Personalized Recommendation System Based on Collaborative Filtering for IoT Scenarios

Zhihua Cui, Xianghua Xu, Fei Xue, Xingjuan Cai, Yang Cao, Wensheng Zhang, and Jinjun Chen, *Senior Member, IEEE*

Abstract—Recommendation technology is an important part of the Internet of Things (IoT) services, which can provide better service for users and help users get information anytime, anywhere. However, the traditional recommendation algorithms cannot meet user's fast and accurate recommended requirements in the IoT environment. In the face of a large-volume data, the method of finding neighborhood by comparing whole user information will result in a low recommendation efficiency. In addition, the traditional recommendation system ignores the inherent connection between user's preference and time. In reality, the interest of the user varies over time. Recommendation system should provide users accurate and fast with the change of time. To address this, we propose a novel recommendation model based on time correlation coefficient and an improved K-means with cuckoo search (CSK-means), called TCCF. The clustering method can cluster similar users together for further quick and accurate recommendation. Moreover, an effective and personalized recommendation model based on preference pattern (PTCCF) is designed to improve the quality of TCCF. It can provide a higher quality recommendation by analyzing the user's behaviors. The extensive experiments are conducted on two real datasets of MovieLens and Douban, and the precision of our model have improved about 5.2 percent compared with the MCoC model. Systematic experimental results have demonstrated our models TCCF and PTCCF are effective for IoT scenarios.

Index Terms—Recommender systems, collaborative filtering, clustering model, personalized recommendation

1 Introduction

With the rapid development of network technology, many technologies are applied to the IoT scenarios, and recommendation technology is a good application. The exponential growth of information resources in IoT has become one of the main barriers for users to efficiently and effectively extract useful information from all the available database. Such an overwhelming amount of data necessitates mechanisms for efficient information filtering. Traditional search services retrieve the same sorting information to all users and cannot personalize the searching results according to different user's interests. To overcome this difficulty, recommender systems have been developed to provide personalized recommendation service for users specific preference and have gained increasing attention in both academic and industrial research.

Recommendation technology is an important part of the Internet of Things (IoT) services [1], which can provide better service for users and help users get information anytime, anywhere. Among them, collaborative filtering (CF) is a well known and widely adopted recommendation technique [2], [3], [4], [5], [6], [7]. The main concept is to acquire the recommendation based on the similarity of users' interests. In other words, CF first analyzes one user's preference according to his behavior records and then makes the recommendations based on others' preferences with similar interests. Collaborative filtering has no special requirements (e.g., description, metadata) for the recommended products. It can deal with various objects, such as books, movies, music. Therefore, it has been widely applied in commercial applications [8], [9], [10]. A recent report from Ref. [11] showed that the recommendation system of Amazon provided more than 30 percent of the sales volume. In addition, recommendation system is also an important part of cloud computing [12], [13], [14], [15].

Although traditional collaborative filtering techniques have achieved great success, some drawbacks still exist and limit their applications to large-volume, complex, and growing database IoT environment. One significant issue of the user-item data is the data sparsity. As reported in Ref. [16], the data sparsity of MovieLens is only 6.3 percent. Another shortcoming is the cold-start [17], i.e., the collaborative filtering cannot make the recommendation if a new item does not have any rating, or a new user does not evaluate any of the items. In addition, user's personalized needs are not considered in most of the traditional models [18]. Therefore, how to precisely and efficiently extract useful information and predict

- *Z. Cui and X. Cai are with the College of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan, Shanxi 030024, China. E-mail: cuizhihua@gmail, xingjuancai@163.com.*
- *X. Xu is with Hangzhou Dianzi University, Hangzhou 310018, China. E-mail: xhxu@hdu.edu.cn.*
- *F. Xue and Y. Cao are with the School of Information, Beijing Wuzi University, Beijing 101149, China. E-mail: xuefei2004@126.com, caoyangcwz@gmail.com.*
- *W. Zhang is with the State Key Laboratory of Intelligent Control and Management of Complex Systems, Institute of Automation Chinese Academy of Sciences, Beijing 100190, China. E-mail: wensheng.zhang@ia.ac.cn.*
- *J. Chen is with Complex System and Computational Intelligence Laboratory, Taiyuan University of Science and Technology, Taiyuan, Shanxi 030024, China, and also with Swinburne Data Science Research Institute, Swinburne University of Technology, Hawthorn, VIC 3122, Australia. E-mail: jinjun.chen@gmail.com.*

Manuscript received 14 Oct. 2018; revised 28 Sept. 2019; accepted 20 Dec. 2019. Date of publication 7 Jan. 2020; date of current version 10 Aug. 2020. (Corresponding author: Xingjuan Cai.)
Digital Object Identifier no. 10.1109/TSC.2020.2964552

user preferences in the big data environment is of significant theoretical and industrial interest.

Challenges. IoT service system involves large-volume, complex, and growing data [19]. That massive information is usually high-dimensional, sparse, and rich in content but complex in structure, indicating that traditional data processing techniques are inadequate to deal with them. Moreover, the rapid growth of the numbers and types of goods causes customers to spend more time in finding the desired products. Browsing a lot of irrelevant information is also time-consuming. Such exhausting purchasing may result in the loss of a large number of customers.

Another challenge is the interest drifting. Traditional recommendation system assumes that the user's interest is unchanged, ignoring the inherent connection between user's preference and time. In reality, the interest of the user varies with time. Recommendation system should predict users' real-time requirements and give the recommendation accordingly. To the authors' best knowledge, few studies were focused on the time effect in the recommendation systems. Therefore, there is a critical need to develop a personalized recommendation system with the consideration of the pattern of user's preferences to handle big data accurately and efficiently.

Contributions. To address the above challenges, this paper makes the following contributions:

- A novel collaborative filtering algorithm based on time correlation coefficient (TCC) and CSK-means (TCCF) is proposed. TCCF can cluster similar users together for further quick and accurate recommendation.
- An effective and personalized recommendation model based on preference pattern (PTCCF) is proposed to improve the quality of TCCF.
- The extensive experiments are carried out on MovieLens and Douban, and systematic experimental results have demonstrated our models TCCF and PTCCF are effective for a fast and accurate recommendation.

The rest of this article is organized as follows. In the next section, a brief review is given of recent investigations on classic CF models, clustering-based CF models, and time-aware CF models. In Section 3, we propose in detail the algorithms and implementation of the collaborative filtering scheme based on TCC and CSK-means (TCCF). Section 4 presents our personalized recommendation model based on preference pattern. This method is then applied to some test cases. The last section concludes this paper.

2 Background and Related Work

This section briefly presents an overview of related research about traditional collaborative filtering, improved collaborative filtering models based on clustering and time-based clustering collaborative filtering.

2.1 Collaborative Filtering Models

Collaborative filtering is a classic recommendation model, which is widely used in Web and IoT services [20], [21]. Traditional CF uses similar users (e.g., similar preferences) recommended items [22], which can be classified into neighborhood-based and model-based algorithms. The neighborhood-based approach, also known as the memory-based approach, uses statistical techniques to build the neighborhood relationship between users (user-based CF model) or between items (item-based CF model) [21]. The former is to explore the users with similar tastes in items and then interlocks based on their favorite items. The latter is to study the users' favorite items [23], and then recommends similar products.

Model-based approach, in contrast, mainly learns a corresponding model from the training data (e.g., user's rating on items). This idea is to utilize data mining techniques to recognize the data patterns and to figure out the connection between users and objects. Some interesting models can be found in Ref. [3], [24]. Besides multi-objective optimization algorithms [25], [26], [27], [28] can improve the quality of recommendation model by trade-offs among different evaluation indicators [29].

