Collaborative Filtering Service Recommendation Based on a Novel Similarity Computation Method

Xiaokun Wu, Bo Cheng, and Junliang Chen

Abstract— Recently, collaborative filtering-based methods are widely used for service recommendation. QoS attribute value-based collaborative filtering service recommendation includes two important steps. One is the similarity computation, and the other is the prediction for the QoS attribute value, which the user has not experienced. In some previous studies, the similarity computation methods and prediction methods are not accurate. The performances of some methods need to be improved. In this paper, we propose a ratio-based method to calculate the similarity. We can get the similarity between users or between items by comparing the attribute values directly. Based on our similarity computation method, we propose a new method to predict the unknown value. By comparing the values of a similar service and the current service that are invoked by common users, we can obtain the final prediction result. The performance of the proposed method is evaluated through a large data set of real web services. Experimental results show that our method obtains better prediction precision, lower mean absolute error (MAE) and faster computation time than various reference schemes considered.

Index Terms—Collaborative filtering, ratio-based, similarity computation, prediction, service recommendation.

1 Introduction

With the development of network technology and the large increase in the number of users, more and more services that have the same or similar function have arisen in networks. To identify optimal services, service users are required to spend significant amounts of time and to make great efforts to search and choose. It is difficult, time-consuming and ineffective. According to the users' personal preferences, historical records or similar users' information, recommending unused suitable services to the service users is attractive to both the users and the service providers. On one hand, the users are unable to expend much time or energy to experience many services that have the same or similar function. On the other hand, the appropriate recommendation may bring potential users or commercial interests to the providers.

Recently, collaborative filtering-based service recommendation methods have received extensive attention. Unlike content-based recommendation, a collaborative filtering method need not analyze the content of the item (here, the service) or extract the features of the content. It relies on users' evaluation for the items and other information to find similar users for the current user. Then, it recommends the items that similar users loved to the current user. Here, "similar users" are defined as those whose preferences or attributes are correlated with the current user. Usually, users' preferences are indicated in the form of a series of quantitative values.

Collaborative filtering assumes that if multiple users all like many of the same things, they will also concurrently like other goods. For example, a group of users have read a number of books and have provided similar evaluations for them. We know that these users have high similarity, thus, it is reasonable to recommend a book that these users loved to the one among them who has not read the book.

Service selection or recommendation must consider both the functional and the non-functional attributes of services. When their functional attributes are the same or similar, service recommendation must be based on their non-functional attributes, i.e., quality-of-service (QoS).

Generally, service QoS includes several attributes, such as response time, reliability, price, etc., which are given in the form of numerical values. Analyzing these attributes, we can see that some of them are determined by the service provider, such as the price of service, which is the cost the user must pay when he invokes the service. The others are determined by the user, such as delay. This attribute objectively depends on the distance between the user and the service, as well as the the network status.

According to the QoS attributes, i.e., whether they depend on the user and whether that dependence is subjective or objective, we divide the attributes into three categories. Table 1 presents these classifications.

QoS attribute value-based collaborative filtering service recommendation usually contains several steps, among which two steps are more important. The first is to compute the similarity between users or between services. The second is to predict the QoS attribute values, that the user has not experienced, according to similar users or(and) similar services.

Presently, the Pearson correlation coefficient (PCC) and cosine (COS) methods are commonly applied to calculate the similarity. However, these methods have limited accuracy. In [1, 2], two new methods have been proposed to compute similarity. However, these methods still have var-

- *Xiaokun Wu, Bo Cheng, and Junliang Chen are with the State Key Lab of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Haidian, Beijing 100876, China. Bo Cheng is the corresponding author of this paper.*
  E-mail:{wuxk, chengbo, chjl}@bupt.edu.cn.

Manuscript received month date, year; revised month date, year.

TABLE 1
Three types of service QoS attributes

Whether they depend on the user		Attribute classification	Examples
Do not depend on the user		The first type	Throughput, Price, Service's location
Depend on the user	Objectively depend on the users	The second type	Delay, Response time
	Subjectively depend on the users	The third type	Degree of user's satisfaction

ious problems. Their accuracy and complexity need to be improved. In this paper, we mainly focus on the memory-based collaborative filtering methods. Aiming at the two important steps of collaborative filtering recommendation, we propose two new methods. One is to compute the similarity and the other is to predict the unknown value.

The main contributions of this paper are as follows:

- We propose a ratio-based method to calculate the similarity. Comparing the attribute values directly, we can calculate the similarity between users or between items. This method can be applied to memory-based collaborative filtering as well as to other aspects.
- Based on the proposed similarity calculation method, we present a new method to conduct the prediction for unknown values.
- Experimental results show that: (1) Our method outperforms the reference schemes in the precision of prediction values. (2) Our method has the smallest mean absolute error (MAE) among the competing methods. (3) From the perspective of computation time, our method displays superiority over others, especially when the sparse user-item matrix has higher density.

The rest of this paper is organized as follows: In section 2, we give the related work. Section 3 introduces our novel similarity computation method. In section 4, we elaborate on our prediction method for unknown values. Section 5 states the service recommendation. The experiments and analyses for them are presented in section 6. Section 7 concludes this paper.

2 Related Work

There are several types of methods for service recommendation, such as context-based [11], trust-based [19], relationships and preferences-based [18], SVD [24, 25], tendencies-based [27], collaborative filtering-based, etc.

Collaborative Filtering (CF) is one of the widely used recommendation methods. Breese et al. [10] classified the CF methods into two categories: memory-based and model-based. The memory-based collaborative filtering methods include: user-based [10], item-based [22, 23], and their combination [31].

Recently, collaborative filtering methods for QoS-aware web service selection and recommendation have been studied in several researches [1, 2, 3, 10, 16, 17].

In CF-based service recommendation, similarity computation is an important step.

PCC [3] and COS [22] are the most commonly used methods to compute the similarity in memory-based CF. By adding a significance weight to the PCC method, Zheng et al. [2] proposed an enhanced PCC similarity computation method. Different from the PCC similarity measurement, Jiang et al. [17] suggested that, in the user similarity measurement, more popular services or services with more stable QoS from user to user should be given less consideration.

H. Sun et al. [1] analyzed the shortcomings of PCC and COS and proposed a Normal Recovery (NR) method to compute the similarity.

In section 3, using a specific example, we will demonstrate that PCC, COS and NR are not without fault.

Prediction for unknown values is another important step in CF-based service recommendation.

In [16], Zhang et al. advised that, when predicting QoS values, we should consider users' experiences, environmental factor and user input factor.

In [10], a user-based method (UPCC) is applied to predict the unknown values, and, in [22, 23], an item-based method (IPCC) is used. Zheng et al. [2] combine UPCC and the IPCC together to conduct the prediction. Additionally, they employ two confidence weights to balance the results from UPCC and IPCC. Based on the NR similarity measure method, Sun et al. [1] proposed the NRCF method to predict the unknown values.

Apart from these methods, UMEAN and IMEAN are other popular prediction methods. In section 4, we will detail them.

To improve the prediction accuracy, several methods have been proposed, such as cluster-based smoothing method [14], zero-sum reward and punishment mechanism [32], etc.

3 Similarity Computation

3.1 Existing Similarity Computation Methods

In this section, we introduce the widely used similarity computation methods in memory-based collaborative filtering. We assume the similarity computation is based on a user-item matrix. It is composed of values of M users invoking N items. For simplicity, we specify the items as services.

Presently, in the collaborative filtering-based recommendation, there are two main types of similarity computation methods: PCC and COS.

When PCC is applied in the user-based collaborative filtering algorithm, we can calculate the similarity between two users. The similarity calculation between users u and v can be expressed as:

$$Sim\left(u,v\right)=\frac{\sum\limits_{i\in I}\left(r_{u,i}-\overline{r}_{u}\right)\left(r_{v,i}-\overline{r}_{v}\right)}{\sqrt{\sum\limits_{i\in I}\left(r_{u,i}-\overline{r}_{u}\right)^{2}}\sqrt{\sum\limits_{i\in I}\left(r_{v,i}-\overline{r}_{v}\right)^{2}}} \tag{1}$$

Here, $I=I_u\bigcap I_v$ is the set of items that are invoked by user u and v together; $r_{u,i}$ is the value of item i invoked by user u; and $\overline{r}_u$ denotes the average value that user u invokes all of the items in I. From the equation, we can know that $-1\leq Sim\left(u,v\right)\leq 1$. The greater the value of $Sim(u,v)$ is, the higher the similarity between users u and v. If $I=\emptyset$, we cannot calculate the similarity between users u and v.

When PCC is applied in the item-based collaborative filtering algorithm, we can calculate the similarity between two items. The similarity calculation between items i and j is given below:

$$Sim\left(i,j\right)=\frac{\sum\limits_{u\in U}\left(r_{u,i}-\overline{r}_{i}\right)\left(r_{u,j}-\overline{r}_{j}\right)}{\sqrt{\sum\limits_{u\in U}\left(r_{u,i}-\overline{r}_{i}\right)^{2}}\sqrt{\sum\limits_{u\in U}\left(r_{u,j}-\overline{r}_{j}\right)^{2}}} \tag{2}$$

Here, $U=U_i\bigcap U_j$ is the set of users who invoked both items i and j; $r_{u,i}$ denotes the value when user u invokes item i; and $\overline{r}_i$ is the average value when all of the users in U invoke item i. Similarly, $Sim(i,j)$ is also in the interval of [-1,1].

