TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 06/12 pp227–238
DOI: 10.26599/TST.2019.9010007
Volume 25, Number 2, April 2020

A Time-Aware Dynamic Service Quality Prediction Approach for Services

Ying Jin, Weiguang Guo, and Yiwen Zhang*

Abstract: Dynamic Quality of Service (QoS) prediction for services is currently a hot topic and a challenge for research in the fields of service recommendation and composition. Our paper addresses the problem with a Time-aWare service Quality Prediction method (named TWQP), a two-phase approach with one phase based on historical time slices and one on the current time slice. In the first phase, if the user had invoked the service in a previous time slice, the QoS value for the user calling the service on the next time slice is predicted on the basis of the historical QoS data; if the user had not invoked the service in a previous time slice, then the Covering Algorithm (CA) is applied to predict the missing values. In the second phase, we predict the missing values for the current time slice according to the results of the previous phase. A large number of experiments on a real-world dataset, WS-Dream, show that, when compared with the classical QoS prediction algorithms, our proposed method greatly improves the prediction accuracy.

Key words: dynamic Quality of Service (QoS) prediction; time-aware; service recommendation

1 Introduction

With the rapid development of service-oriented computing, the model of distributed computing based on services is becoming a major trend in technology development. In the big data context, ever more services (cloud services, microservices, mobile services, edge services, and IoT (Internet of Things) services, etc.) with the same or similar functions and varying Qualities of Service (QoS) are emerging. How to efficiently select and recommend services that meet users needs amongst a large-scale services market has therefore become a tremendous challenge[1, 2]. Nowadays, QoS is being widely used in service composition[3–6], service selection[7–9], and service recommendation systems[10–13]. However, it is not possible for a user to use all services to get their QoS values, so it is often necessary to select or recommend services in the absence of QoS information. Therefore, the QoS prediction of services becomes critical.

Over recent years, the main prediction method for QoS has been based on Collaborative Filtering (CF) recommendation technology, including memory-based[14, 15] and model-based approaches[16, 17]. The main process of collaborative filtering prediction based on memory is to find similar users or services through Pearson Correlation Coefficient (PCC) computing similarity, then predict the missing values using the QoS values of those similar users or services. The model-based method forms a complex model based on training set data and machine learning methods, and predicts the value of the target service by combining the historical data of similar users, thereby predicting the missing QoS value. Although much work has been

• Ying Jin and Weiguang Guo are with the Department of Management, Hefei University, Hefei 230601, China. E-mail: jiny@hfuu.edu.cn; guoweiguang@hfuu.edu.cn.
• Yiwen Zhang is with the School of Computer Science and Technology, and also with the Key Laboratory of Intelligent Computing and Signal Processing, Ministry of Education, Anhui University, Hefei 230601, China. E-mail: zhangyiwen@ahu.edu.cn.
∗ To whom correspondence should be addressed.
Manuscript received: 2018-10-05; revised: 2018-12-25; accepted: 2019-03-11

done on QoS prediction methods based on collaborative filtering, some significant problems remain.

(1) QoS dynamics. In the real world, users may access the same service in a different time slice, and the QoS value of each user accessing a service will change over time with different network environments. Traditional collaborative filtering algorithms rarely consider the dynamics nature of a service's QoS, and recommendation systems need to utilize these QoS dynamics to more accurately recommend services to meet the needs of users.

(2) Data sparsity. Due to the restrictions of real-world conditions, it is impossible for users to call all services to obtain QoS values, resulting in a highly sparse user-service matrix for the service recommendation system. Furthermore, the lack of historical data negatively affects the prediction accuracy of traditional collaborative filtering methods.

As shown in Fig. 1, due to the limitations of real-world conditions, it is impossible for a user to invoke all services in different time slices to get the QoS value of the service, so in the time slice t_1, t_2, ..., t_p, user-service-time matrices are very sparse, and each time slice has a large amount of missing data. From Fig. 1 we can see that, for time slices t_1 and t_2, user u_2 has invoked the service i_1 on the time slice t_1 to get its QoS value $q_{2,1,1}$, and also invoked the same service i_1 on the time slice t_2 to get its QoS value $q_{2,1,2}$. The two QoS values are not necessarily the same because of the changed network environment. Therefore, when recommending a service to a user, it is unreasonable for the recommendation system to consider only the QoS value in a certain time slice in predicting the missing value on the current time slice. In this situation, we need to consider all data on the historical time slices t_1, $t_2, \dots, t_p$ if we are to predict the QoS value for the user calling the service on the current time slice.