Although the traditional collaborative filtering techniques have achieved great success, some drawbacks, such as scalability and cold-start, still exist. To date, many techniques and models have been proposed to address these issues. For example, Matrix Factorization models (MF), Singular Value Decomposition (SVD)[30] can solve the data sparsity problem effectively. To eliminate the cold-start, some hybrid recommendation models were proposed [17], [31], [32]. For example, Sharma *et al*. designed a personalized recommendation model named FBSM [32], which learns insightful relations among items based on factorized bilinear similarity model. Extensive experiments demonstrated the validity of that approach.

2.2 Clustering-Based Collaborative Filtering Models

Large-volume, multi-dimensional and growing data in Big data brings a challenge to the traditional recommendation system [2]. Clustering-based collaborative filtering techniques [33], [34], [35], [36], [37] are used for dealing with this problem by reducing the data size with accurate prediction. For the collaborative filtering, clustering is a pre-processing that collects the information from similar users for further recommendation.

Clustering-based collaborative filtering methods have been investigated extensively, which include several types: user-based clustering, item-based clustering, and co-clustering. Zahra *et al.* [36] employed the K-means algorithm to cataloging movies based on the ratings provided by users. Users themselves can also be clustered based on the item cluster they rated. Bu *et al.* [3] designed a Multiclass Co-Clustering (MCoC) model, which utilized the user-item subgroups to group similar users. Simulation results proved that the subgroups strategy could improve recommendation performance compared to other methods.

Some other clustering models tried to include the user's social network into consideration. Liu *et al*. [38] used the social network technique to identify a user's neighborhood, and then employed a classic CF algorithm to provide the recommendations. Their approach depends on the social relationships among users. Hu *et al.* proposed a novel CF model, named ClubCF [34], to provide like-minded services. The strategy of two stages of ClubCF, clustering and collaborative filtering, make it capable of providing good service in a large database environment.

2.3 Time-Aware Collaborative Filtering Models

Most traditional collaborative filtering algorithms only use the rating information to make recommendations without the time effect. Apparently, the active usage of the time information will can generate more accurate recommendations [39], [40].

The recommendation algorithm based on time series information enables the model to learn the dynamic changes of users preferences and to further optimize the recommendation decision. Guo *et al.* [41] proposed a novel recommendation algorithm based on time weights of different items. Their assumption is that the users' purchase habits change with time. Thus the weight to previous data is diminished. Koren *et al.* [42] added time information to user's feature vector. Their model traced the behavior changing throughout the data life and effectively solved the interest drifting.

In addition, some works combine time factors with other factors (e.g., tags of the item) to achieve better performance. In Ref. [40], tripartite graphs are used to catch the relationship between user-item-tag and user-item-attribute. With such basis, a new personalized recommendation algorithm was developed with all the information of users, tags, attributes of items and time weights. Li *et al.* proposed a cross-domain CF framework based on time and Bayesian latent factor model [43]. Their method can visualize user-interest drift over time.

Different from the above works, this paper uses time information to learn the model of user preference behavior. Our new recommendation framework only estimates the similarity of the users with the same pattern for rapid recommendation making. Furthermore, our model can provide a personalized recommendation for different users.

3 Pruning Based on Improved K-Means

In this section, an improved K-means clustering with optimized centroids by Cuckoo search [44], [45], [46] and time-aware is designed to improve the classic CF models. The clustering method is a pre-processing to cluster similar users together for a quick and accurate recommendation.

3.1 Classic CF Model

The traditional CF model generally uses user/item-based neighbor algorithms to find similar users/items. The user-based CF model generally includes three stages: Calculate the correlation between users; Find the similar neighbors (users with similar preferences); Recommended the high-quality preferences of neighbors to the user.

Calculating the user's similarity based on their historical raking records is the key of CF model. The correlation is usually calculated using the cosine similarity and Pearson correlation coefficient (PCC). PCC is the most popular method, which (between users u and v) can be expressed follows:

$$sim(u,v) = \frac{\sum_{i \in P_{uv}} (r_{ui} - \overline{r_u})(r_{vi} - \overline{r_v})}{\sqrt{\sum_{i \in P_{uv}} (r_{ui} - \overline{r_u})^2} \sqrt{\sum_{i \in P_{uv}} (r_{vi} - \overline{r_v})^2}}, \quad (1)$$

where, $\overline{r_u}$ and $\overline{r_v}$ are the average scores of users u and v respectively.

Then, some users who are closest to the target user (highest similarity) are selected as the nearest neighbors. The current user's preference for a item is predicted according to the actual scores of the nearest neighbors. The prediction score is calculated as follows:

$$P_{ui} = \overline{r_u} + \frac{\sum_{v \in NBS_u} sim(u,v) \times (r_{vi} - \overline{r_v})}{\sum_{v \in NBS_u} sim(u,v)}, \quad (2)$$

where, P_{ui} is predictive score of user u for item i, r_{vi} is the score of user v for item i, and NBS_u is the neighbor set of user u.

Finally, several items with higher scores are selected as the recommended results to the current user.

Unfortunately, with the explosive growth of information in Big data environment, traditional collaborative filtering algorithm cannot meet user's fast recommended requirements. In the face of a large-volume data, the method of finding neighborhood by comparing whole user information will result in a low recommendation efficiency of the algorithm. At the same time, this algorithm ignores the higher interest similarity of the users who evaluate the same item simultaneously, which makes the algorithm reduce the recommendation accuracy.

To solve the above problems, this paper employs the K-means algorithm to separate big data problem into several smaller manageable problems. The clustering method is a pre-processing that cluster similar users together for further quick and accurate recommendation. Moreover, we design a time factor to trace the similarity of interest with time.

However, for the classical K-means algorithm, different clustering effects may be obtained due to the selection of initialization factors. The selection of initialization factors is a complex optimization problem. Intelligent optimization algorithm is an effective way to solve the complex optimization problems [47], [48], [49], [50]. In this paper, we employ a novel intelligent optimization algorithm, Cuckoo search, to optimize the initialization factor of a clustering algorithm, so as to improve the clustering effect.

3.2 Improved K-Means Based Cuckoo Search

3.2.1 K-Means

K-means clustering model [51] is a well known and widely adopted clustering algorithm. The similarity within clusters is high, and the similarity between clusters is low. The K-means can be composed of the following steps:

1) Select k objects randomly from dataset as the initial group centroids;
2) Calculate the distance of each object to all centroids, and then assign it to the closest cluster;
3) Update the centriods of all clusters;
4) Repeat (2)$\sim$(3) until the termination condition (e.g., all centroids are not changing or reaches the number of iterations) is satisfied.

Some studies use K-means to optimize the collaborative filtering algorithm [36]. Different cluster centers will lead to different clustering effects. If an outlier is chosen, the clustering result is terrible. The initial centers optimization is an optimization problem.

There are various forms of data clustering analysis methods, and some algorithms are very good in terms of computational efficiency or accuracy. However, quite a few of these algorithms require users to provide a priori information about clustering analysis. The sensitive to the input parameters greatly

reduces the adaptability of the algorithm. A swarm intelligent algorithm is an effective solution to resolve complex problems. To address this, we use a simple and efficient algorithm, cuckoo search (CS), to optimize the initial center of K-means.

3.2.2 CS Algorithm

CS is a novel swarm intelligence optimization algorithm inspired by the cuckoo searching behaviour. Since the CS was proposed, it has been applied to many optimization problems due to its usability and search abilities. Cui *et al.* [45] used a modified cuckoo search algorithm to improve the performance of DV-Hop.

