UBCF and IBCF are somewhat different in the following aspects. (1) Different purposes. IBCF is more applicable to when the number of users is far beyond the number of items, such as on shopping websites. Meanwhile, some websites that frequently update content

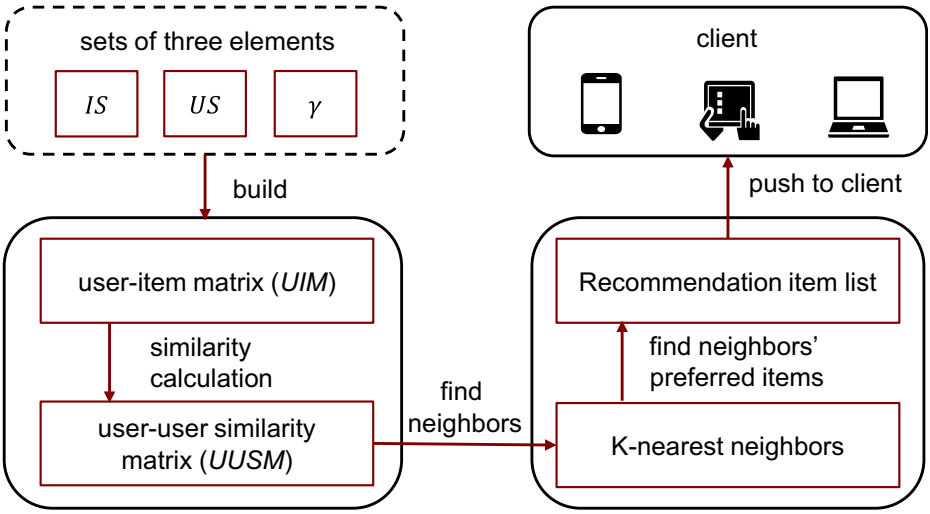


Figure 3 Process of Recommendation in the server

such as social networks are better for UBCF. (2) The diversity of recommendations. From the perspective of the diversity of the recommendations to individual users, UBCF is better than IBCF, but IBCF is more advantageous in a variety of systems. (3) The influence of user characteristics on the algorithm. Whether the UBCF is effective largely depends on the amount and similarity of a user's neighbours. In addition, in IBCF, a user may get more satisfactory results if he/she has high degree of self-similarity. In this work, considering that there are many items and frequently updated content on personalized websites, by combining the differences between UBCF and IBCF, we employ UBCF algorithm in the ICFR model. As a result, the system provides recommended items to a user based on their network neighbours sharing common interests.

The similarity calculation and finding the nearest neighbours are the two main jobs in the UBCF algorithm. In the ICFR, the detailed recommendation process is illustrated as Figure 3. In addition, the required data sets (*IS*, *US* and γ) in the recommendation process are obtained from the former part.

The user-item matrix (*UIM*) $N * W$ is built by *IS*,*US* and γ. In *UIM*, column indicates the item, and row indicates the user. Also, N is the number of users and W is the number of items, r_{ij}represents the rating of the user i for the item j. The detailed structure is shown in Figure 4.

The similarity degree of two users can reflect the common relationships, social circles or preferences between them. Therefore, when two highly similar users are searching a website, they very likely have similar purposes. For example, there are two users, user A and user B, who are highly similar, and their behaviours are predicted by analysing their daily browsing records. In a certain period of time, user A has visited a large number of Web pages about "interviews of Java

Figure 4 Structure of user-item matrix

$$UIM = \begin{Bmatrix} r_{11}r_{12} \cdots r_{1W} \\ r_{21}r_{22} \cdots r_{2W} \\ \cdots \quad \cdots r_{ij} \quad \cdots \\ r_{N1}r_{N2} \cdots r_{NW} \end{Bmatrix}$$

Figure 5 Structure of user-user similarity matrix

$$UUSM = \begin{Bmatrix} S_{11}S_{12}\cdots S_{1N} \\ S_{21}S_{22}\cdots S_{2N} \\ \cdots\ \cdots\ \cdots\ \cdots \\ S_{N1}S_{N2}\cdots S_{NN} \end{Bmatrix}$$

programmers" because he is a Java programmer and looking for a job. To a great extent, user B is in the same situation as user A, and the system is able to automatically recommend the Web pages that user A has browsed. Thus, user B may save much time that would have been spent repeatedly searching for similar information and may eliminate a great deal of repetitive work.

The similarity calculation between users or items is a pivotal step in UBCF. In the ICFR model, the PCC is a suitable method to calculate the similarity between users. However, in the traditional CF recommendation systems using the PCC, they do not consider the importance of co-ratings between users. That is, the similarity between them is not necessarily high according to the PCC value. The situation may exist that the two users have both browsed some similar Web pages, but their preferences for these pages are quite opposite. Under this situation, we may get a relatively high similarity value by using the PCC method, which deviates from the actual situation. To address this problem, this paper uses an advanced PCC method that considers the ratio of the co-ratings to compute the similarity between users.

The traditional PCC formula is defined as follows:

$$sim(u_1,u_2) = \frac{\sum_{n_i\in I_{u_1,u_2}}\left(r_{u_1,n_i}-\overline{r_{u_1}}\right)\cdot\left(r_{u_2,n_i}-\overline{r_{u_2}}\right)}{\sqrt{\sum_{n_i\in I_{u_1,u_2}}\left(r_{u_1,n_i}-\overline{r_{u_1}}\right)^2}\cdot\sqrt{\sum_{n_i\in I_{u_1,u_2}}\left(r_{u_2,n_i}-\overline{r_{u_2}}\right)^2}} \tag{2}$$