In some accidental circumstances, two users or items are not actually similar, but they happen to have several close values in the user-item matrix. In this case, equations (1) and (2) will yield incorrect similarity results. To overcome this type of error brought about by coincidence factors, an enhanced PCC method was proposed in [2]. They employ a *significance weight* to reduce the influence caused by coincidence factors.

The COS method is also commonly used for the similarity calculation between users or between items. The calculation of similarity between users is shown in equation (3) and between items is shown in equation (4).

$$Sim\left(u,v\right)=\frac{\sum\limits_{i\in I}r_{u,i}r_{v,i}}{\sqrt{\sum\limits_{i\in I}r_{u,i}^{2}}\sqrt{\sum\limits_{i\in I}r_{v,i}^{2}}} \tag{3}$$

$$Sim\left(i,j\right)=\frac{\sum\limits_{u\in U}r_{u,i}r_{u,j}}{\sqrt{\sum\limits_{u\in U}r_{u,i}^{2}}\sqrt{\sum\limits_{u\in U}r_{u,j}^{2}}} \tag{4}$$

Here, $I=I_u\bigcap I_v$ is the set of items invoked by users u and v, and $U=U_i\bigcap U_j$ is the set of users who invoked both items i and j.

Although PCC and COS can calculate the similarity, they also have many shortcomings. In [1], these problems are analyzed. The authors believe that the PCC method does not take the differences of QoS attributes values given by different users into account. Although the COS method can measure the angles of the vectors, which are composed by the users or services, it neglects the lengths of the vectors.

To overcome these shortcomings, they have proposed a new similarity computation method. They normalize the values of users invoking items to the same range and then unify the similarity of the scaled user (or item) vectors in different multidimensional vector spaces.

The similarity between users u and v is:

$$Sim\left(u,v\right)=1-\frac{\sqrt{\sum\limits_{i\in I}\left(\frac{r_{u,i}-r_{u\,\min}}{r_{u\,\max}-r_{u\,\min}}-\frac{r_{v,i}-r_{v\,\min}}{r_{v\,\max}-r_{v\,\min}}\right)^{2}}}{\sqrt{|I|}} \tag{5}$$

Here, $I=I_u\bigcap I_v$ is the set of items that both users u and v have invoked; $|I|$ is the number of items in I; $r_{u\,\max}$ and $r_{u\,\min}$ are the maximum and minimum values that user u has obtained when he invoked all of the items; and $r_{u,i}$ denotes the value of user u invoking item i.

In [1], the similarity between items i and j is:

$$Sim\left(i,j\right)=1-\frac{\sqrt{\sum\limits_{u\in U}\left(\frac{r_{u,i}-r_{i\,\min}}{r_{i\,\max}-r_{i\,\min}}-\frac{r_{u,j}-r_{j\,\min}}{r_{j\,\max}-r_{j\,\min}}\right)^{2}}}{\sqrt{|U|}} \tag{6}$$

Here, $U=U_i\bigcap U_j$ denotes the users who invoked both items i and j; $|U|$ is the number of users in U; $r_{i\,\max}$ and $r_{i\,\min}$ are the maximum and minimum values of item i when all of the users invoke it; and $r_{u,i}$ is the value when user u invokes item i.

The range of both $Sim(u,v)$ and $Sim(i,j)$ is [0, 1]. In this space, a score of 1 indicates the users or items have the highest similarity, and a score of 0 indicates they are not similar at all.

Equations (5) and (6) can solve the distribution deviation problem caused by different users who invoke the same item. However, there are still some problems with this method. We take Table 2 as an example, which gives the values of 7 users invoking 5 items.

TABLE 2
The values of 7 users invoking 5 items

	i_1	i_2	i_3	i_4	i_5
u_1	1	1	1	1	2
u_2	1	6	1	3	4
u_3	2	2	2	2	3
u_4	2	2		4	5
u_5	3		3	5	
u_6	4	4	4	4	5
u_7	5	5	5	5	6

In the table, $u_1 ... u_7$ are the users, and $i_1 ... i_5$ are the items. For instance, we can regard these items as services, and the values in the table are the response time when the users invoke these services. A null value represents that the user does not invoke the item.

According to equation (5) , we can obtain $Sim(u_1,u_7)$ = 1, $Sim(u_3,u_7)$ = 1, $Sim(u_6,u_7)$ = 1, $Sim(u_2,u_5)$ = 1, and $Sim(u_4,u_5)$ = 1.

From these results, we can see that there are some unreasonable situations. The values of u_1 are close to the minimum, and the values of u_7 are close to the maximum, but their similarity $Sim(u_1, u_7) = 1$. Furthermore, the values of user u_6 differ considerably from those of u_1, but $Sim(u_6, u_7)$ is equal to $Sim(u_1, u_7)$. These results do not reflect the actuality very well, which shows that NR method has anomalous situation, also.

3.2 Proposed Similarity Computation Method

In this paper, we propose a new method to calculate the similarity.

Generally, the QoS attributes experienced by the user are given in the form of numerical values, and these values are non-negative. The similarity represents the degree of two objects' consistency. We can use the ratio of two values to express the consistency. The ratio of two attribute values which is the results of two users invoking the same item reflects the users' consistency on this item, i.e., the single similarity. Summing up all the single similarities together and getting the average, we can obtain the final similarity between two users.

Our method to measure the similarity between users u and v is:

$$Sim\,(u,v) = \frac{\sum\limits_{i\in I} \dfrac{\min(r_{u,i}, r_{v,i})}{\max(r_{u,i}, r_{v,i})}}{|I|} \tag{7}$$

Here, $r_{u,i}$ and $r_{v,i}$ are the values when users u and v invoke item i; $\min(r_{u,i}, r_{v,i})$ is the minimum of $r_{u,i}$ and $r_{v,i}$; $\max(r_{u,i}, r_{v,i})$ is the maximum of them; $I = I_u \bigcap I_v$ is the set of items that are invoked by both users u and v; and $|I|$ is the number of items in I.

To demonstrate the methodology, we assume the attribute value is the response time when a user invokes a service. Assume that users u and v invoke 5 services $i_1, i_2 ..., i_5$, and that user u obtains 5 values: $r_{u,i_1} = 5$, $r_{u,i_2} = 2.5$, $r_{u,i_3} = 7$, $r_{u,i_4} = 2$, $r_{u,i_5} = 4$, and user v obtains: $r_{v,i_1} = 2$, $r_{v,i_2} = 4$, $r_{v,i_3} = 3$, $r_{v,i_4} = 3$, $r_{v,i_5} = 3.5$. These values are shown in Table 3.

TABLE 3
The values of 2 users invoking 5 items

	i_1	i_2	i_3	i_4	i_5
u	5	2.5	7	2	4
v	2	4	3	3	3.5

As shown in Fig. 1, we depict these values in the coordinate. Fig. 1(a) expresses the values of user u, and Fig. 1(b) expresses those of user v. To compute the similarity between u and v, we overlay Fig. 1(a) and Fig. 1(b) to obtain Fig. 1(c). In Fig. 1(d), the shaded segment is composed of 5 minimum values. The ratio of minimum and maximum values at the same point indicates the consistency of two users for one item. The average of these ratios is the similarity between users u and v.

The value of $\frac{\min(r_{u,i}, r_{v,i})}{\max(r_{u,i}, r_{v,i})}$ is in the interval between 0 and 1, so $Sim(u, v) \in [0, 1]$, with higher $Sim(u, v)$ representing higher similarity between users u and v.

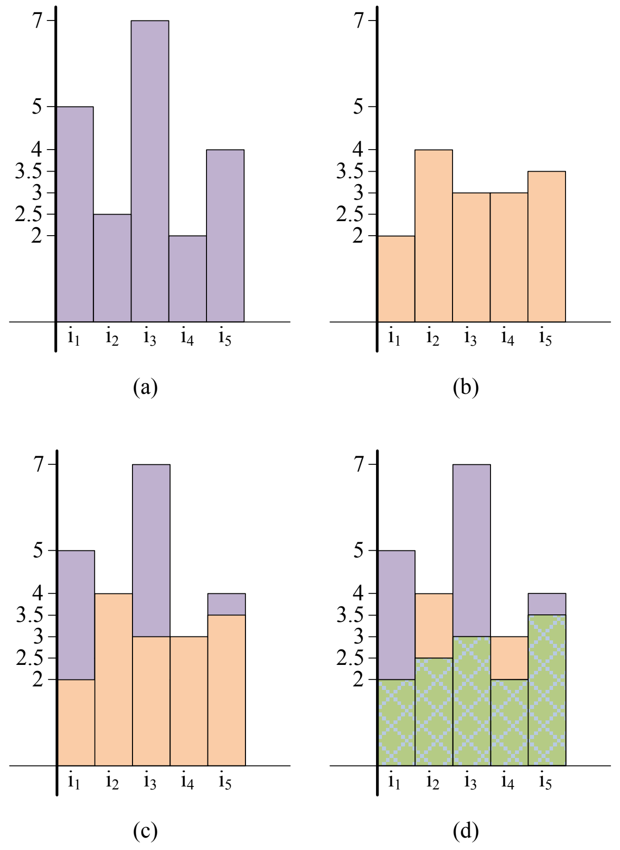


Fig. 1. The ratio-based similarity computation method.

In essence, our method for similarity computation is to compare the values, so it can be called ratio-based method (RA, for short).