Algorithm 1. Cuckoo Search

Input: $seting$: parameters setting; $f(x)$: objective function;
Output: $results$: optimal solution for $f(x)$;
$(cockoos, p_a, N) \leftarrow$ **Initialize**$(seting)$;
$fitness \leftarrow$ **Evaluate**$(cockoos)$;
while $t < maxIterations$ **do**
  for $cockoo_i \subset cockoos$ **do**
    $x_i(t^{'}) \leftarrow$ **updatePosition**$(x_i(t))$ as Eq. (3);
    $fitness \leftarrow$ **Evaluate**$(cockoos)$;
    if $rand > p_a$ **then**
      $x_i(t^{'}) \leftarrow$ **updatePosition**$(x_i(t))$ as Eq. (4);
      $fitness \leftarrow$ **Evaluate**$(cockoos)$;
    end
  end
  $results \leftarrow$ **getOptimal**$(cockoos)$;
end
Return$(results)$;

For CS, there are some idealized rules:

- Each cuckoo lays one egg (corresponds to an optimization objective) at a time and puts the egg in a randomly chosen nest.
- Those best nests with a high quality of egg (i.e., solution) will be retained for subsequent generations.
- For each cuckoo, if its nest if is found by the owner ($p_a \in [0, 1]$), it will built a new nest with one egg by *Lévy* flight.

Based on the above three idealized rules, the global location update formula is as follows:

$$x_i(t^{'}) = x_i(t) + \alpha * 0.01 * L * (x_i(t) - p_g(t)) * r, \tag{3}$$

where $x_i(t^{'})$ and $x_i(t)$ are two position for ith cuckoo at time t. $\alpha > 0$ is the weight of moving step. L is a random number based on *Lévy* distribution. r is a random number based on Gaussian distribution $N(0, 1)$.

For the local search (cuckoo is found with $r > p_a$), the update formula is as follows:

$$x_i(t+1) = x_i(t^{'}) + rand * (x_k(t^{'}) - x_j(t^{'})), \tag{4}$$

where $x_k(t^{'})$ and $x_j(t^{'})$ are randomly selected from the current positions of cuckoo. The pseudo codes of CS are listed in Algorithm 1. For each cuckoo, the *initialize* function is used to initialize cuckoo's positions; the *evaluate* function is used to evaluate its fitness; the function *updatePosition* is used to update its position according to the rules. Finally, we obtain an optimal solution for multi-objective function $f(x)$ when the stop criterion is met. Cuckoo search is an uncertain algorithm, and its time complexity is related not only to the number of iterations and population size, but also to the fitness function. For the simplest case that the fitness is $O(1)$, the time complexity of cuckoo search is $O(n^2)$.

3.2.3 CSK-Means Algorithm

K-means clustering model has two crucial parameters: the number k of clusters and initial centers, which is randomly selected in the traditional algorithm. Randomly selection for those centers will impact the final clustering effect. In this subsection, we designed a variant of K-means based on CS, named CSK-means, which employ the CS to optimize the initial centers.

The evaluation function of clustering quality is used as the fitness function. μ_j is the centroid of the cluster $C_j (j = 1, 2, \ldots, k)$, the fitness function of CS can be defined by the formula:

$$f(x) = \sum_{j=1}^{k} \sum_{x_i \in C_j} \parallel x_i - \mu_j \parallel, \tag{5}$$

where, x_i is a vector of user's raking on items. For $\sum \parallel x_i - \mu_j \parallel, x_i \in C_j$ represents the density of the data, the smaller, the more dense, and the higher the quality of the cluster. CSK-means algorithm can be composed of the following steps:

Step 1: Select k data items from datasets D randomly as the position of cockoos;

Step 2: Optimize the initial cluster center by Algorithm 1 with Eq. (5);

Step 3: Run the K-means clustering algorithm and output the optimized clusters;

3.3 Proposed CF Based on TCC and CSK-Means

The traditional collaborative filtering (CF) techniques recommend items to users based on their historical ratings. In reality, the interest of the user varies over time since they are affected by moods, friends, and fashion trends, that is, interests drift. For a certain period, a user's interests may only focus on one or a few items. Therefore, traditional CF models based on the whole historical records may recommend inappropriate items.

We design an improved PCC based on time correlation coefficient (TCC) to describe the user similarity over time in the collaborative filtering recommendation algorithm. Then we use the above clustering algorithm to cluster the user information, and finally use the user-based CF algorithm in the user cluster to recommend the target user, so as to achieve high-precision and efficient collaborative filtering algorithm.

3.3.1 Time Correlation Coefficient

Interest drift means that a user's interests on items changes over time. In CF, we want to find neighbors that are similar to current users' recent interests. The user's interests in items generally follow a normal distribution. So the TCC can be given as follows:

$$tcc_i = 1 - \frac{1}{\sqrt{2\pi}\sigma}\left(1 - \exp\left(-\frac{\Delta t^2}{2\sigma^2}\right)\right), \tag{6}$$

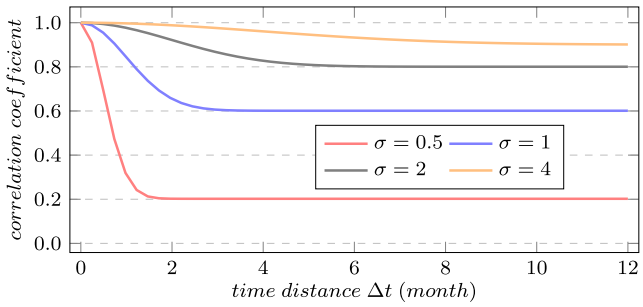


Fig. 1. Time similar coefficient.

where, $\Delta t = t_i - t_1$ is temporal distance, t_1 is the timestamp of the latest ranking item, t_i is the timestamp of the ith ranking item. σ is an attenuation coefficient of interest in item. Different σ corresponds to different models, as shown in Fig. 1. When $\sigma = 0.5$, users' interest fell fast. When $\sigma = 4$, the user has always had a high interest.

The improved formula for calculating the user similarity is as follows:

$$sim(u,v)_{new} = \frac{\sum_{i \in P_{uv}} (r'_{ui} - \overline{r'_u})(r'_{vi} - \overline{r'_v})}{\sqrt{\sum_{i \in P_{uv}} (r'_{ui} - \overline{r'_u})^2} \sqrt{\sum_{i \in P_{uv}} (r'_{vi} - \overline{r'_v})^2}}, \tag{7}$$

where, r'_u is the new raking vector of user u, which can be given as follows:

$$r'_u = r_u \times tcc. \tag{8}$$

3.3.2 TCCF Model

Users with higher similarity to the target user have a more valuable reference than other users. In this section, we proposed a improved CF model based on TCC and CSK-means (TCCF) to provide a quick and accurate recommendation. The TCCF model is shown in Algorithm 2.

Algorithm 2. TCCF Model

Input: $seting$: parameters setting; $data$: user-item raking matrix; u: recommended user;
Output: $results$: top-N recommended items;
1: $(k, N) \leftarrow$ **Initialize**$(seting)$;
2: $clusters \leftarrow$ **clusterUsers**$(data, k)$ by CSK-means;
3: **if** $u \in Users$ **then**
4: $neighbors \leftarrow$ **getNeighbors**$(u, clusters)$;
5: $results \leftarrow$ **recommendItems**$(neighbors, data, N)$ as Eq. (2);
6: **end**
7: **else**
8: $results \leftarrow$ **recommendItems**$(data, N)$ as Eq. (9);
9: **end**
10: **Return**$(results)$;

The input of TCCF is: cluster number k, recommended number of items N, user-item raking matrix $data$ and other parameters for CS. First, users are clustered based on CSK-means (line 2). Then, if the target user has records, find his neighbors, and recommend the neighbor's favorite items to him (lines 3-6).