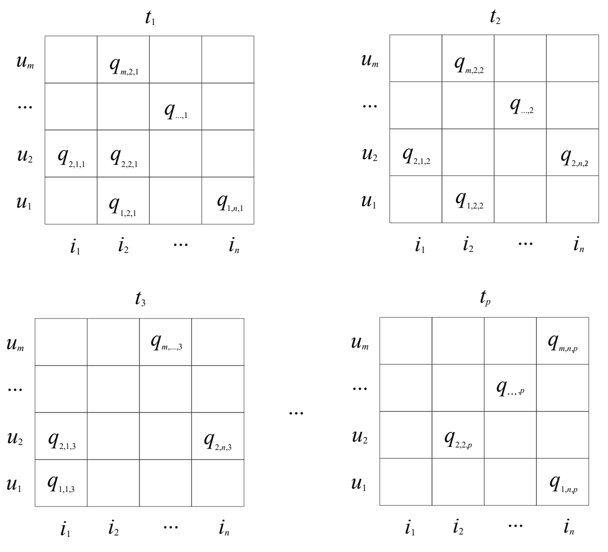

t_1
t_2
t_3
t_p
u_m
u_2
u_1
i_1 i_2 ... i_n
$q_{m,2,1}$ $q_{...,1}$ $q_{2,1,1}$ $q_{2,2,1}$ $q_{1,2,1}$ $q_{1,n,1}$
$q_{m,2,2}$ $q_{...,2}$ $q_{2,1,2}$ $q_{2,n,2}$ $q_{1,2,2}$
$q_{m,...,3}$ $q_{2,1,3}$ $q_{2,n,3}$ $q_{1,1,3}$
$q_{m,n,p}$ $q_{...,p}$ $q_{2,2,p}$ $q_{1,n,p}$

Fig. 1 User-service matrix of time-aware.

Based on the above analysis, we first need to predict and fill in the missing values in historical time slices, before the missing value of the current time slice is predicted according to the historical QoS values. Therefore, in order to solve this problem, this paper proposes a Time-aWare dynamic service Quality Prediction method, or TWQP. The main contributions of this paper are as follows:

(1) We propose a novel two-phase method named TWQP, which consists of a QoS prediction based both on historical time slices and on the current time slice. This method can effectively solve the problem of dynamic service quality prediction across different practical dynamic situations.

(2) We use a novel approach for the rapid identification of users and services that are most similar to the target user and service. Our selection method is to choose the most frequently occurring users and services in the same cluster, which helps to find more similar users and services.

(3) We conduct extensive experiments to comprehensively verify the prediction accuracy of the proposed algorithm and to compare the performance of TWQP with existing approaches, specifically, the classical CF algorithm, the improved PCC prediction algorithm, and the *k*-means-based prediction algorithm. Measuring prediction accuracy on real-world WS-Dream datasets, the experimental results demonstrate that TWQP outperforms those alternative approaches significantly.

The rest of this paper is organized as follows: Section 2 reviews the related work; Section 3 describes the prediction method and process; Section 4 presents the experimental results and analysis; and Section 5 concludes the paper.

2 Related Work

QoS prediction in the fields of service selection and recommendation has become an important research topic. The most common prediction method, which has been widely studied, is based on collaborative filtering. Collaborative filtering algorithms can be divided into two categories: model-based and memory-based.

Shao et al.[18] put forward a QoS prediction algorithm based on collaborative filtering. The algorithm first calculates the similarity between users by PCC based on historical data, and then predicts the missing QoS value based on the QoS values of similar users. Amazon, on the other hand, makes wide use of a different method of prediction[19]. The similarity between different services is calculated by PCC, and then the missing value is predicted based on services that are similar to the target service. However, the accuracy achieved by predicting missing values based only on similar users or similar services is not very high. In order to solve this problem, Zheng et al.[20] proposed a hybrid prediction method that can improve prediction significantly. The similarity between users and services is calculated by PCC. Once this is done, similar users and similar services are identified, and information from similar users and similar services is combined to predict the missing value. These methods have in common the application of PCC to identify similar users or similar services, and then using these to predict missing values.