Similarly, we can determine the similarity between items i and j:

$$Sim\,(i,j) = \frac{\sum\limits_{u\in U} \dfrac{\min(r_{u,i}, r_{u,j})}{\max(r_{u,i}, r_{u,j})}}{|U|} \tag{8}$$

Here, $r_{u,i}$ and $r_{u,j}$ represent the values obtained by user u when he invokes items i and j, respectively; $\min(r_{u,i}, r_{u,j})$ and $\max(r_{u,i}, r_{u,j})$ are the minimum and maximum of $r_{u,i}$ and $r_{u,j}$; $U = U_i \bigcap U_j$ is the set of users who invoke both items i and j; and $|U|$ is the number of elements in U.

Similar to $Sim(u, v)$, $Sim(i, j)$ is also in the interval of [0, 1].

Applying the RA method, we recalculate the similarity between the users in Table 2. We obtain: $Sim(u_1, u_7)$ = 0.2267, $Sim(u_3, u_7)$ = 0.42, $Sim(u_6, u_7)$ = 0.8067, $Sim(u_2, u_5)$ = 0.4222, and $Sim(u_4, u_5)$ = 0.7333. However, using the NR method (equation 5), all the results are equal to 1. The results of PCC, COS, NR and RA are shown in Table 4.

In Table 2, the values of u_1 are almost the minimum, and values of u_7 are close to the maximum. That is, u_1 differs considerably with u_7. However, when using the COS method, they have a very high similarity (0.9701). Similarly, the values of u_6 are nearly two times the values of u_3, but $Sim(u_3, u_7)$ and $Sim(u_6, u_7)$ are almost equal.

Similar to the NR method, when applying PCC, $Sim(u_1, u_7)$, $Sim(u_3, u_7)$ and $Sim(u_6, u_7)$ are equal to 1.

However, in Table 2, the values of u_1,u_3 and u_6 vary greatly.

These results from the competing methods do not correctly reflect the actual fact. Analyzing these methods, we can see they are more complex than RA. Furthermore, RA has clear meaning and is easy to understand.

TABLE 4
The comparison results of 4 methods

Similarity	PCC	COS	NR	RA
$Sim(u_1, u_7)$	1	0.9701	1	0.2267
$Sim(u_3, u_7)$	1	0.9947	1	0.4200
$Sim(u_6, u_7)$	1	0.9998	1	0.8067
$Sim(u_2, u_5)$	0.3849	0.4035	1	0.4222
$Sim(u_4, u_5)$	0.4321	0.5664	1	0.7333

As seen from Table 4, the results of RA are more reasonable.

In the previous section, we have divided the QoS attributes into three categories. Our method is applicable to all kinds of QoS attributes which are given in the numerical values. However, some of the qualitative and subjective QoS attributes are expressed in non-numerical value, such as “very good”, “good”, and so on. According to certain rules, these evaluations can be transformed into numerical values, and then our method can be used.

4 Prediction for Unknown Values

After calculating the similarity between users or items (services), the prediction for unknown values becomes another important step in the QoS attribute value-based collaborative filtering.

Service recommendation refer to introducing appropriate service(s) to a user who has not experienced it(them) previously. Generally, service QoS attributes that the users have experienced are presented in the form of numerical values. To recommend unused services to the user, it is necessary to predict the unknown values. According to these values, the recommendation system can execute the recommendation.

To facilitate the statement, we can define the item to be a service as was done above, and the value of the item is the response time.

Additionally, we give several definitions:

- *Current user,* the user who wants to invoke a service, and he requires the recommendation system to recommend service to him.
- *Current service,* the service that the current user has not invoked and he prepare to do so. This service is a recommendation candidate.
- *Predicted value,* the unknown value of the current user predicted for the current service.

Presently, several methods have been proposed to predict values. In this section, we will introduce the followings: UMEAN, IMEAN, UPCC, IPCC, WSRec and NRCF.

We assume the prediction is based on a user-item matrix, which is composed of the response time values of M users invoking N services. The current user is u and the current service is i. User u has not invoked service i. Now, it is required to predict the value of u invoking i.

4.1 UMEAN

UMEAN is a user-based prediction method. This method average the values that the current user invokes all of the other services (except for the current service i). UMEAN takes the average value as the prediction result.

4.2 IMEAN

Different from UMEAN, the IMEAN method is item-based. The IMEAN method average the values that all of the other users (except for the current user u) invoke the current service. Similar to UMEAN, IMEAN takes the average as the prediction result.

4.3 UPCC

UPCC (user-based PCC) is a common prediction method. If this method is going to be used, it is required to find several users who are most similar to the current user. According to the values of the most similar users invoking service i, UPCC can get the prediction result.

When the UPCC method is employed to predict unknown values, the following equation is used:

$$\hat{r}_{u,i} = \overline{u} + \frac{\sum\limits_{u' \in U} Sim(u', u)(r_{u',i} - \overline{u'})}{\sum\limits_{u' \in U} Sim(u', u)} \tag{9}$$

Here, $\hat{r}_{u,i}$ is the predicted value that u invokes service i; $\overline{u}$ is the average value of the current user u for all of the other services (except for the current service i); $\overline{u'}$ is the average value of similar user u' for all services; U represents the set of the top k users who are most similar to u, and these users all have invoked the service i; and $Sim(u', u)$ is defined in equation (1).

4.4 IPCC

Similar to UPCC, IPCC (item-based PCC) is another prediction method. After the completion of similarity calculation, IPCC selects the top k services that are most similar to the current service i to conduct the prediction. According to the value that u invokes these similar services, IPCC can predict the value of u invoking i.

Employing the IPCC method to execute the prediction, the following equation is commonly used:

$$\hat{r}_{u,i} = \overline{i} + \frac{\sum\limits_{i' \in I} Sim(i', i)(r_{u,i'} - \overline{i'})}{\sum\limits_{i' \in I} Sim(i', i)} \tag{10}$$

Here, $\hat{r}_{u,i}$ refer to the predicted value; $\overline{i}$ is the average value that all the users (except for the current user u) for the current service i; $\overline{i'}$ denotes the average value that all of the users invoke the similar service i'; I is the set of the top k services that are most similar to service i, and these services have all been invoked by the current user u; and $Sim(i', i)$ is defined in equation (2).

4.5 WSRec

In [2], the authors have proposed a new prediction method which is the integration of UPCC and IPCC. In essence, this method is based on PCC (both user-based and item-based).

However, they have taken several measures to improve the prediction.

To improve the accuracy of similarity calculated by PCC, they add a *significance weight*. Moreover, they consider that user-based and item-based methods may achieve different prediction results. For the sake of balancing the results of these two parts, they employ two *confidence weights* to adjust them. Additionally, in order to overcome the diversity of data distributions and correlation natures, which are implied in the different data sets, they introduce a factor λ to determine the ratio for how their final result depends on the user-based and item-based methods.

4.6 NRCF

The NRCF prediction method is proposed in [1]. The N-RCF includes user-based and item-based NRCF. The final prediction result is composed of these two parts. They introduced a parameter λ to determine to what extent the final prediction result is dependent on the user-based and item-based NRCF.

4.7 Our Method

Based on our similarity computation method (RA), we propose a novel memory-based prediction method, named ratio-based collaborative filtering method (RACF). In this section, we derive RACF and give the prediction process for unknown value.

Suppose that service j is a similar service of i, $((r_{1,i}, r_{1,j}), (r_{2,i}, r_{2,j}), ..., (r_{u-1,i}, r_{u-1,j}))$ are the values of users 1, 2,..., $u-1$ for services i and j, and these values are known. We can see this in Table 5.

Let $U^- = \{(r_{1,i}, r_{1,j}), (r_{2,i}, r_{2,j}), ..., (r_{u-1,i}, r_{u-1,j})\}$.

We add an element $(r_{u,i}, r_{u,j})$ to U^-, and we get:

$U = \{(r_{1,i}, r_{1,j}), (r_{2,i}, r_{2,j}), ..., (r_{u-1,i}, r_{u-1,j}), (r_{u,i}, r_{u,j})\}$.

The numbers of elements in U^- and U are $u-1$ and u.

In the element $(r_{u,i}, r_{u,j})$, $r_{u,j}$ is the value of user u invoking service j, which is a known value. Because user u has not invoked i and $r_{u,i}$ is an unknown value, we are required to predict it.

According to our similarity computation method (equation (8)), we can determine the similarity between services j and i:

$$Sim(i,j)_{U^-} = \frac{\sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})}}{u-1} \tag{11}$$

Here, $w \in U^-$, U^- is the set of values that $u-1$ users invoke both services i and j; $u-1$ is the number of element in U^-; $\min(r_{w,i}, r_{w,j})$ is the minimum of $r_{w,i}$ and $r_{w,j}$; and $\max(r_{w,i}, r_{w,j})$ is the maximum of $r_{w,i}$ and $r_{w,j}$.

There is no unknown element in U^-, thus we can calculate $Sim(i,j)_{U^-}$.

Assuming we have already known the value of user u invoking service i, i.e., $r_{u,i}$, according to equation (8), we can also calculate the similarity between services j and i:

$$Sim(i,j)_U = \frac{\sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})} + \frac{\min(r_{u,i}, r_{u,j})}{\max(r_{u,i}, r_{u,j})}}{u} \tag{12}$$

We can see that $Sim(i,j)_{U^-}$ is the similarity between services j and i, and $Sim(i,j)_U$ is also their similarity. When the number of elements in U^- is large enough, we have reason to believe that $Sim(i,j)_U = Sim(i,j)_{U^-}$. Namely, when $u-1$ is large enough, similarity between services j and i calculated according to u elements should be equal to the similarity calculated according to $u-1$ elements.