Where I_{u_1,u_2} represents the set of items that user u_1 and user u_2 have jointly rated, namely, $I_{u_1,u_2} = \{n_i | n_i \in IS, r_{u_1,n_i}\neq 0, r_{u_2,n_i}\neq 0\}$. $\overline{r_{u_1}} = \frac{1}{|I_{u_1,u_2}|}\sum_{n_i\in I_{u_1,u_2}}\sum_{n_i\in I_{u_1,u_2}} r_{u_1,n_i}$. Based on formula (2), we add the weighting factor *CW* of the common rating into it. The factor *CW* is calculated as $CW = \sqrt{\frac{S_{u_1}^{(u_1,u_2)}}{S(u_1)}}\cdot\sqrt{\frac{S_{u_2}^{(u_1,u_2)}}{S(u_2)}}$. $S(u_i)(i\in[1,N])$ represents the total rating that user u_i assigns tor all Web pages. $S_{u_i}^{(u_i,u_j)}(i,j\in[1,N])$ represents the total rating that user u_i assigns to the items that user u_i and user u_j have jointly rated. The calculation formula is as follows:

$$S(u_i) = \sum_{n_k\in I_{u_i}} r_{u_i,n_k}, S_{u_i}^{(u_i,u_j)} = \sum_{n_k\in I_{u_iu_j}} r_{u_i,n_k} \tag{3}$$

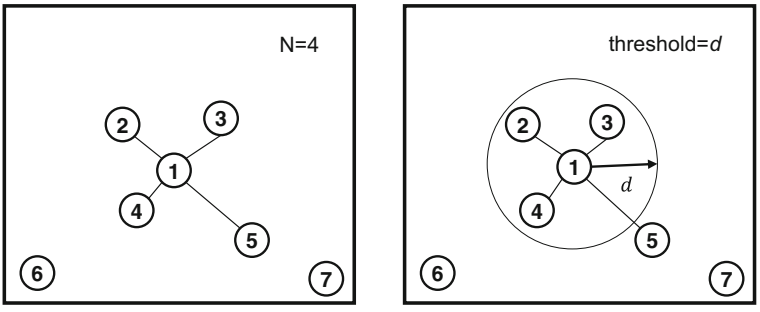


Figure 6 KNN algorithm and threshold-based neighborhood algorithm

Thus, the advanced PPC similarity formula is calculated as follows:

$$sim(u_1, u_2)' = CW \cdot sim(u_1, u_2) \tag{4}$$

The time complexity of the traditional PCC is $O(|I_{u_1,u_2}|)$ because only their co-ratings need to be traversed when computing the similarity between users. The advanced PCC method adds the weighting factor *CW*, and its time complexity is $O(K|I_{u_1,u_2}|)$ (K is constant). Therefore, the two methods have similar time complexity. However, based on the original calculation costs, the improved PCC improves the accuracy of the similarity between users.

After computing the similarity values between all users, we get the user-user similarity matrix (UUSM), which records the similarity values between every two users and is shown in Fig. 5. From the *UUSM*, the current user's nearest neighbors are selected by the nearest neighbour algorithm, which is also a crucial step in UBCF. The K-nearest neighborhoods (KNN) and Threshold-based neighborhoods are two common methods to solve the problem. The schematic diagrams of the above two methods are shown in Figure 6.

The KNN algorithm chooses the K nearest users of the current user as neighbours. The method obtains a fixed number of neighbors for the current user, even though there are not enough similar users to it. The threshold-based neighbourhood algorithm limits the minimum similarity of a neighbor's distance from the current user. While the similarity values between different users vary widely, it is not easy to find a reasonable threshold to ensure the accuracy of the system recommending item results for most users. Combining the characteristics of the two algorithms and considering the probable cold start problem in the ICFR, we employ the KNN algorithm to find the nearest neighbors of the current user. In the algorithm, *NNV* represents the number of neighbours, u_c is the current user and the nearest neighbour set $NNS = \{u_k | u_k \in US,\ 1 \le k \le N\}$ is the output, which is the result representing the nearest neighbors of the current user. The concrete procedure is presented as Algorithm 2.

Algorithm 2.

KNN Algorithm

```
Input: UUSM;NNV;u_c;
Output: NNS = {u_k | u_k ∈ US, 1 ≤ k ≤ N}; //Nearest neighbor set
  cid = u_c.u_identity;
  Sort(UUSM) in ascending order;
  for i = UUSM.size to UUSM.size − NNV do
    if s_cid,i is the element of row cid and column i in the UUSM then
      get u_k from US where i = u_k.u_identity;
   NNS.add(u_k);
    end if;
  end for;
  return NNS;
```

Finally, we give the following formula to predict the blank ratings in the *UUSM*:

$$P(R_{u_c,n_i}) = \frac{S(u_c)}{|I_{u_c}|} + \frac{\sum_{u_\chi \in NNS} sim(u_c, u_\chi)' \cdot \left(r_{u_\chi,n_i} - \frac{S(u_\chi)}{I_{u_\chi}}\right)}{\sum_{u_\chi \in NNS} sim(u_c, u_\chi)'} \tag{5}$$

where $P(R_{u_c,n_i})$ represents the predicted rating that user u_c may provide for item n_i, $S(u_c)$ and $S(u_\chi)$ are defined as in formula (3), and $sim(u_c, u_\chi)'$ is defined as in formula (4). $\frac{S(u_c)}{|I_{u_c}|}$ represents the average rating for user u_c for all items.

For the current user u_c, we have filled the blank ratings for those items that user u_c did not rate. By sorting the set of $P(R_{u_c,n_i})$ values in descending order, the final recommendation item list (*RIL*) is obtained, which is defined as $RIL(u_c) = \{n_i | \forall n_i \in IS, P(R_{u_c,n_i}) \geq P(R_{u_c,n_{i+1}}), r_{u_c,n_i} = 0\}$.

3.4 Incremental updating computation

The scalability is one of great challenges for the CF algorithm. In the ICFR model, the system should maintain two matrices: *UIM* and *UUSM*. Traditionally, the whole matrix *UUSM* would be recalculated if there are changes in the matrix*UIM*. The time complexity of this process is high. Normally, it will produce a series of behaviours when a user visits a web site. It is difficult for the system to update the matrix *UUSM* in real-time according to every user's behaviour. Moreover, a user's preferences are generally stable over a certain period of time. Therefore, we set the incremental updating calculation program to be triggered when the user closes their current session with the server. The flow of the incremental calculation process is shown as Figure 7.