If the user u is a newer, we recommend the most popular items to the new target users (line 8). The formula of the item popularity can be given as follows:

$$ItemPop_i = \frac{|U_i|}{\sqrt{\sum_{i \in I} |U_i|^2}}, \tag{9}$$

where, U_i is the set of users evaluated for item i, I is the collection of all items.

4 Personalized Recommendation

The traditional collaborative filtering recommendation algorithm usually adopts user-based or item-based recommendations to users. These two methods recommend the high-score items to users. However, these items may be the same type. A single type of recommendation cannot meet the needs of some users. To address this, we proposed a novel personalized recommendation model, which recommend different items for users based on their preference pattern.

4.1 Preference Pattern Based on Time Entropy

The user's behavioral preferences are different. Some users have always liked the same type of movies. For some users, in contrast, the favorites types of movies may change over time since they are affected by friends, fashion trends, or moods. According to the evaluation of an item, the user's preference can be divided into the *Likes* and *Dislikes*. Similarly, if measured by time, it includes *Recent* and *Past*. So we simply divide users into four patterns according to the user's behavioral preferences.

The first pattern of users is $RecentLikes, PastLikes$. They are regular users with a regular interest. Such users usually only like one type of movie, and the duration of the movie has not changed for a long time. The second pattern of users is $RecentLikes, PastDislikes$, the third pattern is $Recent DisLikes, Pastlikes$. For these users, the interests varies over time. The last type is, for other users, $RecentDisLikes, PastDislikes$. The information evaluated by such users presents a diversified trend and the items appear randomly and irregularly.

The value and time information of raking on items imply users' behavior patterns. For example, if a user frequently accesses the same type of movie, we think he is a user who follows the first pattern. This pattern is simple, the same type of film is evenly distributed. On the contrary, for the users in the fourth pattern, there is no regularity.

Inspired by this, we proposed a novel pattern mining method based on $time\ entropy$. It is a measure of time distribution of frequent items or tags in users historical records, which can be measured as follows:

$$H = -\sum_{i=1}^{n} \frac{m_i \cdot score_i}{M} \log \frac{m_i}{M}, \tag{10}$$

where, $score_i$ is the raking on item for user i, m_i is the sum of raking on an item in a time interval, M is the sum of all m_i.

Usually, an item has multiple tags, which are a simple description of the item. For a film, the tags can represent its type. For a user, he can comment same tags at the different time, namely, the scores of a tag subordinate to several time

TABLE 1
An Example of User-Item Matrix With Four Users Raking on Eight Items

	$i_1(tag_1, tag_3)$	$i_2(tag_1)$	$i_3(tag_1, tag_3)$	$i_4(tag_3)$	$i_5(tag_1, tag_2, tag_3)$	$i_6(tag_1, tag_2)$	$i_7(tag_2, tag_3)$	$i_8(tag_1, tag_3)$
u_1	$(5,\ t_1)$	$(3,\ t_2)$	$(2,\ t_1)$	$(4,\ t_2)$	$(3,\ t_3)$	$(2,\ t_3)$	$(1,\ t_2)$	$(4,\ t_3)$
u_2	$(2,\ t_1)$	$(1,\ t_2)$	$(3,\ t_1)$	$(5,\ t_2)$	$(2,\ t_3)$	$(3,\ t_3)$	$(5,\ t_2)$	$(1,\ t_3)$
u_3	$(4,\ t_1)$	$(5,\ t_1)$	$(3,\ t_1)$	$(4,\ t_1)$	$(3,\ t_5)$	$(2,\ t_5)$	$(1,\ t_5)$	$(1,\ t_5)$
u_4	$(5,\ t_1)$	$(4,\ t_2)$	$(5,\ t_2)$	$(4,\ t_4)$	$(3,\ t_5)$	$(3,\ t_4)$	$(1,\ t_5)$	$(3,\ t_2)$

periods. The *time entropy* can measure the confusion of timestamp, and the higher score, the better. It implies how much the user likes the tag.

Table 1 shows a user-item matrix with four users raking on eight items. The $u_i(i = 1, 2, 3, 4)$ are users, and $i_k(k = 1, 2, \ldots, 8)$ are items. The user's score between 1 to 5. We can list the number of visits and scores of the tags.

In Table 2, there are five time intervals. For the user u_1, his time cluster has three records that can be get from Fig. 2. The u_2 also has three records. But u_1 and u_2 who is better? Their *time entropy* are

$$H_{u_1} = -\frac{3.5 * 2}{6}\log\frac{2}{6} - \frac{3 * 1}{6}\log\frac{1}{6} - \frac{3 * 3}{6}\log\frac{3}{6} = 3.22 \tag{11}$$

$$H_{u_2} = -\frac{2.5 * 2}{6}\log\frac{2}{6} - \frac{1 * 1}{6}\log\frac{1}{6} - \frac{2 * 3}{6}\log\frac{3}{6} = 1.91. \tag{12}$$

It shows that u_1 likes the tag_1 more than u_2 due to $H_{u_1} > H_{u_2}$. This is consistent with the facts that the u_1 gives a higher score than u_2 on the tag_1.

4.2 Personalized Recommendation

We can not specify the user's interest information and can not effectively calculate the rules of their interest for users with more extensive interest in type of users. Due to the recommended algorithm can not make up for the system cold start problem for the new users and not clear their interest information at the same time. In view of the two problems mentioned above, we use diversity recommendation algorithms to recommend users to solve the system cold start problems and explore the user's diverse interests.

The diversity recommendations attract a lot of attention as a way to improve the quality of recommendations by reconciling the similarities and dissimilarities of items in the item list. A natural standard measure of diversity is to maximize the disparate sum of the items. Therefore, for a given item, we suggest that its diversity be defined as follows:

$$dv(i) = \frac{fr(i)}{\alpha \cdot k(i) + 1} \tag{13}$$

TABLE 2
Time Distribution of Four Users on tag_1

users	scores with time
u_1	$(3.5,\ 2,\ t_1)$ $(3,\ 1,\ t_2)$ $(3,\ 3,\ t_3)$
u_2	$(2.5,\ 2,\ t_1)$ $(1,\ 1,\ t_2)$ $(2,\ 3,\ t_3)$
u_3	$(3,\ 3,\ t_1)$ $(2,\ 3,\ t_5)$
u_4	$(5,\ 1,\ t_1)$ $(4,\ 3,\ t_2)$ $(3,\ 1,\ t_4)$ $(3,\ 1,\ t_4)$

$$k(i) = \sum_{j \in Set_B} sim(i, j) \tag{14}$$

$$sim(i, j) = \frac{\sum_{u \in U}(r_{u,i} - \overline{r_i})(r_{u,j} - \overline{r_j})}{\sqrt{\sum_{u \in U}(r_{u,i} - \overline{r_i})^2}\sqrt{\sum_{u \in U}(r_{u,j} - \overline{r_j})^2}}. \tag{15}$$

The equation is inspired by [52]: (1) Maximize the sum of the correlation and dissimilarity of a set to maximize diversity; (2) We combine the item of relevance and dissimilarity by using the single-objective formula similar to ours.

Algorithm 3. PTCCF Model

Input: *seting*: parameters setting; *data*: user-item raking matrix; u: recommended user;
Output: *results*: top-N recommended items;
$(k, N) \leftarrow$ **Initialize**$(seting)$;
$clusters \leftarrow$ **clusterUsers**$(data, k)$ by CSK-means;
if $u \in Pattern1$ **then**
  $neighbors \leftarrow$ **getNeighbors**$(u, clusters)$;
  $results \leftarrow$ **recommendItems**$(neighbors, data, N)$ as Eq. (2);
end
else if $u \in Pattern2$ **then**
  $neighbors \leftarrow$ **getNeighbors**$(u, clusters)$;
  $results \leftarrow$ **recommendItems**$(neighbors, data, N)$ as Eq. (2), Eqs. (13)∼(15);
end
else
  $results \leftarrow$ **recommendItems**$(data, N)$ as Eqs. (13)∼(15);
end
Return$(results)$;

Note that, in general, the recommended approach for diversification is to: (1) weigh the similarity between user recommendation list and given user evaluation list; (2) dissimilarity between items in using the similarity measure. Therefore, there is no significant comparability between local importance and classification similarity.