That is:

$$Sim(i,j)_U = Sim(i,j)_{U^-} \tag{13}$$

i.e.,

$$\frac{\sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})} + \frac{\min(r_{u,i}, r_{u,j})}{\max(r_{u,i}, r_{u,j})}}{u} = \frac{\sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})}}{u-1} \tag{14}$$

However, the current user u has not invoked the current service i, and we do not know the value of $r_{u,i}$. We can use the prediction value to replace it. Suppose that the prediction value of u for i is $\tilde{r}_{u,i}$. We replace $r_{u,i}$ with $\tilde{r}_{u,i}$ in equation (14), and we get:

$$\frac{\sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})} + \frac{\min(\tilde{r}_{u,i}, r_{u,j})}{\max(\tilde{r}_{u,i}, r_{u,j})}}{u} = \frac{\sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})}}{u-1} \tag{15}$$

Next, we will derive $\tilde{r}_{u,i}$:

According to equation (15), we can get:

$$\begin{aligned} &(u-1)\left(\sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})} + \frac{\min(\tilde{r}_{u,i}, r_{u,j})}{\max(\tilde{r}_{u,i}, r_{u,j})}\right) \\ &= u \sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})} \end{aligned} \tag{16}$$

$$(u-1)\frac{\min(\tilde{r}_{u,i}, r_{u,j})}{\max(\tilde{r}_{u,i}, r_{u,j})} = \sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})} \tag{17}$$

$$\frac{\min(\tilde{r}_{u,i}, r_{u,j})}{\max(\tilde{r}_{u,i}, r_{u,j})} = \frac{\sum_{w=1}^{u-1} \frac{\min(r_{w,i}, r_{w,j})}{\max(r_{w,i}, r_{w,j})}}{u-1} \tag{18}$$

i.e.,

$$\frac{\min(\tilde{r}_{u,i}, r_{u,j})}{\max(\tilde{r}_{u,i}, r_{u,j})} = Sim(i,j)_{U^-} \tag{19}$$

We have known $Sim(i,j)_{U^-}$ and $r_{u,j}$, but we do not know $\tilde{r}_{u,i}$ and whether $\tilde{r}_{u,i}$ is greater or less than $r_{u,j}$, so we get:

$$\begin{cases} \frac{\tilde{r}_{u,i}}{r_{u,j}} = Sim(i,j)_{U^-}, & if\ \tilde{r}_{u,i} < r_{u,j} \\ \frac{r_{u,j}}{\tilde{r}_{u,i}} = Sim(i,j)_{U^-}. & else \end{cases} \tag{20}$$

that is:

$$\tilde{r}_{u,i} = r_{u,j} \times Sim(i,j)_{U^-}, \quad if\ \tilde{r}_{u,i} < r_{u,j} \tag{21}$$

or,

$$\tilde{r}_{u,i} = \frac{r_{u,j}}{Sim(i,j)_{U^-}}. \quad else \tag{22}$$

Here, $\tilde{r}_{u,i}$ is the predicted value of RACF when the current user u invokes the current service i. The real value of u invoking i is $r_{u,i}$. The value $|r_{u,i} - \tilde{r}_{u,i}|$ indicates the difference between them.

TABLE 5
An example of user-item matrix

Values \ Services / Users	a	b	c	d	$\cdots$	h	i	j	$\cdots$
1		$r_{1,b}$			$\cdots$	$r_{1,h}$	$r_{1,i}$	$r_{1,j}$	$\cdots$
2	$r_{2,a}$		$r_{2,c}$	$r_{2,d}$	$\cdots$		$r_{2,i}$	$r_{2,j}$	$\cdots$
3		$r_{3,b}$		$r_{3,d}$	$\cdots$		$r_{3,i}$	$r_{3,j}$	$\cdots$
4	$r_{4,a}$		$r_{4,c}$		$\cdots$		$r_{4,i}$	$r_{4,j}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$u-1$	$r_{u-1,a}$				$\cdots$	$r_{u-1,h}$	$r_{u-1,i}$	$r_{u-1,j}$	$\cdots$
u		$r_{u,b}$		$r_{u,d}$	$\cdots$		?	$r_{u,j}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

In the previous description, we assume that $u-1$ is large enough. In fact, as long as $u-1>1$, we can think this assumption is established. However, the bigger the $u-1$ is, the more consistent $Sim(i,j)_{U-}$ and $Sim(i,j)_{U}$ are, and the more accurate the prediction result is.

In the above analysis, taking *one* similar service j as an example, we have introduced our method to predict one value of user u invoking service i. Similar to other item-based methods, in practice, our method relies on k of the most similar services to execute the prediction.

Algorithms 1 to 3 describe the complete process of our prediction method. Through these algorithms, we can predict the value of the current user for the current service.

There are some variables and matrices in the algorithms. In Table 6, we gather them together and give brief explanations for them.

TABLE 6
Variables and matrices in the algorithms

Variable or matrix	Meaning
u	the current user
v	a user, $v \neq u$
i	the current service
j	a service, $j \neq i$
$r'_{u,i}$	an intermediate value
$r''_{u,i}$	an intermediate value
$\tilde{r}_{u,i}$	the intermediate predicted value
$\hat{r}_{u,i}$	the final predicted value
C	the sparse user-item matrix
S	this matrix stores the ids of all similar services and their similarities to the current service
V	this matrix stores k values of $\tilde{r}_{u,i}$, according to equation (21)
W	this matrix stores k values of $\tilde{r}_{u,i}$, according to equation (22)

Before the prediction, we are required to calculate the similarity between each service and the current service i. We store the similar service ids and their corresponding similarities to i in the matrix S. The process is given in Algorithm 1.

According to Algorithm 1, we can also know the values of each similar service and the current service invoked by the same users. We compare these values and count them. The results of Algorithm 1 are stored in matrix S.

Algorithm 1: Calculate the similarity between each service and the current service

```
Input:  (1) Matrix $C$ //the sparse user-service matrix
        (2) Current user $u$
        (3) Current service $i$
Output: Matrix $S$
$l = 1$;
for each service $j$ in $C$ do
    $s\_li = 0$, $c\_li = 0$, $sum = 0$, $flag\_s = 0$;
    for each user $v$ in $C$ do
        if $v$ has invoked $i$ and $j$ AND $u$ has invoked $j$ then
            $flag\_s = 1$;
            if $C(v,i) > C(v,j)$ then
                $s\_li = s\_li + 1$;
                $min = C(v,j)$;
                $max = C(v,i)$;
            end
            else
                $c\_li = c\_li + 1$;
                $min = C(v,i)$;
                $max = C(v,j)$;
            end
            $sum = sum + min/max$;
        end
    end
    if $flag\_s == 1$ then
        $S(l,1) = j$;
        $S(l,2) = i$;
        $S(l,3) = sum/(s\_li + c\_li)$; //$Sim(i,j)$
        $S(l,4) = s\_li$;
        $S(l,5) = c\_li$;
        $l = l + 1$;
    end
end
$Sort(S)$;    //rearrange the rows of $S$ according to $S(l,3)$ in descending order
Output: $S$.
```

According to Algorithm 1, we can determine k of the most similar services of i. Based on these similar services and the matrix S, we can calculate the intermediate values of u for i, i.e., $r'_{u,i}$ and $r''_{u,i}$. The detailed calculation process can be seen in Algorithm 2.

Algorithm 2: Calculate $r'_{u,i}$ and $r''_{u,i}$

```
Input:  (1) Matrix $S$
        (2) $k$ // number of the most similar service
Output: $r'_{u,i}$ and $r''_{u,i}$
//step 1: To calculate the prediction values according
to $k$ of the most similar service
for $h = 1$ to $k$ do
    $s = S(h, 1)$;
    $V(h, 1) = r_{u,s} \times Sim(i, s)$;
    // $\tilde{r}_{u,i}$, according to equation (21).
    $W(h, 1) = r_{u,s}/Sim(i, s)$;
    // $\tilde{r}_{u,i}$, according to equation (22).
end
$Sort(V)$;    //sort the data of $V$ in ascending order
$Sort(W)$;    //sort the data of $W$ in ascending order
//step 2: To calculate $r'_{u,i}$ and $r''_{u,i}$
$sum\_1 = 0$, $sum\_2 = 0$, $a = round(k/3)$;
for $l = a + 1$ to $k - a$ do
    $sum\_1 = sum\_1 + V(l, 1)$;
    $sum\_2 = sum\_2 + W(l, 1)$;
end
$r'_{u,i} = sum\_1/(k - 2 \times a)$;
$r''_{u,i} = sum\_2/(k - 2 \times a)$;
Output: $r'_{u,i}$ and $r''_{u,i}$.
```

In Algorithm 2, l has a range from $a + 1$ to $k - a$. The reason why we set the parameter as this is given in the following.

According to equation (21), using k similar services, we can get k values of $\tilde{r}_{u,i}$s and store them in matrix V. Each $\tilde{r}_{u,i}$ is a predicted value. Similarly, according to equation (22), using k similar services, we can also obtain k values of $\tilde{r}_{u,i}$s and store them in matrix W.