The processes of analyzing Web logs and extracting user access behaviour to compute user's ratings for items have been introduced before. Now we assume that the user u_c has browsed the website $W_{pw}$for a period of time $T_{session}$ and then closed the session with the server. During u_c browsing the website W_{pw}, the access behaviours of u_c have been recorded in the Web logs. So we can compute the ratings of u_c and express them in a set $R_{\tau_{u_c}} = \left(\tau_{u_c,n_1}, \tau_{u_c,n_2}, \ldots, \tau_{u_c,n_{|W|}}\right)$, where τ_{u_c,n_i} represents the normalized rating value of user u_c for item n_i. Based on the $R_{\tau_{u_c}}$, we can update the user-item matrix *UIM*directly and the user-user similarity matrix *UUSM*by the incremental updating computation algorithm. The *UIM* is easy to be updated by new rating set$R_{\tau_{u_c}}$. As to the matrix *UUSM*, we need to compute the new similarity values between user $u_c$and the others $u_\alpha(\forall u_\alpha \in US(c \neq \alpha))$ through the formula (4). The formula (4) is decomposed into four factors as *CW*, $X = \sum_{n_i \in I_{u_c,u_\alpha}} \left(r_{u_c,n_i} - \overline{r_{u_c}}\right)^2$, $Y = \sum_{n_i \in I_{u_c,u_\alpha}} \left(r_{u_\alpha,n_i} - \overline{r_{u_\alpha}}\right)^2$ and $P = \sum_{n_i \in I_{u_c,u_\alpha}} \left(r_{u_c,n_i} - \overline{r_{u_c}}\right) \cdot \left(r_{u_\alpha,n_i} - \overline{r_{u_\alpha}}\right)$. Assume that CW', X', $Y'$and P' respectively represents the updated decomposition factors, and then the updated similarity value $sim(u_c, u_\alpha)'$should be:

$$sim(u_c, u_\alpha)'' = CW' \cdot \frac{P'}{\sqrt{X'} \cdot \sqrt{Y'}} \tag{6}$$

It can be seen from the formula (2) and formula (4) that the element value $s_{i,j}$ in *UUSM*changes with the I_{u_i,u_j} and the I_{u_i}. So we define three sets, which are I_{u_c,u_α},

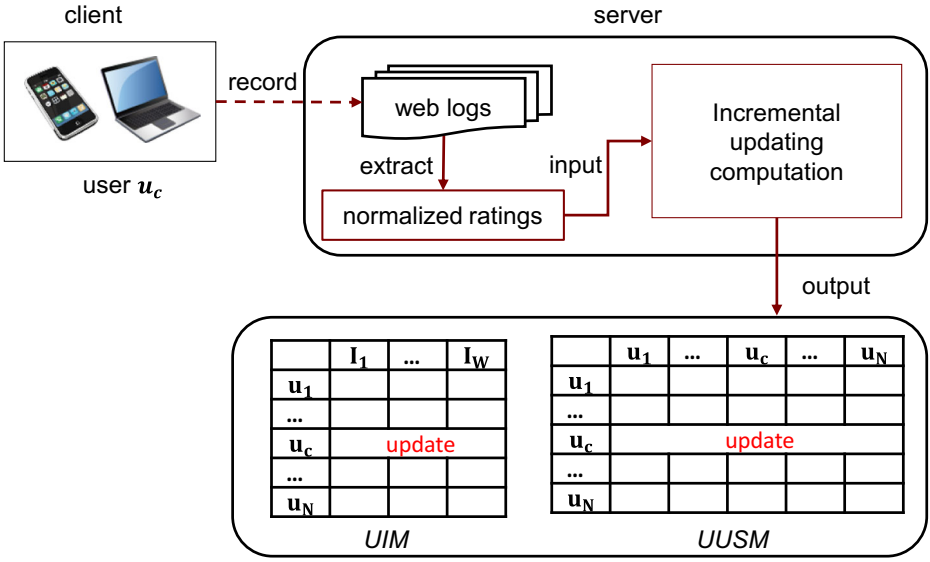


Figure 7 Incremental Updating Calculation Process

$Inc\_I_{u_c}$ (where $\forall n_t \in Inc\_I_{u_c}$, there is $\tau_{u_c,n_t} \neq \varnothing$ and $\tau_{u_c,n_t} \in R_{\tau_{u_c}}$), and $Chg\_I_{u_c,u_\alpha}$ (where $Chg\_I_{u_c,u_\alpha} = Inc\_I_{u_c} \cap I_{u_c,u_\alpha}$) . The change of these three sets is the key factor for incremental updating computation. There are four situations that may happen:

(1) $Inc\_I_{u_c} = \varnothing \text{and} Chg\_I_{u_c,u_\alpha} = \varnothing$

Based on this condition, we can get that there is no change in I_{u_c,u_α} and *UUSM*.

(2) $Inc\_I_{u_c} \neq \varnothing$ and $Chg\_I_{u_c,u_\alpha} = \varnothing$, which indicates that all the rated items in $R_{\tau_{u_c}}$ are not rated by u_c before. Then I_{u_c,u_α} changes into $I_{u_c,u_\alpha}{}' = \left(I_{u_c,u_\alpha} \cup Inc\_I_{u_c}\right) \cap I_{u_\alpha}$, and $\overline{r_{u_c}} \to \overline{r_{u_c}}' = \frac{S_{u_c}^{(u_c,u_\alpha)'}}{|I_{u_c,u_\alpha}{}'|}$, $r_{u_\alpha} \to \overline{r_{u_\alpha}}' = \frac{S_{u_\alpha}^{(u_c,u_\alpha)'}}{|I_{u_c,u_\alpha}{}'|}$. So,

$$\begin{cases} X' = X + \sum\limits_{n_i \in Inc\_I_{u_c}} \left(r_{u_c,n_i} - \overline{r_{u_c}}'\right)^2 \\ Y' = Y + \sum\limits_{n_i \in Inc\_I_{u_c}} \left(r_{u_a,n_i} - \overline{r_{u_a}}'\right)^2 \\ P' = P + \sum\limits_{n_i \in Inc\_I_{u_c}} \left(r_{u_c,n_i} - \overline{r_{u_c}}'\right)^2 \cdot \left(r_{u_a,n_i} - \overline{r_{u_a}}'\right) \\ CW' = \sqrt{\frac{S_{u_c}^{(u_c,u_a)'}}{S(u_c)'}} \cdot \sqrt{\frac{S_{u_a}^{(u_c,u_a)'}}{S(u_a)'}} \end{cases} \tag{7}$$