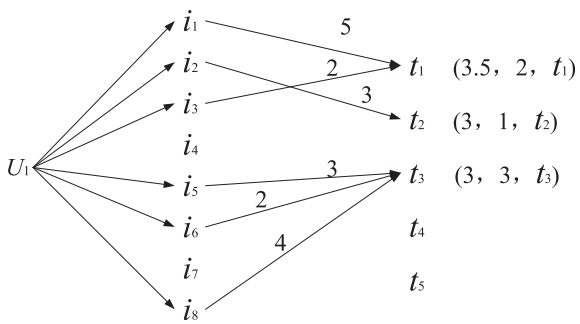


Fig. 2. Illustration of time distribution of u_1 on tag_1.

The input of PTCCF (Algorithm 3) is: cluster number k, recommended number of items N, user-item raking matrix data and other parameters for CS. First, users are clustered based on CSK-means (line 2). Then analyze the user's pattern, if the target user belongs to the pattern 1 and has records, find his neighbors, and recommend the neighbor's favorite items to him (lines 3 $\sim$ 5). Similarly, if the user belongs to pattern 2, he is recommended according to lines 8 $\sim$ 9. In other cases (e.g., new users without records), recommend popular labels (line 12).

5 Experimental Evaluation

To evaluate the effectiveness of our TCCF and PTCCF, we performed experiments on two real datasets of MovieLens [16] and Douban.

5.1 Dataset and Experimental Setting

5.1.1 Data Description

To verify the effectiveness of our optimization strategy on different datasets, we performed experiments on two datasets respectively. One is MovieLens-100M, which contains 71,567 user ratings of 10,000,054 for 10,681 movies. The other dataset is our crawling data from Douban.com, where each user can score from 1 to 5 for books, music, or movies while providing a rating similar to Facebook Social Relationship Service, which allows users to discover their own friends by E-mail. The data information we crawl from the website includes the user's rating information, user tag information, number of users and the number of products on the books. Since the original dataset contains more noise data and can not be directly used in the experiment, we need to de-noising the original data, such as selecting data records with user rating time and movie release time greater than 0 and selecting 10 users movies and users who have rated 10 movies. In the meantime, we set the ratio between the training dataset and the test dataset is 8:2 then verify the efficiency of our model.

5.1.2 Parameter Setting

In the experiment, to reduce the computational complexity of proposed model, we set the parameters of the algorithm through experiments. For the CS, the population size is 50, pa = 0.25, and the number of independent runs is 10.

5.1.3 Evaluation Metric

We employ the $recall$, $precision$, $F-measure$, MAP (Mean Average Precision) and MAE to evaluate the experimental results. Their formulas are shown as follows:

$$R = \frac{|N(i) \cap M(i)|}{|M(i)|} \tag{16}$$

$$P = \frac{|N(i) \cap M(i)|}{|N(i)|} \tag{17}$$

$$F - measure = \frac{2 * P * R}{P + R}, \tag{18}$$

where, N is the num of recommended items to the target user u, $M(i)$ is the the num of favorite items for user u.

For MAE, the smaller, the better recommended quality. Assuming that the predicted user score set is $\{p_1, p_2, \ldots, p_N\}$, and the corresponding actual score set is $\{q_1, q_2, \ldots, q_N\}$, the MAE can be given as follows:

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N}. \tag{19}$$

The MAP is the average value of the prediction accuracy, which can be given as follows:

$$MAP = \frac{1}{n} \sum_{i=1}^{n} precision_i, \tag{20}$$

where n represents the number of experiment. In addition, we employ the $novelty$ in [40] to verify the ability of our algorithm for new users. It can be given by:

$$novelty = Num/N, \tag{21}$$

where, Num is the number of all of the items of top-N recommendation list.

5.2 Experimental Results

To validate the effectiveness and efficiency of our models TCCF and PTCCF, we designed the following experimental verifications: (1) Optimal parameters for TCCF and PTCCF, mainly including the number of cluster k and σ for TCC; (2) Comparison with other collaborative filtering recommendation algorithms including the Probabilistic Matrix Factorization (PMF) [53], Bayesian Probabilistic Matrix Factorization (BPMF) [54], SVD++ [55] and MCoC [3]; (3) Parallelization based on Spark [56] for Big Data applications.

5.2.1 Effect of Parameters

a. Effect of Parameter K

First, we optimal the number of cluster k when $\sigma = 1$. The different clustering factors is set to compare the MAE and MAP of the algorithm. Using MovieLens and DouBan datasets for experiments in Figs. 3 and 4, respectively.

As we can see from Fig. 3, the MAE value of TCCF declines fastest with the increase of clustering factors at the beginning of the experiment. The MAE value of TCCF is the smallest and the recommended result is the best when the clustering factor is 4. In Fig. 3a and 3c, the MAE tends to decrease first and then increase with the increase of clustering factors. This also verifies that the relative accuracy of the target users' neighborhood aggregation is relatively low as the clustering factor increases. We can also see that the MAP has the largest value compared with KCF, CSKCF and TKCF, which also demonstrates that our proposed algorithm outperforms other algorithms with the increase of clustering factors from Fig. 3b and 3d.

b. Effect of Parameter σ

We know that the MAE value of the algorithm reaches the minimum when the number of clusters is 4 according to the above. Based on this, we test the time-dependent attenuation coefficient σ of the project, the values of σ are 0.1, 0.5, 1, 2, 4 and 8 respectively (Fig. 4).

We can see that the MAE value is the smallest and the value of MAP reaches the maximum when the value of σ is 1 from Fig. 4. Since σ is the time-dependent attenuation coefficient of

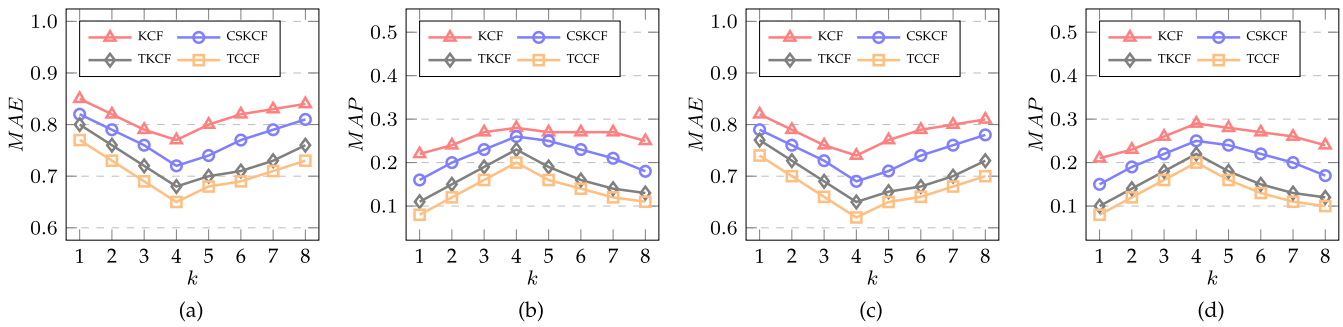


Fig. 3. Performance on different K when $\sigma = 1$ on MovieLens and Douban.