To get the $r'_{u,i}(r''_{u,i})$, we can sum up the k values in $V(W)$, and take the average as $r'_{u,i}(r''_{u,i})$. We consider this method is a traditional one. In our algorithm, in order to improve the accuracy of prediction, we employ a new method. Firstly, we sort the k values in $V(W)$ in ascending order and averagely divide them into three parts. Finally, we average the values in the middle part and take the result as the $r'_{u,i}(r''_{u,i})$. Compared with traditional method, this method really has some effect.

In Algorithm 2, $a+1$ and $k-a$ are the sequence numbers of the beginning value and the end value in the middle part, respectively, and $k - 2 \times a$ is the number of the values in the middle part.

The results of Algorithm 2 are $r'_{u,i}$ and $r''_{u,i}$. To determine the final predicted value of u for i, defined as $\hat{r}_{u,i}$, we present the Algorithm 3.

Through these algorithms, we can employ the similar services to predict the value of the current user for the current service.

In the above, we have described how to use similar services to predict unknown values. Similarly, we can also use similar users to predict unknown values.

Algorithm 3: Calculate the prediction value of user u for service i: $\hat{r}_{u,i}$

```
Input:  (1) Matrix $S$
        (2) $r'_{u,i}$ and $r''_{u,i}$
Output: Final prediction value $\hat{r}_{u,i}$
$ss = 0$, $sc = 0$;
for $h = 1$ to $k$ do
    $ss = ss + S(h, 4)$;
    $sc = sc + S(h, 5)$;
end
if $ss > sc$ then
    $\hat{r}_{u,i} = r''_{u,i}$;
end
else
    $\hat{r}_{u,i} = r'_{u,i}$;
end
Output: $\hat{r}_{u,i}$.
```

Suppose that user v is a similar user of u, we have known the values of u and v for services 1, 2,..., $i - 1$, i.e., $((r_{u,1}, r_{v,1}), (r_{u,2}, r_{v,2}), ..., (r_{u,i-1}, r_{v,i-1}))$, and we also have known the value of v for i, i.e., $r_{v,i}$. We are required to predict the value of u for i, i.e., $r_{u,i}$.

Let $I^- = \{(r_{u,1}, r_{v,1}), (r_{u,2}, r_{v,2}), ..., (r_{u,i-1}, r_{v,i-1})\}$.

We add an element $(r_{u,i}, r_{v,i})$ to I^-, and we get:

$I = \{(r_{u,1}, r_{v,1}), (r_{u,2}, r_{v,2}), ..., (r_{u,i-1}, r_{v,i-1}), (r_{u,i}, r_{v,i})\}$,

The numbers of elements in I^- and I are $i - 1$ and i.

According to equation (7), we can calculate the similarity between v and u:

$$Sim(u, v)_{I^-} = \frac{\sum_{k=1}^{i-1} \frac{\min(r_{u,k}, r_{v,k})}{\max(r_{u,k}, r_{v,k})}}{i - 1} \tag{23}$$

Here, $k \in I^-$; I^- is the set of values that two users invoke the same service concurrently; $i-1$ is the number of element in I^-; $\min(r_{u,k}, r_{v,k})$ is the minimum of $r_{u,k}$ and $r_{v,k}$; and $\max(r_{u,k}, r_{v,k})$ is the maximum of $r_{u,k}$ and $r_{v,k}$.

According to equation (7), we can also determine the similarity between v and u:

$$Sim(u, v)_{I} = \frac{\sum_{k=1}^{i-1} \frac{\min(r_{u,k}, r_{v,k})}{\max(r_{u,k}, r_{v,k})} + \frac{\min(r_{u,i}, r_{v,i})}{\max(r_{u,i}, r_{v,i})}}{i} \tag{24}$$

We can see $Sim(u, v)_{I^-}$ is the similarity between users v and u, and $Sim(u, v)_{I}$ is also their similarity. When the number of elements in I^- is large enough, we have reason to believe that $Sim(u, v)_{I} = Sim(u, v)_{I^-}$. Namely, when $i-1$ is large enough, similarity between users v and u calculated according to i elements should be equal to the similarity calculated according to $i - 1$ elements.

That is:

$$Sim(u, v)_{I} = Sim(u, v)_{I^-} \tag{25}$$

i.e.,

$$\frac{\sum_{k=1}^{i-1}\frac{\min(r_{u,k},r_{v,k})}{\max(r_{u,k},r_{v,k})}+\frac{\min(r_{u,i},r_{v,i})}{\max(r_{u,i},r_{v,i})}}{i}=\frac{\sum_{k=1}^{i-1}\frac{\min(r_{u,k},r_{v,k})}{\max(r_{u,k},r_{v,k})}}{i-1} \tag{26}$$

However, the current user u has not invoked the current service i, and we do not know the value of $r_{u,i}$. Similar to the service part, we can use the prediction value to replace it. Suppose the prediction value of u for i is $\tilde{r}_{u,i}$. We replace $r_{u,i}$ with $\tilde{r}_{u,i}$ in equation (26), and we get:

$$\frac{\sum_{k=1}^{i-1}\frac{\min(r_{u,k},r_{v,k})}{\max(r_{u,k},r_{v,k})}+\frac{\min(\tilde{r}_{u,i},r_{v,i})}{\max(\tilde{r}_{u,i},r_{v,i})}}{i}=\frac{\sum_{k=1}^{i-1}\frac{\min(r_{u,k},r_{v,k})}{\max(r_{u,k},r_{v,k})}}{i-1} \tag{27}$$

$$\frac{\min(\tilde{r}_{u,i},r_{v,i})}{\max(\tilde{r}_{u,i},r_{v,i})}=\frac{\sum_{k=1}^{i-1}\frac{\min(r_{u,k},r_{v,k})}{\max(r_{u,k},r_{v,k})}}{i-1} \tag{28}$$

i.e.,

$$\frac{\min(\tilde{r}_{u,i},r_{v,i})}{\max(\tilde{r}_{u,i},r_{v,i})}=Sim(u,v)_{I-} \tag{29}$$

Similarly, we get:

$$\tilde{r}_{u,i}=r_{v,i}\times Sim(u,v)_{I-},\qquad if\ \tilde{r}_{u,i}<r_{u,j} \tag{30}$$

or

$$\tilde{r}_{u,i}=\frac{r_{v,i}}{Sim(u,v)_{I-}}.\qquad else \tag{31}$$

In the previous description, we assume that $i-1$ is large enough. In fact, as long as $i-1>1$, we can deem this assumption is established. However, as $i-1$ increase, $Sim(u,v)_{I-}$ and $Sim(u,v)_{I}$ become more consistent, which improves the accuracy of the prediction.

Similar to the algorithms of the service part, we can also give the corresponding algorithms of user part. According to these algorithms, we can also determine the final prediction value of the current user u for the current service i.

In our prediction method, we only employ similar services to predict the unknown values. Therefore, it is an item-based method.

5 Service Recommendation

When the predicted value of the current user for the current service has been obtained, service recommendation can be made. The recommendation system can recommend appropriate service(s) to the user according to given conditions. Here, the specific condition given by a user may be constrained by multiple objectives. For example, the user may require a service whose cost is lowest, and, at the same time, the response time must meet the interval specified by him.

The accurate and efficient recommendation method can save a lot of time and energy for the user who wants to identify optimal service(s).

The recommendation system can also provide n suitable alternatives for the user's further selection according to the predicted values. Similarly, referring to the predicted values, the recommendation system can infer the user's preferences or attributes and recommend him other services or products that are in accordance with his characteristics.

6 Experiment

To verify the effectiveness of our method, we conducted an experiment. In this section, we first describe the evaluation methodology, which includes the experimental setup, performance evaluation, and parameter setting. Then, we depict and discuss the experimental results in detail.

The experiment was run on a computer with an Intel Core i3 CPU M350 @2.27GHz and 2.27GHz, 4.00 GB RAM.

6.1 Experimental Setup

The data set for our experiment was from the web service data set of [29], which included the response time record set of 339 users invoking 5825 services and their throughput records. We took the response time data set as the user-item matrix in the experiment. In the response time data set, invalid values were defined as -1, which corresponds to the case where the users had not invoked the corresponding services. The total valid records divided by the total records was the original density of the data set, which was 0.9489.

The details of the data set are described in Table 7.

TABLE 7
Details of the QoS data set

Parameter	Value
Number of users	339
Number of services	5825
Number of records	339 $\times$ 5825
Number of valid records	1873838
Original density	0.9489

In practice, the user-item matrices were often very sparse. As such, we randomly removed values from the data set and made it a sparse matrix. In the experiment, the density of the matrix varied from 0.04 to 0.20 with a step of 0.02.

6.2 Comparison Methods and Parameter Setting

In the experiment, the reference schemes included UMEAN, IMEAN, UPCC, IPCC, WSRec and NRCF, which were introduced in the previous sections. For the prediction of UPCC, IPCC, WSRec and NRCF, we can refer to equations (9), (10) and references [2], [1].

The prediction results of UPCC, IPCC, RACF, WSRec and NRCF are influenced by the number of similar neighbors. We set the neighbors of UPCC, IPCC, RACF, WSRec and NRCF to 20, 20, 20, 10 and 50, respectively.

For WSRec and NRCF, we set the λ to 0.2 for all cases.

6.3 Performance Evaluation

To evaluate the performance of our method, we compared our method with others in the following three aspects:

1) The number of predicted values with smaller deviation.

   Deviation represents the difference between the predicted value and the real value, which is usually given in the form of an absolute value. A predicted value with smaller deviation means it is more accurate and the prediction method is superior.