(3) $Inc\_I_{u_c} = Chg\_I_{u_c,u_\alpha} \neq \varnothing$ which indicates that all the rated items in $R_{\tau_{u_c}}$ have been previously rated by u_c. Then there is no change in I_{u_c,u_α} and $\overline{r_{u_\alpha}}$. $\overline{r_{u_c}} \rightarrow \overline{r_{u_c}}'$. So,

$$\begin{cases} X' = \sum\limits_{n_i \in I_{u_c,u_a}} \left(R_{\tau_{u_c},n_i} - \overline{r_{u_c}}' \right)^2 \\ P' = \sum\limits_{n_i \in I_{u_c,u_a}} \left(R_{\tau_{u_c},n_i} - \overline{r_{u_c}}' \right) \cdot \left(r_{u_a,n_i} - \overline{r_{u_a}} \right) \\ CW' = \sqrt{\frac{S_{u_c}^{(u_c,u_a)'}}{S(u_c)'}} \cdot \sqrt{\frac{S_{u_a}^{(u_c,u_a)}}{S(u_a)}} \end{cases} \tag{8}$$

(4) $Inc\_I_{u_c} \neq \varnothing$, $Chg\_I_{u_c,u_\alpha} \neq \varnothing$ and $Inc\_I_{u_c} \neq Chg\_I_{u_c,u_\alpha}$ which indicates that some of the items in $R_{\tau_{u_c}}$ have been previously rated by u_c, while the others have not. $I_{u_c,u_\alpha} \rightarrow I_{u_c,u_\alpha}{}' = \left(I_{u_c,u_\alpha} \cup \left(Inc\_I_{u_c} - \mathrm{Chg\_I_{u_c,u_\alpha}} \right) \right) \cap \mathrm{I_{u_\alpha}}$, $\overline{r_{u_c}} \rightarrow \overline{r_{u_c}}'$ and $\overline{r_{u_\alpha}} \rightarrow \overline{r_{u_\alpha}}'$. So,

$$\begin{cases} X' = \sum\limits_{n_i \in I_{u_c,u_a}} \left(R_{\tau_{u_c},n_i} - \overline{r_{u_c}}' \right)^2 \\ Y' = Y + \sum\limits_{n_i \in Inc,I_{u_c}} \left(r_{u_a,n_i} - \overline{r_{u_a}}' \right)^2 \\ P' = \sum\limits_{n_i \in Inc,I_{u_c}} \left(r_{u_a,n_i} - \overline{r_{u_a}}' \right)^2 \cdot \left(r_{u_a,n_i} - \overline{r_{u_a}}' \right) \\ CW' = \sqrt{\frac{S_{u_c}^{(u_c,u_a)'}}{S(u_c)'}} \cdot \sqrt{\frac{S_{u_a}^{(u_c,u_a)}}{S(u_a)}} \end{cases} \tag{9}$$

In addition to the above formulas, we need some other extra storage space to store the cache factors, including *X*, *Y*, and *P*. The detailed procedure is shown in Algorithm 3.

Algorithm 3 Incremental Updating Algorithm

Input: u_c;UIM;$R_{\tau_{u_c}}$;I_{u_c}; $Inc\_I_{u_c}$; X;Y;P;

Output: $UUSM$; //Updated $UUSM$

Normalizing($R_{\tau_{u_c}}$) → update matrix UIM;

for a =0 to N do

get I_{u_a} and I_{u_c,u_a} from matrix UIM;

$Chg\_I_{u_c,u_a} = Inc\_I_{u_c} \cap I_{u_c,u_a}$;

if$Inc_I_{u_c} = \emptyset$ and $Chg\_I_{u_c,u_a} = \emptyset$

there is no change in matrix $UUSM$;

else if $Inc\_I_{u_c} \neq \emptyset$ and $Chg\_I_{u_c,u_a} = \emptyset$ then

execute formula (8);

else if $Inc\_I_{u_c} = Chg\_I_{u_c,u_a} \neq \emptyset$ then

execute formula (9);

else

execute formula (10);

end if;

update X, Y, P, and $sim(u_c, u_a)$ in $UUSM$;

end for;

return $UUSM$;

Even though the time complexity of the incremental updating method (or IUM) to compute the similarity values between the current user and the others is O(n), which is nearly the same as the total updating method (or TUM), the method is still advantageous. First, we decrease the computational burden; that is, the system implements one updating process for one user instead of for all users. Second, we divide the incremental updating computations into four cases, which require fewer total computations than the TUM. Thus, in this paper, the incremental updating computations are improved with respect to TUM.

4 Experimental results

4.1 Implementations

Establishing an experimental personalized website is an important aspect of this research. To evaluate the performance, we perform experiments in a real environment.

We implement one situation based on the ICFR model with the server simulating a regular college website. In the simulated personalized website, we acquire nearly 1500 news stories from the real news website of Central South University and register 910 respective users as the item set and the user set of the CF algorithm. The item set contains links from three levels of Web pages, which may be a homepage, a column or a content page. Each link in the item set is regarded as an item.

The experiments are implemented on a common operating platform (2.50 GHz CPU and 8.00 GB RAM). The simulated personalized website is a Java Web project, which is deployed on the Tomcat 7.0.56 software and developed using Eclipse 10. The project runs in JDK 7 or above.

When we apply the ICFR method to the personalized website, there is a cold boot problem, which is one challenge of the CF recommendation algorithm. To alleviate this problem, we randomly generate some access logs for the 910 users and construct the user-item matrix and the user-user similarity matrix. The access Web logs are not entirely random. We divide all the items into 8 themes according to the text information of the links and make each user have a preference for one of these themes. The preference distribution information of the users is shown in Figure 8. Meanwhile, we also stipulate that the range of the ratings that are obtained by the normalization algorithm from users' access behaviours is from 0 to 5.