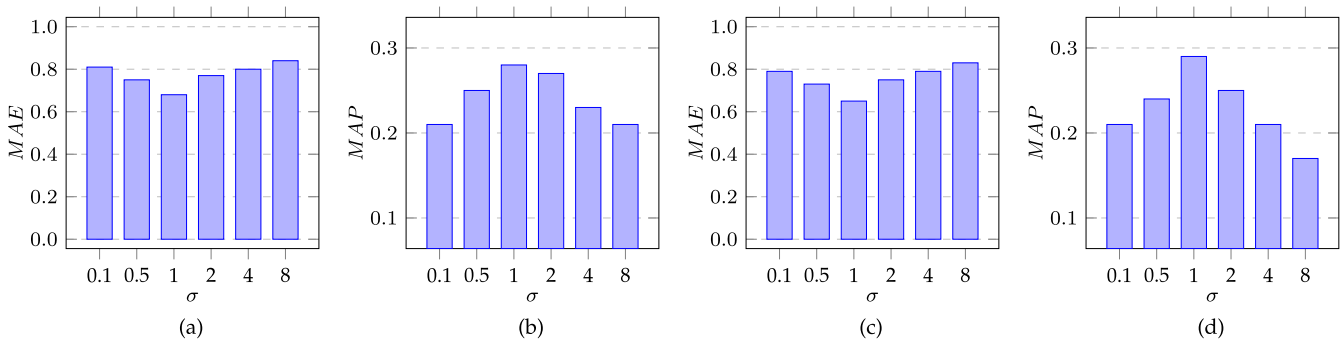


Fig. 4. Performance on different σ when $K = 4$ on MovieLens and Douban.

TABLE 3
Performance Comparisons With PMF, BPMF, SVD++ and MCoC on MovieLens and Douban

Dataset	Recommended method	N=10			N=20			N=50		
		P	F-measure	F	P	F-measure	F	P	F-measure	P
MovieLens	PMF	0.074	0.042	0.095	0.074	0.061	0.033	0.052	0.027	0.0679
	BPMF	0.102	0.053	0.11	0.102	0.093	0.049	0.086	0.041	0.087
	SVD++	0.269	0.159	0.259	0.269	0.258	0.148	0.249	0.137	0.241
	MCoC	0.275	0.145	0.256	0.275	0.267	0.137	0.261	0.126	0.245
	PTCCF	0.293	0.167	0.281	0.293	0.284	0.151	0.272	0.136	0.263
Douban	PMF	0.08	0.044	0.098	0.073	0.041	0.089	0.067	0.037	0.072
	BPMF	0.108	0.055	0.105	0.101	0.052	0.104	0.093	0.045	0.098
	SVD++	0.275	0.161	0.271	0.268	0.158	0.253	0.251	0.154	0.247
	MCoC	0.281	0.147	0.263	0.274	0.144	0.25	0.268	0.142	0.264
	PTCCF	0.297	0.169	0.297	0.292	0.166	0.281	0.286	0.162	0.291

the item, the larger σ means the smaller the temporal correlation of the item and the lower the recommendation accuracy. However, this does not mean that the smaller the value of σ is better. The time-dependent attenuation coefficient of the item is reduced when the value of σ is 0.1, so that the recommendation accuracy is reduced. Therefore, it is especially important to choose the appropriate value of σ.

5.2.2 Comparison With Other CF Models

To verify the effectiveness of our PTCCF, we compare the accuracy, F-measure and MAP of PMF, BPMF, SVD++, MCoC and PTCCF in this paper. The statistical information is shown in Table 3.

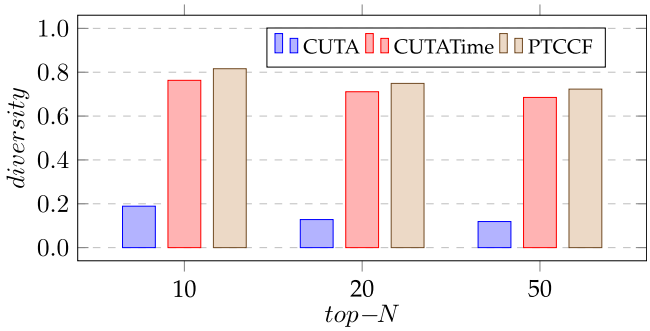
The accuracy of PTCCF is better than that of other algorithms as we can see from Table 3. The accuracy of our PTCCF model is also reduced with the increase of the number of recommended lists.

In view of the diversity recommendation, this paper also carried out relevant experiments and compared the diversity of the algorithm and the other algorithms under different recommended lists (Fig. 5).

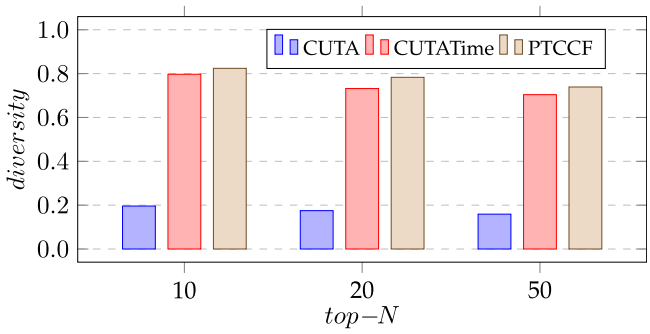
As we can see, the diversity of our PTCCF model is much better than the CUTA and CUTATime algorithm from Fig. 5. This also shows the effectiveness and novelty of this kind of recommendation for different class users.

5.2.3 Algorithm Parallelization

This paper uses the Spark distributed computing framework to implement the algorithm to compare the operating efficiency of our PTCCF model. Several collaborative filtering



(a) *MovieLens*



(b) *Douban*

Fig. 5. Diversity comparisons of CUTA [57], CUTATime and PTCCF on MovieLens and Douban.

recommendation algorithms are executed in standalone environment and their corresponding algorithms are parallelized in Spark environment to compare the running time of the algorithm (Fig. 6).

The advantages of parallel algorithms can not be well reflected in the small size of the dataset as can be seen from Fig. 6. But the advantage of Spark parallelization is becoming more and more obvious with the increasing scale of data sets. This fully reflects the huge advantage of Spark memory-based computing. Meanwhile, the running time of the proposed PTCCF is smallest when the data set size is relatively large, which fully verifies the efficiency of the proposed model.

6 Conclusion

This paper proposed a novel collaborative filtering algorithm based on time correlation coefficient (TCC) and CSK-means (TCCF). TCCF use CSK-means algorithm to separate big data problem into several smaller manageable problems. The clustering method is a pre-processing that cluster similar users together for further quick and accurate recommendation. First, we employed a novel intelligent optimization algorithm, Cuckoo search, to optimize the K-means algorithm to improve the clustering effect. Then, we designed a time factor to resolve the interest drift over time. Finally, we designed an effective and personalized recommendation model based on preference pattern (PTCCF) to improve the quality of TCCF. It can provide a higher quality recommendation by analyzing the user's behaviors. Systematic experimental results on MovieLens and Douban demonstrate that our proposed models TCCF and PTCCF are effective and efficient for a fast and accurate recommendation.

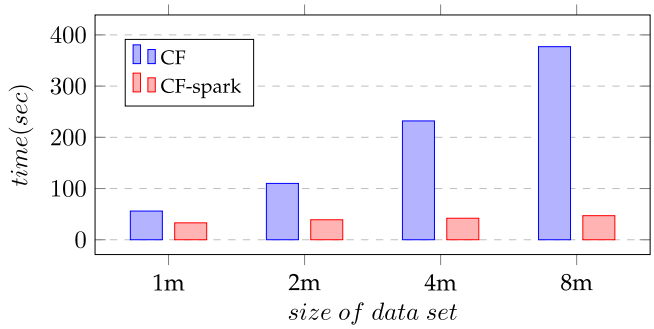


Fig. 6. Comparisons of the recommendation performance.