Although these methods are simple and easy to implement, their prediction accuracy decreases with the degree of sparsity of the user-service matrix. To cope with the data sparsity problem, Yao et al.[21] proposed a unified collaborative filtering and content-based service recommendation method. This method not only considers the historical QoS value of the service, but also takes into account the semantic content of the service, and uses a set of variables to express the user's preference for the service. Zhang et al.[22] proposed a QoS prediction method based on fuzzy clustering. This method combines traditional fuzzy clustering with a similarity calculation between users, but the selection of initial points in clustering affects the prediction accuracy. Zheng et al.[23] proposed a matrix decomposition technique to predict missing values, considering the local information of similar users and the global information of the user-service matrix. Yu and Huang[24] applied the k-means algorithm to cluster users and services; this method can reduce the complexity of updating clustering when there are new users and new services. All of these methods improve the prediction accuracy to some extent, but they do not take QoS dynamics into account. Therefore, this paper proposes a time-aware QoS prediction method for services. To reflect the dynamic nature of QoS, this method has two main phases, corresponding to historical time slices and the current time slice.

3 Prediction Method and Process

3.1 Prediction based on historical time slices

As shown in Fig. 1, assuming that we need to predict the missing values on the t_3 time slice, we select the historical data from the two time slices closest to the current time slice as a reference. There are two main parts to the QoS prediction based on historical time slices. Firstly, it can be seen from Fig. 1 that user u_1 has invoked the service i_2 in both of the time slices t_1 and t_2, and the QoS values are $q_{1,2,1}$ and $q_{1,2,2}$, respectively. Therefore, in the prediction of the missing value $q_{1,2,3}$ for the t_3 time slice, the average value of the two previous time slice is used as the prediction value, that is $q_{1,2,3} = (q_{1,2,1} + q_{1,2,2})/2$.

Secondly, for missing values $q_{m,1,3}$, since user u_m has not called service i_1 in the first two time slices, the above method cannot predict the missing value. In order to solve this problem, this paper proposes an overlay algorithm to predict the missing value.

Suppose there are now six users and six service, i.e., $m = 6$, $n = 6$. Using the above method, the missing values on the t_3 time slice are partially predicted and filled, and a new user service matrix is obtained. As shown in Fig. 2, this matrix contains the response time for a user to invoke a service. In this paper, we use $M = [q_{u,i}]m \times n$ to denote the user-service matrix, where m represents the number of users and n represents the number of services. In the matrix, each element $q_{u,i}$ represents the QoS value of the service i called by the user u, where $1 \leqslant u \leqslant m$, $1 \leqslant i \leqslant n$. A null value means that the service has not been invoked by the user.

As shown in Fig. 2, we now need to predict the missing value $q_{3,1,3}$, where the user set $U = \{u_1, u_2, \ldots, u_m\}$, and the service set $I = \{i_1, i_2, \ldots, i_n\}$. Details of three main progresses follow.

(1) Clustering The Covering Algorithm (CA) proposed by professor Zhang Ling and academician

	i_1	i_2	i_3	i_4	i_5	i_6
u_1	400			360	320	480
u_2	390		420	390	450	
u_3		320	420	410		
u_4	490	280				470
u_5			430		470	420
u_6	480		410	260	310	

Fig. 2 An example of a user-service matrix on the t_3 time slice.

Zhang Ba is based on the geometric meaning of neural networks[25,26]. Using the M-P neuron model[27], a rule of domain covering is obtained with the advantages of high recognition rate and fast computing speed.

The algorithm proceeds as follows. Given an input set $K = \{x^1, x^2, \ldots, x^k, x^i \in \mathbf{R}^n\}$, let K be a set of points in m-dimensional Euclidean spaces. Let K be divided into n subsets; that is, the output of these samples has n kinds of cases. Construct the sphere of class k sample X^k as follows. For points in any sample that have not yet been covered, $x^i \in X^k$,

$$d_1 = \max\{\langle x, x^1\rangle\}, x \notin X_k \tag{1}$$

$$d_2 = \min\{\langle x, x^1\rangle | \langle x, x^1\rangle > d_1\}, x \in X_k \tag{2}$$

$$r = \frac{d_1 + d_2}{2} \tag{3}$$

where d_1 denotes the distance between the current center x^1 and the nearest heterogenous point, d_2 indicates the distance between the current center and the farthest similar points when the distance is less than d_1, r is used as a radius to construct the cover, $\langle x, x^1\rangle$ denotes the inner product of x and x^1. All covering of samples is obtained by this method.

According to the analysis of the above CA, we can get the main steps of using the CA to cluster the QoS values as follows:

Step 1: The center of gravity of all uncovered sample points is calculated and the nearest sample point is found as the center of the covered circle.

Step 2: The distances between each of the uncovered sample points and the center of the circle are calculated.