For comparison purposes, the performance evaluation of the ICFR model is based on the following three criteria: the recommendation accuracy, the time costs and the degree of modularization.

4.2 Recommendation accuracy

The accuracy of the recommendation in the ICFR model is estimated using the expectations of each result in the recommended list. Therefore, the recommendation accuracy *accuracy* can be defined as follows:

$$accuracy = \frac{1}{Count} \sum_{i=1}^{Count} Exp(i) \tag{10}$$

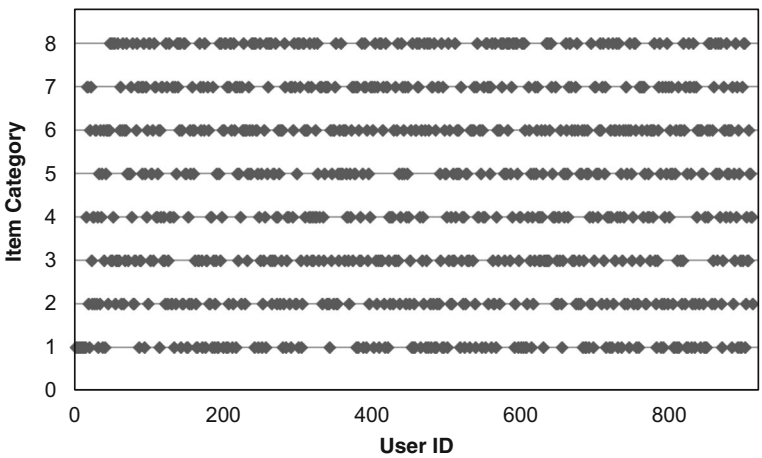


Figure 8 The preference distribution information of the users

where *Count* is the number of recommended results and $Exp(i)$ represents the user's expectation of item i in the range of $0 \leq Exp(i) \leq 1$. In the ICFR model, the number of neighborhoods of the current user when predicting ratings can greatly influence the accuracy. To get the proper number of neighbourhoods, which is expressed as Num_{neigh}, we perform 10 different retrievals with Num_{neigh} set to 8 different values. Then, we compute the *accuracy* of the first N_{res} in the recommended list, where N_{res} represents the number of results in the recommended list from the start. Figure 9 illustrates the *accuracy* with different Num_{neigh}.

From Figure 9, we can obtain two main conclusions: (1) when Num_{neigh} remains the same, as N_{res} increase, *accuracy* is stable at first, but it decreases if N_{res} continues to increase; and (2) when N_{res} remains unchanged, at the start, the *accuracy* is nearly the same in the 8 different Num_{neigh}s, but *accuracy* increases as Num_{neigh} increases. Considering that the increase in Num_{neigh} leads to more calculations and Num_{neigh} influences *accuracy* in our simulated environment, we set the value of Num_{neigh} to 12. In the following experiments, Num_{neigh} is set as 12 and N_{res} is set to 120.

4.3 The accuracy of advanced PCC algorithm

The second experiment compares the accuracy of the PCC algorithm and the advanced PCC algorithm that is used in the ICFR model. Both of them have been described in the previous section. In this experiment, we randomly choose 10 users and perform 10 different individual retrievals using the above two algorithms. Then, we compute the average accuracy. Figure 10 illustrates the average accuracy of 10 users according to the PCC and advanced PCC.

In Figure 10, we can draw the general conclusion that the advanced PCC is more accurate than the traditional PCC for the ICFR model.

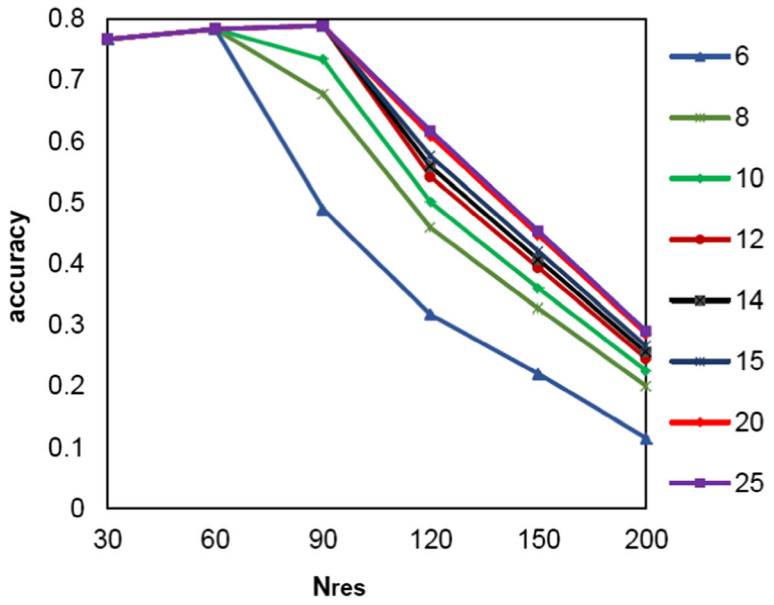


Figure 9 *accuracy* of Different Num_{neigh}

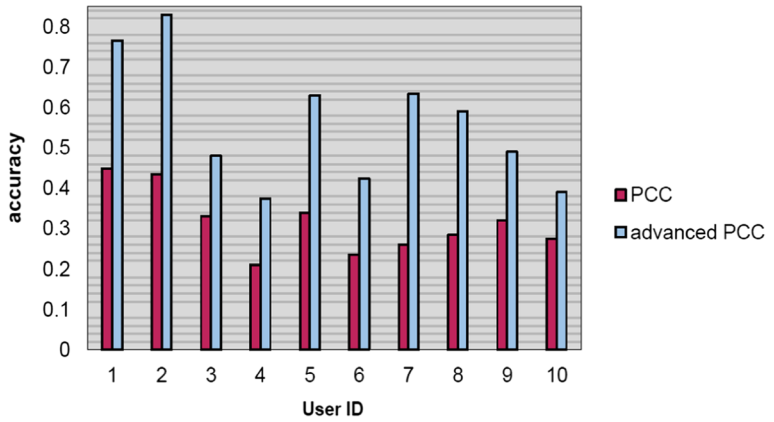


Figure 10 Comparison of PCC and Advanced PCC

4.4 Comparison of UBCF and IBCF

UBCF and IBCF are two main methods of the CF algorithm. The UBCF algorithm is applied to the ICFR model. This experiment is performed to compare the UBCF and IBCF algorithms in terms of their accuracy and diversity. We choose 10 different users and each user performs 10 different retrievals. Then, we compute the average accuracy and diversity values. The diversity *diversity* is defined as follows:

$$diversity = \frac{Cnt_{cover}}{Cnt_{at}} \tag{11}$$

where Cnt_{at} is the total size of all themes on the simulated website, and Cnt_{cover} is the size of the themes that are included in the recommendation results.