Using two methods to predict one real value, each method will produce a predicted value and a deviation. Comparing their deviation, we can identify which method performs better. While using two methods to predict N real values, each method will have N predicted values. We count the number of predicted values with the smaller deviation of each method and take this number as one of the evaluation indexes.

For clarity, we define *the predicted value with smaller deviation* as *PVSD* and define *the number of predicted values with smaller deviation* as N_{PVSD}.

A higher N_{PVSD} means the method is superior. The results are shown in Fig. 2.

2) MAE

MAE (Mean Absolute Error) is defined as the ratio of the sum of the deviations and the number of deviations:

$$MAE = \frac{\sum_{i=1}^{N} |r_{u,i} - \hat{r}_{u,i}|}{N} \tag{32}$$

Here, $r_{u,i}$ is the real value; $\hat{r}_{u,i}$ is the predicted value; and N is the number of deviations. The smaller the value of $|r_{u,i} - \hat{r}_{u,i}|$ is, the more accurate the prediction result, and the better the prediction method. The results can be seen in Fig. 3.

3) Computation time

Computation time (or prediction time) reflects the speed of the algorithm. In the experiment, we compared the computation times for the 5 methods to predict 10500 unknown values. The results are shown in Fig. 4.

6.4 Experimental Results and Analysis

Fig. 2 shows the N_{PVSD} result of RACF compared with the others 6 methods.

In each sub figure, under each density condition, the number of predicted values is 10000, i.e., the N_{PVSD} of RACF + N_{PVSD} of the other method is 10000. For example, in Fig. 2(c), when the density is 0.18, the N_{PVSD} of RACF is 7201, N_{PVSD} of UPCC is 2799, and $7201 + 2799 = 10000$.

From Fig. 2, several conclusions can be drawn.

1) The N_{PVSD} of RACF is higher than all the other methods under all density condition.
2) In Fig. 2(a), (b), (c), (d) and (e), the values of N_{PVSD} for RACF are relatively stable and are far greater than those of the other methods. Even in the worst case (Fig. 2(b), density = 0.04), RACF (6853) performs better than IMEAN (3147).
3) In Fig. 2(f), with the increase of density, N_{PVSD} of RACF decreases slowly, but it is still much higher than NRCF.

Fig. 3 plots the MAE result of every method under different density conditions.

As seen from Fig. 3, RACF consistently achieves better performance than the others do. Among UPCC, IPCC, WSRec and NRCF, WSRec and NRCF are better. The best MAE performances of WSRec, NRCF and RACF are 0.4766, 0.4628 and 0.3886. Compared with WSRec and NRCF, RACF improves the prediction accuracy (22.65 percent and 19.09 percent better than WSRec and NRCF, respectively).

Conducting the prediction, UMEAN takes the average of the current user for all other services (except for the current service) as the result. When predicting the response time, this method does not take into account the deployment location of the service, nor the distance between the user and the service. The values of the response time fluctuate greatly when the user invokes different services, because the user's location is fixed while different services are deployed at different places. This method has a large deviation due to its lesser consideration.

When IMEAN is used to predict the unknown value, this method takes the average value of all other users (except for the current user) for the same service as the result. Similar to UMEAN, this method also has a large deviation.

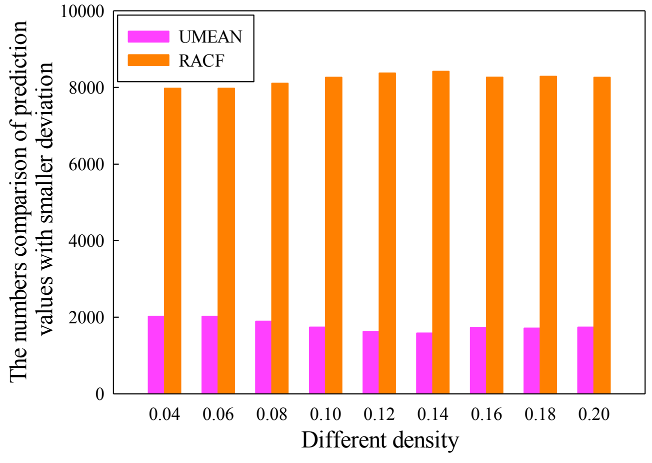
UPCC and IPCC employ the similar users and services to predict the unknown values, respectively. The results of UPCC and IPCC are not ideal, which is likely to be caused by the similarity computation formula of PCC. According to the analysis in the previous section, PCC cannot calculate the similarity between users or services accurately.

The WSRec method combines UPCC and IPCC. Meanwhile, it adds two correction factors $\frac{2\times|I_u \bigcap I_v|}{|I_u|+|I_v|}$ and $\frac{2\times|U_i \bigcap U_j|}{|U_i|+|U_j|}$. The purpose of these factors is to prevent the problem of too high similarity caused by accidental situations. The union $|I_u \bigcap I_v|$ is the number of services invoked by both the user u and user v, while $|U_i \bigcap U_j|$ is the number of users who invoked both service i and service j concurrently. When the matrix is very sparse, the services invoked concurrently by two users and the users who invoke two services are not determined by the users themselves. In these cases, the correction factors may have a limited effect on the final results in practice.

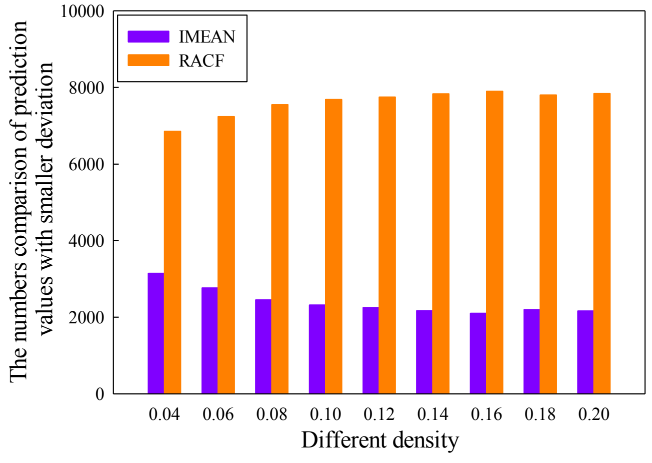
In order to improve the accuracy of the prediction, WSRec removes the user or service whose similarity (with the current user or current service) is non-positive. They have also taken other measures to improve the accuracy. Comparing the result of WSRec with UPCC and IPCC, we can see that these measures play an important role.

As can be seen from Fig. 2 and Fig. 3, the NRCF method has certain improvements over UMENA, IMENA, UPCC, IPCC and WSRec. It shows that NRCF is better than these methods. However, comparing NRCF with RACF, RACF is superior to NRCF in N_{PVSD} and MAE under every density condition.

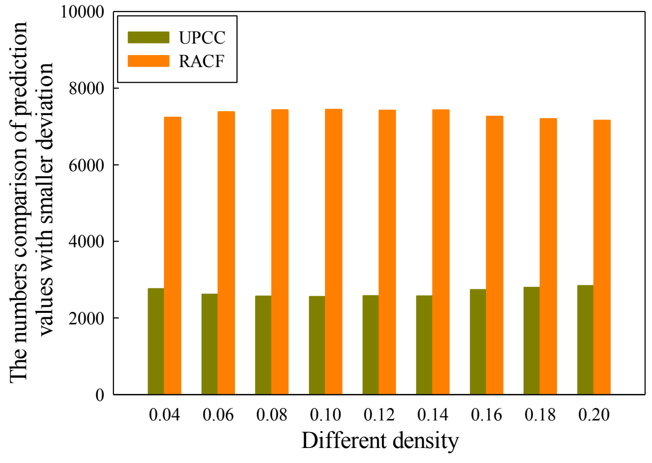
In the process of predicting unknown values, NRCF requires conducting row normalization and column normalization for the sparse user-item matrix. The purpose of these operations is to eliminate the differences in e-valuation caused by the subjectivity of different users. In the above, we have known that not every type of QoS attribute is determined by the user's subjectivity. When the sparse user-item matrix is normalized, different densities of the matrix will have certain influence from the different user's subjective valuation style. Normalization does not necessarily eliminate the differences caused by the users'