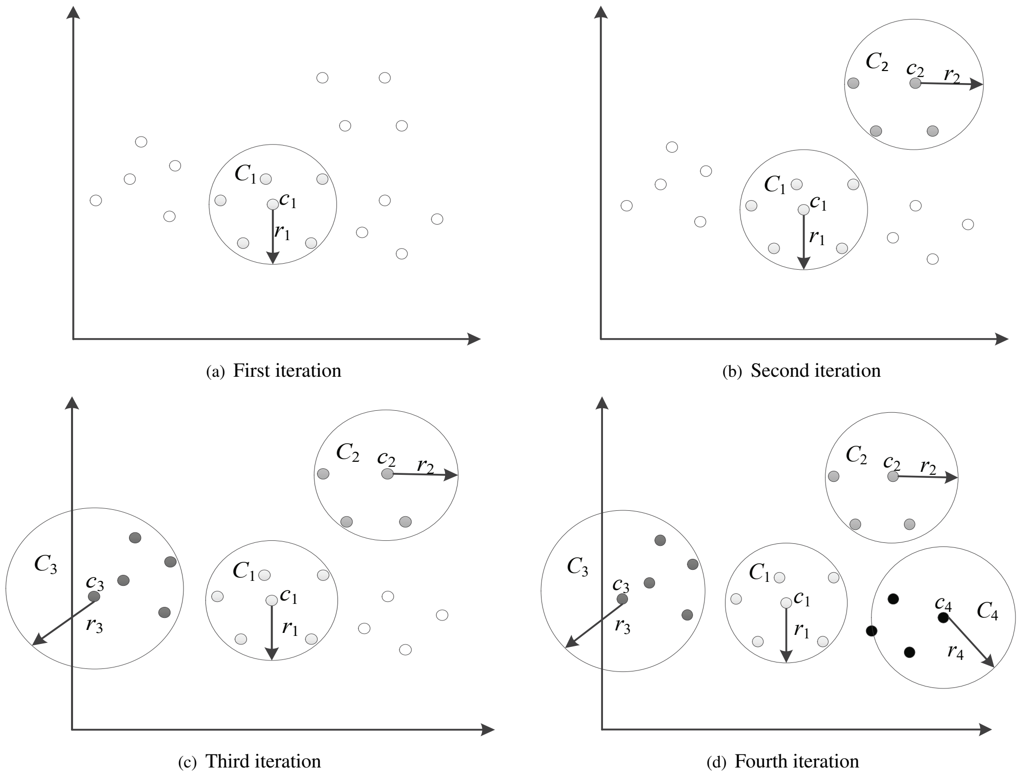
Step 3: The average distance of all the distances calculated in Step 2 is obtained, which is taken as the radius of the covering.

Step 4: The center of gravity of the current covering is obtained, the center of the circle of covering is re-determined, and a new covering is obtained. And these operations are repeated until the number of covered samples no longer changes.

Taking the sample point farthest from the current covering center as the next covering center, Steps 2 to 4 are repeated until all sample points are covered.

In this paper, in order to determine the similar user set $S(u)$ of the target user u, we first apply the overlay clustering algorithm to cluster all the users who have called the service i. The users clustered into the same cluster are considered similar, whereas the users in different clusters are different. A diagram of the clustering process is shown in Fig. 3.

In order to cluster these data points, four iterations are carried out and four clusters are obtained $C_1, \ldots, C_4$,



(a) First iteration (b) Second iteration (c) Third iteration (d) Fourth iteration

Fig. 3 Process diagram of clustering.

with their centers $c_1, \dots, c_4$. In the process of clustering, our clustering algorithm does not need to supply the number of clusters in advance, and the clustering results are not affected by the initial point.

After clustering, an $m \times m$ user relationship matrix $\boldsymbol{M_U}$ is used to record the clustering results. Every element $x_{u,v}$ $(1 \leqslant u, v \leqslant m)$ in a matrix $\boldsymbol{M_U}$ represents the total number of times that user u and user v are clustered into the same cluster. The matrix is initialized with each element set to 0. The user relationship matrix shown in Fig. 4 records the results of clustering the users in the user-service matrix in Fig. 2. When user u and user v are in the same cluster, $x_{u,v} = x_{u,v} + 1$ and $x_{v,u} = x_{v,u} + 1$. As shown in Fig. 4, $x_{1,2} = 2$, meaning that users 1 and 2 are in the same cluster twice in the clustering process. Where $x_{u,v} = x_{v,u}$ $(1 \leqslant u, v \leqslant m)$; and when $u = v$, $x_{u,v} = x_{v,u} = 0$.