The averages *accuracy* and *diversity* are illustrated in Figure 11.

Figure 11 illustrates that UBCF and IBCF have almost the same accuracy but UBCF is obviously superior to IBCF with respect to the diversity in the ICFR model. In addition, another reason for choosing the UBCF algorithm in the ICFR is that there are far more items than users on personalized websites, and UBCF performs fewer calculations than IBCF when the incremental updating computation is performed.

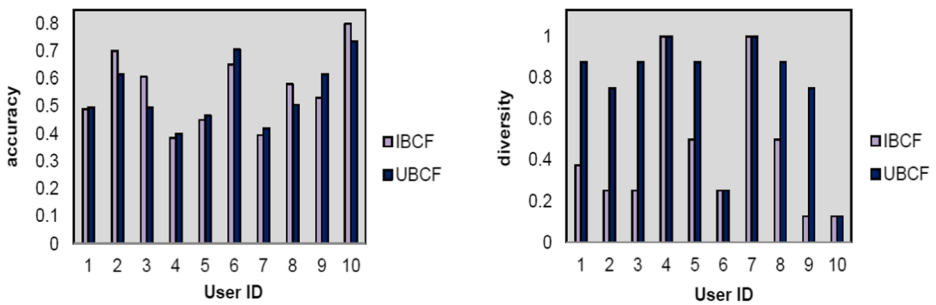


Figure 11 Average *accuracy* and *diversity* of UB and IB

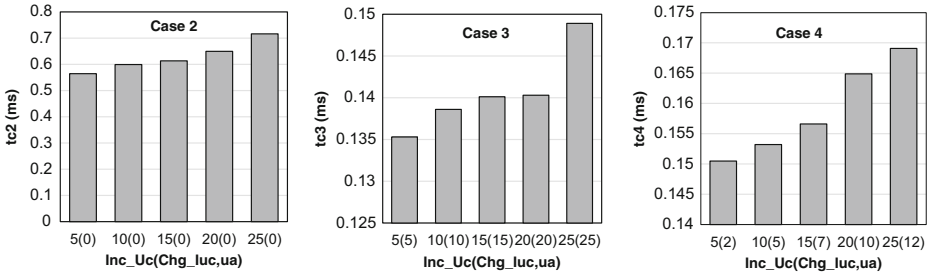


Figure 12 Time cost of Four Cases in IUM

4.5 Time cost evaluation

This experiment compares the time costs of the IUM and TUM. We analyse the situation in the four cases when performing the incremental updating calculations. The time consumption of one single incremental updating process T_{inc} includes the time of the similarity computation between two users and the time of other processing procedures. T_{inc} is defined as follows:

$$T_{inc} = t_{oth} + t_{sim} \tag{12}$$

where t_{sim} is the time of the similarity computation. It satisfies the following:

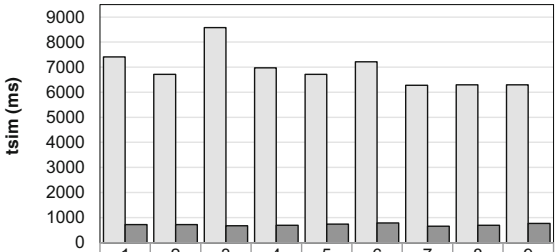
$$t_{sim} = \sum_{i=1}^{4} t_{ci} * num_{ci} \tag{13}$$

where t_{ci} is the time of the case ci and num_{ci} is the frequency of the occurrence of case ci.

t_{oth} represents the time of other processing procedures.

In case1, $t_{c0} = 0$. So the figure of case1 is been omitted, and the comparison of the other three cases is shown in Figure 12. Especially in case4, we assume that all the elements in $Chg_{I_{u_c,u_\alpha}}$ are from $Inc\_I_{u_c}$, and the size of $Chg\_I_{u_c,u_\alpha}$ is half of $Inc\_I_{u_c}$. The result is shown in Figure 12.

Another time cost comparison experiment is conducted to assess the IUM and TUM in the ICFR model. Only the current user's similarity information is needed to compute the IUM in the ICFR. However, in the TUM, the whole similarity matrix should be updated. Therefore, we conduct 9 different updating processes in the above two mechanisms. The experiment results are shown in Figure 13.



	1	2	3	4	5	6	7	8	9
TUM	7413	6714	8579	6972	6716	7216	6277	6294	6296
IUM(single user)	722	717	673	694	736	782	658	690	763

Figure 13 Time Cost Comparison between IUM (one user) and TUM

We can see from Figure 13 that the TUM costs far more time than the IUM and the latter occupies approximately 10.3% of that of the former.

4.6 Performance of the IUM

The next experiment is performed to demonstrate the effectiveness of the IUM in the UBCF algorithm. Different numbers of users Num_{user} (Num_{user} = 300, 500, 700, 910) are simulated. When Num_{user} is constant, we use two different updating mechanisms (IUM and TUM) and simulate 10 visits of one single user. In each visit, the user performs browsing behaviours (e.g., browsing Web content and collecting pages) approximately 10 times. Furthermore, the user's browsing behaviours in the 10 visits are exactly the same in these different cases. The *UUSM* is updated each time the current user conducts a $T_{session}$ operation in the IUM, while it is updated every four $T_{session}$s in the TUM. Figure 14 shows the *accuracy* of these cases.

In Figure 14, we can see that as the number of the users increases, the *accuracy* of the ICFR model remains basically the same. In addition, the IUM has a better recommendation effect than the TUM, which is because the TUM does not update in real time.