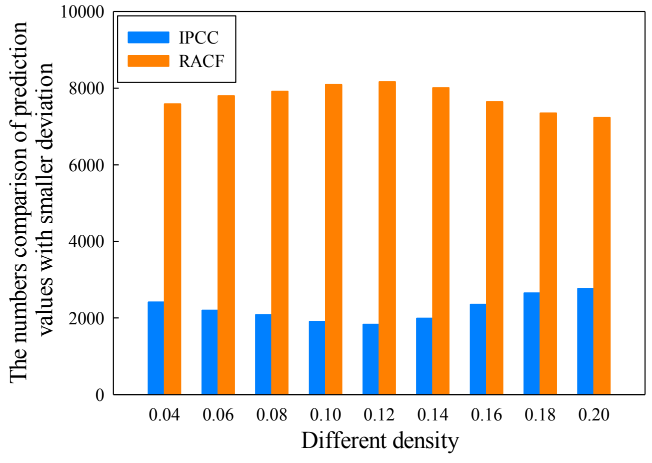
(a) Results of UMEAN and RACF



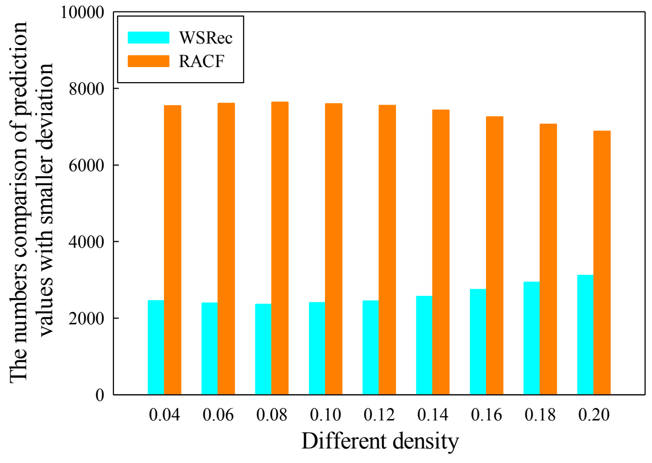
(b) Results of IMEAN and RACF



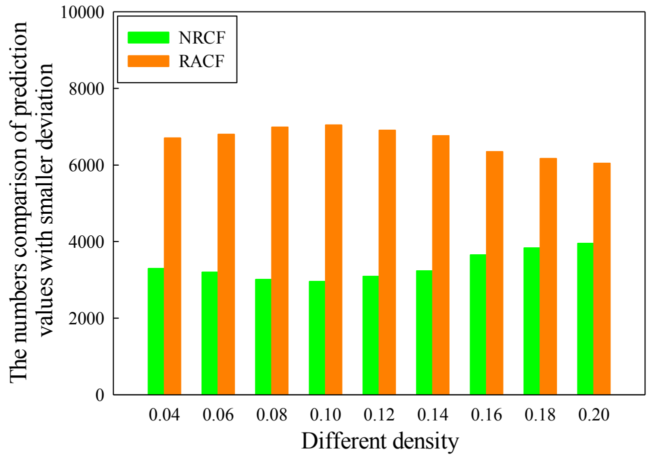
(c) Results of UPCC and RACF



(d) Results of IPCC and RACF



(e) Results of WSRec and RACF



(f) Results of NRCF and RACF

Fig. 2. Comparison of N_{PVSD}.

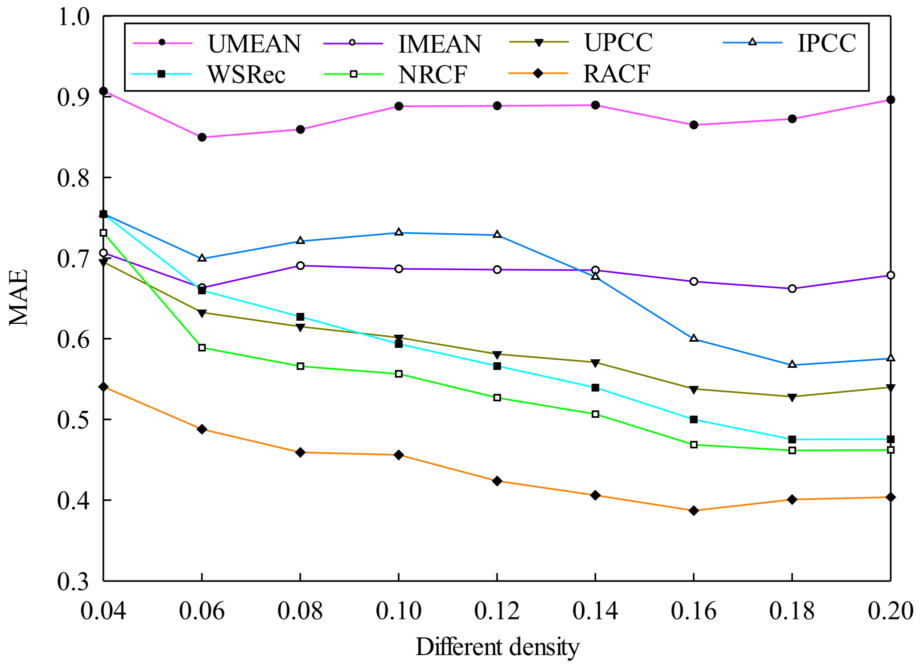


Fig. 3. Comparison of MAE.

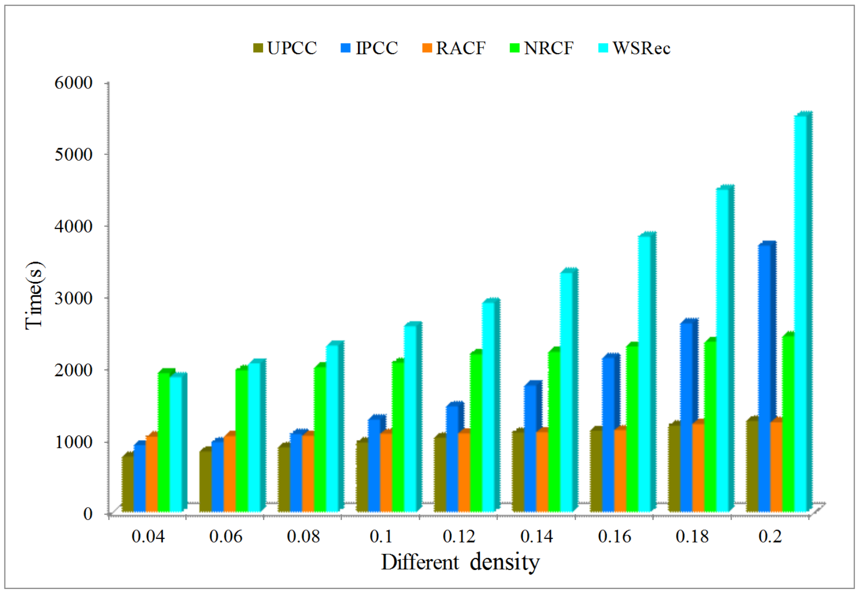


Fig. 4. Comparison of computation time.

subjective assessment completely. Additionally, similar to PCC and COS, the similarity calculation of NRCF is not accurate. It cannot correctly determine the most similar users and the most similar services. This will also affect the final prediction result.

Fig. 4 is a comparison of computation times for the 5 methods when they are each employed to predict 10500 unknown values.

As observed in Fig. 4, RACF outperforms IPCC, NRCF and WSRec. Under each density condition, the time RACF cost is relatively stable. The variety of matrix density has less effect on RACF than those of the others.

The reason why RACF is superior to others is due to its simplicity. Among the 5 methods, there is a common step, which is to find similar users or similar services. After the completion of this step, the main operation of RACF is to compare the values of services invoked by users, which is much easier than the complex computations of other methods.

In the low density environment, the time RACF cost is slightly more than UPCC. With the increase of matrix density, these two methods cost almost equivalent amounts of computation time. When the density continues to increase, RACF costs less time than UPCC.

This is reasonable because the similar users of UPCC are rare when the density is low. The small number of similar users leads to a small amount of calculations. With the increase of density, the number of similar users increases as well. This raises the amount of calculations and the computation time of UPCC.

Additionally, in the user-item matrix, the number of services is 5825, which is far more than the number of users (339). Accordingly, in the course of the experiment, the number of similar services is larger than the number of similar users. Thus, the time the item-based method cost is more than those of the user-based method. This observation can be confirmed from the times that IPCC and UPCC cost. As IPCC is item-based, it costs more time than UPCC. From the perspective of computational time, the user-based method inherently has certain advantages. This is why UPCC costs less time than RACF at first, but is more than RACF when the density increases to a certain extent.

When the density is low, IPCC and RACF spend similar amounts of time. However, when the density is higher than 0.08, the time IPCC cost exceed that of RACF. This is because the increase of matrix density brings more similar services and thus leads to the increase of computation time.

Under each density condition, RACF demonstrates obvious advantages over WSRec and NRCF, especially when the density is high.

As WSRec is composed of UPCC and IPCC, the time it cost is the sum of UPCC and IPCC. Moreover, this method adds a few extra steps for correction and adjustment, which also increase the computation time.

The NRCF method requires conducting row normalization and column normalization for the user-item matrix, and this method comprises a plurality of intermediate procedures. All of these operations require extra time.

6.5 Analysis for Computation Complexity

In this section we discuss the computational complexity of the above 5 methods. We assume the user-item matrix is composed of the values of m users invoking n services.

When using UPCC, IPCC, RACF, WSRec and NRCF to conduct predictions, the first step is to compute the similarity. User-based methods are needed to compute the similarity between m users and the current user. The complexity is $O(m)$. For each user, the complexity is $O(n)$, because there may be n services that are invoked concurrently by him and the current user in the worst condition. Thus, in the stage of similarity computation, the complexity of the user-based methods is $O(mn)$.

Similarly, the complexity of item-based methods is $O(mn)$, because there are n services, and for each service, there may be m users who have concurrently invoked it and the current service. UPCC is user-based. IPCC and RACF are item-based. WSRec and NRCF are the linear combinations of user-based and item-based methods. Through the above analysis, we can know, in the similarity computation stage, the complexities of UPCC, IPCC, RACF, WSRec and NRCF are $O(mn)$.

After the similarity computation, the next step of all methods is to predict unknown value. At this stage, all methods predict an unknown value according to the given number (k) of similar users or(and) similar services. The complexities of all methods are $O(1)$ at this step.

Combining the above two steps, we can see the complexities of all the methods are $O(mn)$. Although these methods have the same complexity, they involve diverse operations and additional calculations. Both WSRec and NRCF contain 2 $O(mn)$, while UPCC, IPCC and RACF each contains only 1 $O(mn)$. Thus, they cost different times when predicting equivalent amounts of unknown values.