Similarly, the TWQP method uses the clustering algorithm to cluster all services invoked by the same user, and records the clustering results using an $n \times n$ service relation matrix $\boldsymbol{M_I}$. Each element in the matrix $y_{i,j}$ $(1 \leqslant i, j \leqslant n)$ denotes the number of times that the service i and the service j are clustered into the same cluster, with an initial value of 0. When the service i and the service j are in the same cluster, the service relation matrix element $y_{i,j}$ is updated with $y_{i,j} = y_{i,j} + 1$ and $y_{j,i} = y_{j,i} + 1$. Obviously, the service relation matrix is also a symmetric matrix.

(2) Selection

After clustering all users and services, we select users similar to the target user u and services similar to the target service i. Supposing both user u and user v have called the service i, they will be placed in the same cluster if they have similar QoS values for service i.

Therefore, the proposed TWQP method mainly selects users who have called a large number of services in common with the target user u as similar users. The user matrix $m \times m$ in $\boldsymbol{M_U}$ contains the results of clustering, where the element $x_{u,v}$ represents the total number of times that user u and user v have been placed into the same cluster. A greater value of the element $x_{u,v}$ indicates that user u and user v have called a large number of services in common and have similar quality evaluation for these services. As shown in Fig. 4, users u_1, u_2, u_4, and u_5 are all placed into the same cluster, but users u_1, u_2, and u_5 are placed into the same cluster more often than u_4. Therefore, in selecting users similar to target user u_1, preference is given to users u_2 and u_5.

For target services i, TWQP applies similar methods to select the former k similar services. Whether choosing a similar user or a similar service, k is a variable parameter, and different datasets usually have their own characteristics, so there are different optimal k values.

(3) Prediction

In the previous progress, we selected k similar users for the target user u. The missing value can then be predicted based on the quality evaluation of these similar users. In this paper, Eq. (4) is used to predict the missing value of user u calling service i,

$$q_{u,i}(u) = \frac{\sum\limits_{v \in S_u} x_{u,v} \times q_{v,i}}{\sum\limits_{v \in S_u} x_{u,v}} \tag{4}$$

where $x_{u,v}$ represents the total number of times that user u and user v are placed into the same cluster; $q_{v,i}$ is the quality value of the user v calling service i; and S_u represents a set of users similar to user u.

Similar to the user-based method, the missing value of user u invoking service i is predicted based on application of Eq. (5),

$$q_{u,i}(i) = \frac{\sum\limits_{j \in S_i} x_{i,j} \times q_{u,j}}{\sum\limits_{j \in S_i} x_{i,j}} \tag{5}$$

where $x_{i,j}$ represents the total number of times that service i and service j are placed into the same cluster; $q_{u,j}$ is the quality value of the user u calling the service j; and S_i represents a set of services similar to service i.

Equations (4) and (5) are based on the use of similar users and similar services, respectively, to predict missing values. However, due to the sparsity of the user-service matrix, there may be a very small number of similar users or services, which leads to a low accuracy of quality prediction methods based only on either of

$$\boldsymbol{M_U} = \begin{array}{cccccc} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \\ \left[\begin{matrix} 0 & 2 & 0 & 1 & 2 & 0 \\ 2 & 0 & 1 & 0 & 2 & 1 \\ 0 & 1 & 0 & 3 & 0 & 2 \\ 1 & 0 & 3 & 0 & 1 & 0 \\ 2 & 2 & 0 & 1 & 0 & 1 \\ 0 & 1 & 2 & 0 & 1 & 0 \end{matrix}\right. & & & & & \left.\right] \end{array}$$

Fig. 4 An example of a user relationship matrix.

these formulate. Therefore, combining the prediction results based on similar users and similar services will greatly improve the prediction accuracy. The combined prediction is

$$q_{u,i} = \lambda q_{u,i}(u) + (1-\lambda) q_{u,i}(i) \tag{6}$$

where the parameter $\lambda (0 \leqslant \lambda \leqslant 1)$ is a weight value, which indicates that the proposed method TWQP depends on the proportion of similar users and similar services. The parameter λ is a variable and its optimal value is related to the characteristics of the corresponding dataset. In a real-world environment, the number of users similar to the target user and the number of similar to the target service will vary.

3.2 Prediction based on current time slice

After all the missing values in the historical time slices are predicted and filled, it is necessary to predict the QoS values of the services of all the user-service matrices in the current time slice.