The PTCCF model, based on the time entropy method, can only analyze the user's pattern. However, the user behavior pattern also obeys more accurate models, such as the Gauss model, the cosine model. Therefore, in our future work, we will try to introduce more data mining technologies to learn users' model.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (Grant No. 2018YFC1604000), National Natural Science Foundation of China (Grant Nos. 61806138, U1636220, 61961160707, 61976212), Key R&D program of Shanxi Province (International Cooperation, Grant No. 201903D421048), and Key R&D program of Shanxi Province (High Technology, Grant No. 201903D121119).

References

[1] R. Ranjan *et al.*, "The next grand challenges: Integrating the Internet of Things and data science," *IEEE Cloud Comput.*, vol. 5, no. 3, pp. 12–26, May/Jun. 2018.

[2] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *ACM Comput. Surv.*, vol. 47, no. 1, 2014, Art. no. 3.

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

[4] W.-S. Hwang, J. Parc, S.-W. Kim, J. Lee, and D. Lee, ""Told you I didn't like it": Exploiting uninteresting items for effective collaborative filtering," in *Proc. IEEE 32nd Int. Conf. Data Eng.*, 2016, pp. 349–360.

[5] S. Wang, S. Huang, T.-Y. Liu, J. Ma, Z. Chen, and J. Veijalainen, "Ranking-oriented collaborative filtering: A listwise approach," *ACM Trans. Inf. Syst.*, vol. 35, no. 2, 2016, Art. no. 10.

[6] X. Wu, B. Cheng, and J. Chen, "Collaborative filtering service recommendation based on a novel similarity computation method," *IEEE Trans. Services Comput.*, vol. 10, no. 3, pp. 352–365, May/Jun. 2017.

[7] Q. Lu, F. Guo, and R. Zhang, "User-based collaborative filtering recommendation method combining with privacy concerns intensity in mobile commerce," *Int. J. Wireless Mobile Comput.*, vol. 17, no. 1, pp. 63–70, 2019.

[8] C.-D. Wang, Z.-H. Deng, J.-H. Lai, and S. Y. Philip, "Serendipitous recommendation in E-commerce using innovator-based collaborative filtering," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2678–2692, Jul. 2019.

[9] Q. Zhang, L. Cao, C. Zhu, Z. Li, and J. Sun, "CoupledCF: Learning explicit and implicit user-item couplings in recommendation for deep collaborative filtering," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 3662–3668.

[10] F. Xue, H. Tang, Q. Su, and T. Li, "Task allocation of intelligent warehouse picking system based on multi-robot coalition," *KSII Trans. Internet Inf. Syst.*, vol. 13, no. 7, pp. 3566–3582, 2019.

[11] B. Smith and G. Linden, "Two decades of recommender systems at Amazon.com," *IEEE Internet Comput.*, vol. 21, no. 3, pp. 12–18, May/Jun. 2017.

[12] R. Irfan *et al.*, "MobiContext: A context-aware cloud-based venue recommendation framework," *IEEE Trans. Cloud Comput.*, vol. 5, no. 4, pp. 712–724, Oct.–Dec. 2017.

[13] D. Hussein, S. N. Han, G. M. Lee, and N. Crespi, "Social cloud-based cognitive reasoning for task-oriented recommendation," *IEEE Cloud Comput.*, vol. 2, no. 6, pp. 10–19, Nov./Dec. 2015.

[14] A. Khoshkbarforoushha, R. Ranjan, R. Gaire, E. Abbasnejad, L. Wang, and A. Y. Zomaya, "Distribution based workload modelling of continuous queries in clouds," *IEEE Trans. Emerg. Topics Comput.*, vol. 5, no. 1, pp. 120–133, Jan.–Mar. 2017.

[15] A. Khoshkbarforoushha, A. Khosravian, and R. Ranjan, "Elasticity management of streaming data analytics flows on clouds," *J. Comput. Syst. Sci.*, vol. 89, pp. 24–40, 2017.

[16] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An open architecture for collaborative filtering of netnews," in *Proc. ACM Conf. Comput. Supported Cooperative Work*, 1994, pp. 175–186.

[17] W. Zhang and J. Wang, "A collective Bayesian poisson factorization model for cold-start local event recommendation," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1455–1464.

[18] L. Wang, Y. Liu, and J. Wu, "Research on financial advertisement personalised recommendation method based on customer segmentation," *Int. J. Wireless Mobile Comput.*, vol. 14, no. 1, pp. 97–101, 2018.

[19] Z. Cui, Y. Cao, X. Cai, J. Cai, and J. Chen, "Optimal leach protocol with modified bat algorithm for big data sensing systems in Internet of Things," *J. Parallel Distrib. Comput.*, vol. 132, pp. 217–229, 2019.

[20] J. Liu, M. Tang, Z. Zheng, X. F. Liu, and S. Lyu, "Location-aware and personalized collaborative filtering for web service recommendation," *IEEE Trans. Services Comput.*, vol. 9, no. 5, pp. 686–699, Sep./Oct. 2016.

[21] L. Yao, Q. Z. Sheng, A. H. Ngu, J. Yu, and A. Segev, "Unified collaborative and content-based web service recommendation," *IEEE Trans. Services Comput.*, vol. 8, no. 3, pp. 453–466, May/Jun. 2015.

[22] H. Chen and J. Li, "Learning multiple similarities of users and items in recommender systems," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 811–816.

[23] W. Pan and L. Chen, "CoFiSet: Collaborative filtering via learning pairwise preferences over item-sets," in *Proc. SIAM Int. Conf. Data Mining*, 2013, pp. 180–188.

[24] H. Zhao, Q. Yao, J. T. Kwok, and D. L. Lee, "Collaborative filtering with social local models," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 645–654.

[25] X. Cai, P. Wang, L. Du, Z. Cui, W. Zhang, and J. Chen, "Multi-objective three-dimensional DV-hop localization algorithm with NSGA-II," *IEEE Sensors J.*, vol. 19, no. 21, pp. 10 003–10 015, Nov. 2019.

[26] Z. Cui *et al.*, "A pigeon-inspired optimization algorithm for many-objective optimization problems," *Sci. China Inf. Sci.*, vol. 62, pp. 070 212:1–070 212:3, 2019.

[27] X. Cai *et al.*, "An under-sampled software defect prediction method based on hybrid multi-objective cuckoo search," *Concurrency Comput., Practice Experience*, 2019, Art. no. e5478. [Online]. Available: https://doi.org/10.1002/cpe.5478

[28] X. Cai, M. Zhang, H. Wang, M. Xu, J. Chen, and W. Zhang, "Analyses of inverted generational distance for many-objective optimisation algorithms," *Int. J. Bio-Inspired Comput.*, vol. 14, no. 1, pp. 62–68, 2019.

[29] N. E. I. Karabadji, S. Beldjoudi, H. Seridi, S. Aridhi, and W. Dhifli, "Improving memory-based user collaborative filtering with evolutionary multi-objective optimization," *Expert Syst. Appl.*, vol. 98, pp. 153–165, 2018.

[30] A. Paterek, "Improving regularized singular value decomposition for collaborative filtering," in *Proc. KDD Cup Workshop*, 2007, pp. 5–8.

[31] I. Barjasteh, R. Forsati, D. Ross, A.-H. Esfahanian, and H. Radha, "Cold-start recommendation with provable guarantees: A decoupled approach," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 6, pp. 1462–1474, Jun. 2016.

[32] M. Sharma, J. Zhou, J. Hu, and G. Karypis, "Feature-based factorized bilinear similarity model for cold-start top-n item recommendation," in *Proc. SIAM Int. Conf. Data Mining*, 2015, pp. 190–198.