7 Conclusion

Similarity measurements and prediction of unknown values are two important procedures of memory-based collaborative filtering service recommendation. In this paper, we proposed a new method to compute the similarity.

Based on our new similarity measurement method, we propose a prediction method. The experimental results show that our method is superior.

In the future, we will consider more reference methods.

Acknowledgments

This research is supported by National Natural Science Foundation of China under Grant No. 61132001); National High-tech R&D Program of China (863 Program) under Grant No. 2013AA102301; Program for New Century Excellent Talents in University (Grant No. NCET-11-0592); The National Key Technology Research and Development Program of China (Grant No.2012BAH94F02); The National Grand Fundamental Research 973 Program of China (Grant No. 2012CB315802; 2013CB329102)

References

[1] H. Sun, Z. Zheng, J. Chen, and M. R. Lyu, "Personalized Web Service Recommendation via Normal Recovery Collaborative Filtering," *IEEE Trans. Service Computing*, vol. 6, no. 4, pp. 573-579, 2013.

[2] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-Aware Web Service Recommendation by Collaborative Filtering," *IEEE Trans. Service Computing*, vol. 4, no. 2, pp. 140-152, 2011.

[3] U. Shardanand and P. Maes, "Social Information Filtering: Algorithms for Automating Word of Mouth," *Proc. SIGCHI Conf. Human Factors in Computing Systems*, 1995.

[4] L. Yu, L. Liu, and X. Li, "A hybrid collaborative filtering method for multiple-interests and multiple-content recommendation in E-Commerce," *Expert Systems with Applications*, vol. 28, no. 1, pp. 67-77, 2005.

[5] J. Cao, Z. Wu, Y. Wang, and Y. Zhuang, "Hybrid Collaborative Filtering algorithm for bidirectional Web service recommendation," *Knowledge and Information Systems*, vol. 36, pp. 607-627, 2013.

[6] Z. Zheng, H. Ma, M.R. Lyu, and I. King, "WSRec: A Collaborative Filtering Based Web Service Recommender System," *Proc. IEEE Int'l Conf. Web Services (ICWS '09)*, pp. 437-444, 2009.

[7] J. Wang, A.P. de Vries, and M.J. Reinders, "Unifying User-Based and Item-Based Collaborative Filtering Approaches by Similarity Fusion," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval (SIGIR '06)*, pp. 501-508, 2006.

[8] Y. Koren, "Factor in the Neighbors: Scalable and Accurate Collaborative Filtering," *ACM Trans. Knowledge Discovery from Data*, vol. 4, no. 1, pp. 1-24, 2010.

[9] J. Wu, L. Chen, Y. Feng, Z. Zheng, M. Zhou, and Z. Wu, "Predicting Quality of Service for Selection by Neighborhood-Based Collaborative Filtering," *IEEE Trans. Systems, Man, and Cybernetics: Systems*, vol. 43, no. 2, pp. 428-439, 2013.

[10] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," *Proc. Int'l Conf. Uncertainty in Artificial Intelligence (UAI '98)*, pp. 43-52, 1998.

[11] L. Liu, Nikolay Mehandjiev, and D. Xu, "Context Similarity Metric for Multidimensional Service Recommendation," *International Journal of Electronic Commerce*, vol. 18, no. 1, pp. 73-103, 2013.

[12] Y. H. Cho, J. K. Kim, and S. H. Kim, "A Personalized Recommender System Based on Web Usage Mining and Decision Tree Induction," *J. Expert Systems with Applications*, vol. 23, no. 3, pp. 329-342, 2002.

[13] Y. Cai, Ho-fung. Leung, Q. Li, H. Min, J. Tang, and J. Li, "Typicality-Based Collaborative Filtering Recommendation," *IEEE Trans. Knowledge and Data Engineering*, vol. 26, no. 3, pp. 766-779, 2014.

[14] G. Xue, C. Lin, Q. Yang, W. Xi, H Zeng, Y. Yu, and Z. Chen, "Scalable collaborative filtering using cluster-based smoothing," *Pro. ACM SIGIR Ann. Int'l Conf. Research and Development in Information Retrieval (SIGIR '05)*, pp. 114-121, 2005.

[15] X. Chen, Z. Zheng, X. Liu, Z. Huang, and H. Sun, "Personalized QoS-Aware Web Service Recommendation and Visualization," *IEEE Trans. Service Computing*, vol. 6, no. 1, pp. 35-47, 2013.

[16] L. Zhang, B. Zhang, Y. Liu, Y. Gao, and Z. Zhu, "A Web service QoS prediction approach based on collaborative filtering," *Proc. IEEE Asia-Pacific Services Computing Conference*, pp. 725-731, 2010.

[17] Y. Jiang, J. Liu, M. Tang, and X. Liu, "An Effective Web Service Recommendation Method based on Personalized Collaborative Filtering," *Proc. IEEE Int'l Conf. Web Services (ICWS '11)*, pp. 211-218, 2011.

[18] H. Liu, J. He, T. Wang, W. Song, and X. Du, "Combining user preferences and user opinions for accurate recommendation," *Electronic Commerce Research and Applications*, vol. 12, no. 1, pp. 14-23, 2013.

[19] S. Deng, L. Huang, J. Wu, J. Wu, and Z. Wu, "Trust-based personalized service recommendation: A network perspective," *J. Computer Science and Technology*, vol. 29, no. 1, pp. 69-80, 2014.

[20] R. Jin and L. Si, "A Study of Methods for Normalizing User Ratings in Collaborative Filtering," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval (SIGIR '04)*, pp. 568-569, 2004.

[21] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-Based Systems*, vol. 46, pp. 109-132, 2013.

[22] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," *Proc. Int'l Conf. World Wide Web (WWW '01)*, pp. 285-295, 2001.

[23] M. Deshpande and G. Karypis, "Item-Based Top-N Recommendation," *ACM Trans. Information System*, vol. 22, no. 1, pp. 143-177, 2004.

[24] J D M. Rennie, N Srebro, "Fast maximum margin matrix factorization for collaborative prediction," *Proc. the 22nd international conference on Machine learning*, pp. 713-719, 2005.

[25] B. Sarwar, G. Karypis G, J. Konstan, et al, "Application of dimensionality reduction in recommender system-a case study, " *Pro. the ACM WebKDD Workshop*, 2000.

[26] A. C. M. Fong, Baoyao Zhou, S. C. Hui, Guan Y. Hong, and The Anh Do, "Web Content Recommender System based on Consumer Behavior Modeling," *IEEE Trans. Consumer Electronics*, vol. 57, no. 2, pp. 962-969, 2011.

[27] F. Cacheda, V. Carneiro, D. Fernández, and V. Formoso, "Comparison of collaborative filtering algorithms: limitations of current techniques and proposals for scalable, high-performance recommender system," *ACM Transactions on the Web*, vol. 5, no. 1, 2011.

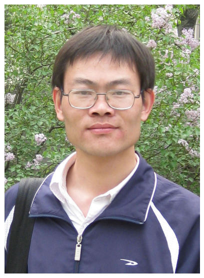
[28] Ilaria Bartolini, Z. Zhang, and Dimitris Papadias, "Collaborative Filtering with Personalized Skylines," *IEEE Trans. Knowledge and Data Engineering*, vol. 23, no. 2, pp. 190-203, 2011.

[29] Z. Zheng, Y. Zhang, and M. R. Lyu, "Distributed QoS Evaluation for Real-World Web Services," *Proc. IEEE Int'l Conf. Web Services (ICWS '10)*, pp.83-90, 2010.

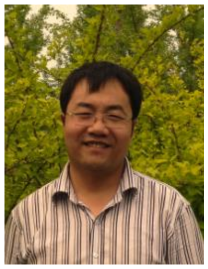
[30] Q. Yu, "QoS-aware service selection via collaborative QoS evaluation," *World Wide Web*, vol. 17, no. 1, pp. 33-57, 2014.

[31] H. Ma, I. King, and M.R. Lyu, "Effective Missing Data Prediction for Collaborative Filtering," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval (SIGIR '07)*, pp. 39-46, 2007.

[32] L. Nan and C. Li, "Zero-sum reward and punishment collaborative filtering recommendation algorithm," *Pro. IEEE/WIC/ACM Int'l Conf. Web Intelligence Agent Technology*, pp. 548-551, 2009.



Xiaokun Wu is currently a Ph.D. candidate in State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT), Beijing, China. He received his Master degree from BUPT in 2010. His research interests include service computing and mobile service.



Bo Cheng received his Ph.D. degree in computer science and engineering in July 2006, from University of Electronic Science and Technology of China. He has been working in the Beijing University of Posts and Telecommunications (BUPT) since 2008. He is now an associate professor of the Research Institute of Networking Technology of BUPT. His current research interests include network services and intelligence, Internet of Things technology, communication software and distribute computing, etc.

Junliang Chen is the chairman and a professor of the Research Institute of Networking and Switching Technology at Beijing University of Posts and Telecommunications (BUPT). He has been working in BUPT since 1955. He received the B.S. degree in electrical engineering from Shanghai Jiaotong University, China, in 1955, and his Ph.D. degree in electrical engineering in May, 1961, from Moscow Institute of Radio Engineering, formerly Soviet Russia. His research interests are in the area of communication networks and next generation service creation technology. Prof. Chen was elected as a member of the Chinese Academy of Science in 1991, and a member Chinese Academy of Engineering in 1994.