4.7 Modularity of the functions

In this paper, a recommendation model for personalized websites is proposed, and the key functions are modularized. The website developers can realize the query recommendation

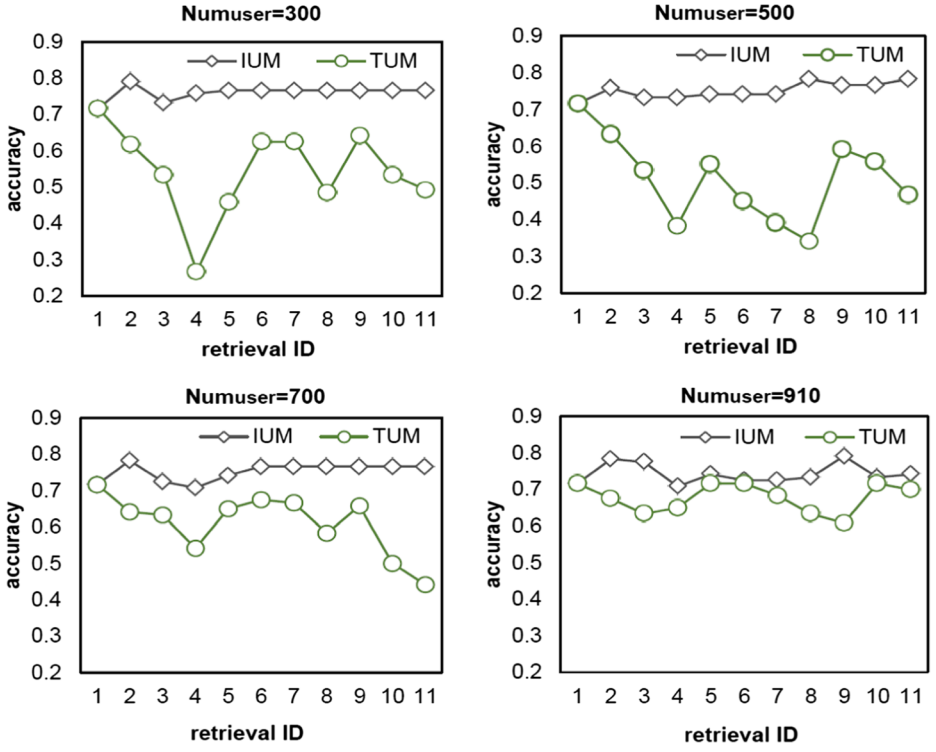


Figure 14: *Accuracy* of Different Num_{user}

Table 1 Comparison of Code Lines

Main Function	Branch Function	Total Amount of Code (Line)	Total Amount of Code to Call (Line)	Proportion
Similarity Calculation Module	Normalization Calculation	14	1	7.14%
	Similarity Calculation	65	1	1.54%
	Rating Prediction Calculation	78	1	1.28%
	Total	157	3	1.91%
IUM	/	227	2	0.88%

function based on the users' intentions without modifying or adding a great deal of codes to the original website. In the ICFR model, the similarity computations (which includes normalization calculations, similarity calculations and rating prediction calculations) and incremental updating computations are the two most important processes. In this experiment, we compare the lines of the key function codes that are needed in the model before and after modular processing. Table 1 illustrates the comparison.

Table 1 shows that it can significantly reduce the workload of personalized website developers when implementing recommendation functions after modularization.

5 Conclusions and future work

In this paper, we propose a recommendation model – the ICFR model – for personalized websites. We first select the user-based collaborative filtering algorithm and apply it to the model. We also redefine the three major elements of the UBCF algorithm, which are the users, items and ratings. Second, we introduce the specific scheme that shows how to apply the UBCF algorithm to the personalized website. In the scheme, an advanced PCC is used to compute the similarity between users. Third, we address the update mechanism of the UBCF. Compared with the traditional updating method, this mechanism reduces the calculations of the system to a certain extent. Finally, we modularize the key functions in the ICFR model so that Web developers can reuse it.

The model that is proposed in this paper has achieved satisfactory recommendation results, but it has high concurrency when the ICFR model is performing incremental updating operations. Since the updating operation is triggered by the end of the current session of the user, it may cause a large amount of computations for the server if many users end sessions at the same time. Moreover, the modified framework has certain universality and can be applied to various personalized website optimization.

In our future work, we plan to find a better incremental updating strategy for the collaborative filtering recommendation algorithm.

Acknowledgements This work is supported by Natural Science Foundation of China (61672535, 61472005), Natural Science Foundation of Hunan Province (2019JJ20025, 2018JJ3203), Research Foundation of Education Bureau of Hunan Province (17C0679), Hunan University of Science and Engineering Research Project (17XKY071) and the construct program of applied characteristic discipline in Hunan University of Science and Engineering. The authors declare that they have no conflict of interests. This manuscript presents the extended work based on our former publication "A novel recommendation framework towards personalized websites" in CyberSciTech 2018.