[33] Z. Zhou, Z. Cheng, L.-J. Zhang, W. Gaaloul, and K. Ning, "Scientific workflow clustering and recommendation leveraging layer hierarchical analysis," *IEEE Trans. Services Comput.*, vol. 11, no. 1, pp. 169–183, Jan./Feb. 2018.

[34] R. Hu, W. Dou, and J. Liu, "ClubCF: A clustering-based collaborative filtering approach for big data application," *IEEE Trans. Emerg. Topics Comput.*, vol. 2, no. 3, pp. 302–313, Sep. 2014.

[35] N. Mirbakhsh and C. X. Ling, "Leveraging clustering to improve collaborative filtering," *Inf. Syst. Front.*, vol. 20, no. 1, pp. 111–124, 2018.

[36] S. Zahra, M. A. Ghazanfar, A. Khalid, M. A. Azam, U. Naeem, and A. Prugel-Bennett, "Novel centroid selection approaches for Kmeans-clustering based recommender systems," *Inf. Sci.*, vol. 320, pp. 156–189, 2015.

[37] A. Salah, N. Rogovschi, and M. Nadif, "A dynamic collaborative filtering system via a weighted clustering approach," *Neurocomputing*, vol. 175, pp. 206–215, 2016.

[38] F. Liu and H. J. Lee, "Use of social network information to enhance collaborative filtering performance," *Expert Syst. Appl.*, vol. 37, no. 7, pp. 4772–4778, 2010.

[39] D. Lian, Z. Zhang, Y. Ge, F. Zhang, N. J. Yuan, and X. Xie, "Regularized content-aware tensor factorization meets temporal-aware location recommendation," in *Proc. IEEE 16th Int. Conf. Data Mining*, 2016, pp. 1029–1034.

[40] H. Yu and J. Li, "Algorithm to solve the cold-start problem in new item recommendations," *J. Softw.*, vol. 26, no. 6, pp. 1395–1408, 2015.

[41] G. Guo, F. Zhu, S. Qu, and X. Wang, "PCCF: Periodic and continual temporal co-factorization for recommender systems," *Inf. Sci.*, vol. 436, pp. 56–73, 2018.

[42] Y. Koren, "Collaborative filtering with temporal dynamics," *Commun. ACM*, vol. 53, no. 4, pp. 89–97, 2010.

[43] B. Li, X. Zhu, R. Li, C. Zhang, X. Xue, and X. Wu, "Cross-domain collaborative filtering over time," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 2293–2298.

[44] X.-S. Yang and S. Deb, "Cuckoo search via lévy flights," in *Proc. World Congr. Nature Biologically Inspired Comput.*, 2009, pp. 210–214.

[45] Z. Cui, B. Sun, G. Wang, Y. Xue, and J. Chen, "A novel oriented cuckoo search algorithm to improve DV-hop performance for cyber–physical systems," *J. Parallel Distrib. Comput.*, vol. 103, pp. 42–52, 2017.

[46] Z. Cui, F. Xue, X. Cai, Y. Cao, G.-G. Wang, and J. Chen, "Detection of malicious code variants based on deep learning," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3187–3196, Jul. 2018.

[47] Pooja, P. Chaturvedi, P. Kumar, and A. Tomar, "A novel differential evolution approach for constraint optimisation," *Int. J. Bio-Inspired Comput.*, vol. 12, no. 4, pp. 254–265, 2018.

[48] Y. Cao, Z. Ding, F. Xue, and X. Rong, "An improved twin support vector machine based on multi-objective cuckoo search for software defect prediction," *Int. J. Bio-Inspired Comput.*, vol. 11, no. 4, pp. 282–291, 2018.

[49] R. S. Parpinelli, G. Plichoski, R. S. Da Silva, and P. H. Narloch, "A review of techniques for online control of parameters in swarm intelligence and evolutionary computation algorithms," *Int. J. Bio-Inspired Comput.*, vol. 13, no. 1, pp. 1–20, 2019.

[50] Z. Cui, Y. Chang, J. Zhang, X. Cai, and W. Zhang, "Improved NSGA-III with selection-and-elimination operator," *Swarm Evol. Comput.*, vol. 49, pp. 23–33, 2019.

[51] K. Krishna and M. N. Murty, "Genetic K-means algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 3, pp. 433–439, Jun. 1999.

[52] G. J. Fakas, Z. Cai, and N. Mamoulis, "Diverse and proportional size-l object summaries using pairwise relevance," *VLDB J.*, vol. 25, no. 6, pp. 791–816, 2016.

[53] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.

[54] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using Markov chain monte carlo," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 880–887.

[55] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 426–434.

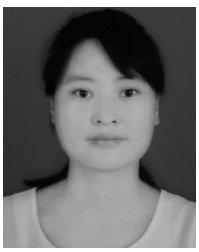
[56] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proc. 2nd USENIX Conf. Hot Topics Cloud Comput.*, 2010, Art. no. 95.

[57] C. Yu and L. Huang, "Time-aware collaborative filtering for QoS-based service recommendation," in *Proc. IEEE Int. Conf. Web Services*, 2014, pp. 265–272.

Zhihua Cui received the PhD degree in control theory and engineering from Xi'an Jiaotong University, Xi'an, China, in 2008. He is currently a professor with the School of Computer Science and Technology, Taiyuan University of Science and Technology, China. He is the editor-in-chief of the *International Journal of Bio-inspired Computation*. His research interests include computational intelligence, stochastic algorithm, and combinatorial optimization.

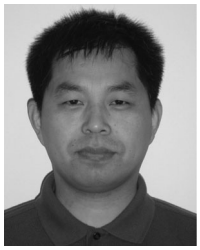
Xianghua Xu received the bachelor's degree from the Hangzhou Institute of Electronic Engineering, Hangzhou, China, and the PhD degree in computer science from Zhejiang University, Hangzhou, China. He is currently a professor with the School of Computer Science, Hangzhou Dianzi University, China. His research interests include pattern recognition, parallel and distributed computing, and IoT security. He has published more than 100 peer reviewed journal and conference papers. He has received Best Paper Award of IISWC2012.

Fei Xue received the MS degree in computer application technology from the Taiyuan University of Science and Technology, Taiyuan, China, in 2011, and the PD degree in computer science and technology from the Beijing University of Technology, Chaoyang, China, in 2016. He is currently a lecturer with the School of Information, Beijing Wuzi University. His research interests include swarm intelligence and network security.



Xingjuan Cai received the PhD degree in control theory and engineering from Tongji University, Zha Bei Qu, Shanghai, China, in 2017. She is currently an associate professor with the School of Computer Science and Technology, Taiyuan University of Science and Technology, China. Her interest includes bio-inspired computation and applications.

Yang Cao received the MS degree in computer science and technology from the Taiyuan University of Science and Technology, Taiyuan, China, in 2015, and the PD degree in computer science and technology from the Beijing University of Technology, Chaoyang, China, in 2019. He is currently a lecturer with the School of Information, Beijing Wuzi University. His research interests include swarm intelligence and big data analysis.



Wensheng Zhang received the PhD degree in pattern recognition and intelligent systems from the Institute of Automation, CAS, Beijing, China, in 2000. He is currently a professor of Machine Learning and Data Mining and the director of the Research and Development, Institute of Automation, Chinese Academy of Sciences (CAS). His research interests include computer vision, pattern recognition, artificial intelligence, and computer-human interaction.

Jinjun Chen is currently a professor with the Faculty of Science, Engineering and Technology, the deputy director of Swinburne Data Science Research Institute, Swinburne University of Technology, Australia. His research interests include big data, data systems, cloud computing, security, and privacy. He received various awards such as Vice Chancellor Research Award. He is a senior member of the IEEE.