References

1. Agarwal, V., Bharadwaj, K.K.: A collaborative filtering framework for friends recommendation in social networks based on interaction intensity and adaptive user similarity[J]. Soc. Netw. Anal. Min. **3**(3), 359–379 (2013)
2. Aggarwal C C. Content-based recommender systems[M]//recommender systems. Springer International Publishing: 139–166 (2016)
3. Aggarwal C C. Model-based collaborative filtering[M]//recommender systems. Springer International Publishing: 71–138 (2016)
4. Bellogín, A., Castells, P., Cantador, I.: Neighbor selection and weighting in user-based collaborative filtering: a performance prediction approach[J]. ACM Transactions on the Web (TWEB). **8**(2), 12 (2014)
5. Benesty, J., Chen, J., Huang, Y., et al.: Pearson Correlation Coefficient[M]//Noise Reduction in Speech Processing, pp. 1–4. Springer, Berlin Heidelberg (2009)
6. Chang A D, Liao J F, Chang P C, et al.: Application of artificial immune systems combines collaborative filtering in movie recommendation system[C]//computer supported cooperative work in design (CSCWD), proceedings of the 2014 IEEE 18th international conference on. IEEE 277–282 (2014)
7. Chen X, Xia M, Cheng J, et al.: Trend prediction of internet public opinion based on collaborative filtering[C]//natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD), 2016 12th international conference on. IEEE, 2016: 583–588 (2016)
8. de Gemmis M, Lops P, Musto C, et al.: Semantics-aware content-based recommender systems[M]// recommender systems handbook. Springer US: 119–159 (2015)
9. Elahi M, Ricci F, Rubens N. Active learning in collaborative filtering recommender systems[C]//international conference on electronic commerce and web technologies. Springer International Publishing: 113–124 (2014)
10. Fernández-Tobías, I., Braunhofer, M., Elahi, M., et al.: Alleviating the new user problem in collaborative filtering by exploiting personality information[J]. User Model. User-Adap. Inter. **26**(2–3), 221–255 (2016)
11. George T, Merugu S. A scalable collaborative filtering framework based on co-clustering[C]//Data Mining, Fifth IEEE international conference on. IEEE, 2005: 4 pp.
12. Tang Y., Wang H., Guo K., et al. Relevant Feedback Based Accurate and Intelligent Retrieval on Capturing User Intention for Personalized Websites[J]. IEEE Access. **6**, 24239–24248 (2018)
13. Hasan M, Ahmed S, Malik M A I, et al.: A comprehensive approach towards user-based collaborative filtering recommender system[C]//computational intelligence (IWCI), international workshop on. IEEE: 159–164 (2016)
14. Herlocker J L, Konstan J A, Borchers A, et al.: An algorithmic framework for performing collaborative filtering[C]//proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval. ACM: 230–237 (1999)
15. Jamali M, Ester M.: Trustwalker: a random walk model for combining trust-based and item-based recommendation[C]//proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. ACM: 397–406 (2009)
16. Jia, D., Zhang, F., Liu, S.: A robust collaborative filtering recommendation algorithm based on multidimensional trust model[J]. JSW. **8**(1), 11–18 (2013)
17. Jia Z, Yang Y, Gao W, et al. User-based collaborative filtering for tourist attraction recommendations[C]// Computational Intelligence & Communication Technology (CICT), 2015 IEEE international conference on. IEEE: 22–25 (2015)
18. Jin, R., Si, L., Zhai, C.: A study of mixture models for collaborative filtering[J]. Inf. Retr. **9**(3), 357–382 (2006)
19. Guo K., Liang Z., Shi R., et al. Transparent learning: An incremental machine learning framework based on transparent computing[J]. IEEE Network. **32**(1),146-151 (2018)
20. Li J, Wang Y, Wu J, et al.: Application of User-Based Collaborative Filtering Recommendation Technology on Logistics Platform[C]//Business Intelligence and Financial Engineering (BIFE), 2013 Sixth international conference on. IEEE: 135–138 (2013)
21. Li W, Xu H, Ji M, et al.: A hierarchy weighting similarity measure to improve user-based collaborative filtering algorithm[C]//computer and communications (ICCC), 2016 2nd IEEE international conference on. IEEE, 2016: 843–846 (2016)
22. Ma W, Ren C, Wu Y, et al. Personalized recommendation via unbalance full-connectivity inference[J]. Physica A: Statistical Mechanics and its Applications, 2017, 483: 273–279
23. Meehan K, Lunney T, Curran K, et al.: Context-aware intelligent recommendation system for tourism[C]// pervasive computing and communications workshops (PERCOM workshops), 2013 IEEE international conference on. IEEE 328–331 (2013)

24. Papagelis M, Rousidis I, Plexousakis D, et al. Incremental collaborative filtering for highly-scalable recommendation algorithms[C]//International Symposium on Methodologies for Intelligent ‘
25. Guo K., Liang Z., Tang Y., et al. SOR: An optimized semantic ontology retrieval algorithm for heterogeneous multimedia big data[J]. Journal of computational science. **28**, 455- 465 (2018)
26. Sarwar, B., Karypis, G., Konstan, J., et al.: Item-based collaborative filtering recommendation algorithms[C]//proceedings of the 10th international conference on world wide web. ACM. 285–295 (2001)
27. Shardanand U, Maes P.: Social information filtering: algorithms for automating “word of mouth”[C]// proceedings of the SIGCHI conference on human factors in computing systems. ACM Press/Addison-Wesley Publishing Co.: 210–217 (1995)
28. Veena C, Babu B V. A User-Based Recommendation with a Scalable Machine Learning Tool[J]. International Journal of Electrical and Computer Engineering, 2015, 5(5)
29. Wang Y, Feng D, Li D, et al.: A mobile recommendation system based on logistic regression and gradient boosting decision trees[C]//neural networks (IJCNN), 2016 international joint conference on. IEEE: 1896–1902 (2016)
30. Wang, J., Cao, Y., Li, B., et al.: Particle swarm optimization based clustering algorithm with mobile sink for WSNs[J]. Futur. Gener. Comput. Syst. **76**, 452–457 (2017)
31. Wang, J., Cao, J., Ji, S., et al.: Energy-efficient cluster-based dynamic routes adjustment approach for wireless sensor networks with mobile sinks[J]. J. Supercomput. **73**(7), 3277–3290 (2017)
32. Zhang, J., Peng, Q., Sun, S., et al.: Collaborative filtering recommendation algorithm based on user preference derived from item domain features[J]. Physica A: Statistical Mechanics and its Applications. **396**, 66–76 (2014)
33. Zhao Z D, Shang M S. User-based collaborative-filtering recommendation algorithms on hadoop[C]// Knowledge Discovery and Data Mining, 2010. WKDD'10.Third International Conference on. IEEE: 478–481 (2010)
34. Zhou, X., Wu, B., Jin, Q.: Analysis of user network and correlation for community discovery based on topic-aware similarity and behavioral influence[J]. IEEE Transactions on Human-Machine Systems. **48**(6), 559–571 (2018)
35. Zhou X, Liang W, Kevin I, et al.: Academic Influence Aware and Multidimensional Network Analysis for Research Collaboration Navigation Based on Scholarly Big Data[J]. IEEE Transactions on Emerging Topics in Computing, (2018)