As shown in Fig. 5, none of the users in the current time slice has invoked any services, but when the recommendation system recommends a service to the user to meet their needs, based on the QoS values in the historical slices, the missing QoS value of the user accessing the service in the current time slice needs to be predicted and filled. For this reason, we use the historical data from the previous T time slices as a reference; that is, $t_{p-T+1}, \ldots, t_{p-1}$, and t_p time slices. The following formula is then used to calculate all the missing values in the matrix t_{current}:

$$q_{u,i}^{\text{currrent}} = \sum_{p-T+1}^{p} \frac{q_{u,i,t}}{T} \tag{7}$$

where $q_{u,i,t}$ represents the QoS value of user u accessing the service i on time slice t, and T represents the number of historical slices selected.

4 Experimental Results and Analysis

4.1 Dataset

In order to evaluate the effectiveness of the proposed method, we ran a number of experiments on the real-world dataset WS-Dream[28]. The WS-Dream dataset contains two attribute values – Response Time (RT) and ThroughPut (TP) – obtained from 142 users invoking 4532 real services in 64 time slices. This dataset has been used for many important previous studies. In the real world, because most users have called only a few services, the generated user-service-time matrix is sparse. For this reason, we randomly move part of the data to serve as the training set and the rest is employed as the test set.

4.2 Alternative methods for comparison and evaluation indicators

4.2.1 Alternative methods for comparison

• UPCC (User-based collaborative filtering using PCC). This is chosen in order to compare the prediction accuracy of the user-based TWQP method with the traditional collaborative filtering method. In the phase of historical time slice prediction, this method uses PCC to calculate similarity and find similar users. The missing values in historical time slices are then predicted based on these similar users. Finally, the user-service matrix of the current time slice is predicted according to the data in historical time slices.

• IPCC (Item-based collaborative filtering using PCC). Similar to UPCC, this method uses PCC to calculate similarity and find similar services, and then predicts the missing values in historical time slices based on similar services.

• UIPCC (hybrid User and Item-based using PCC). This method combines UPCC and IPCC to predict the missing values in historical time slices based on similar users and similar services. Finally, the QoS value of the current time slice is predicted according to the historical time slice data.

• TOSEM. This method uses an improved PCC to calculate the similarity between users and services and find similar users and similar services, and then predicts the missing values in historical time slices[29].

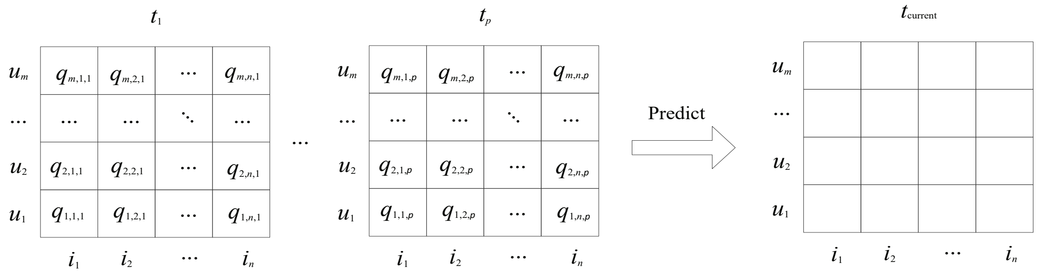

t_1
t_p
t_{current}
u_m
u_2
u_1
$q_{m,1,1}$ $q_{m,2,1}$ $q_{m,n,1}$
$q_{2,1,1}$ $q_{2,2,1}$ $q_{2,n,1}$
$q_{1,1,1}$ $q_{1,2,1}$ $q_{1,n,1}$
$q_{m,1,p}$ $q_{m,2,p}$ $q_{m,n,p}$
$q_{2,1,p}$ $q_{2,2,p}$ $q_{2,n,p}$
$q_{1,1,p}$ $q_{1,2,p}$ $q_{1,n,p}$
Predict
i_1 i_2 i_n

Fig. 5 Prediction process based on current time slice.

• CAPK. In this method, the k-means clustering algorithm is used to predict the missing values in historical time slices based on similar users and similar services[30].

• MF. This method uses a neighbor-based matrix decomposition algorithm to predict the missing values for a service in a historical time slice[22].

4.2.2 Evaluation index

In order to evaluate the effectiveness of different methods, the following two indexes are compared: the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE). These two indexes have been widely used in the research of services prediction.

MAE is defined as

$$MAE = \frac{\sum\limits_{u,i} |R(u,i) - P(u,i)|}{N} \tag{8}$$

where $R(u,i)$ indicates the true value of user u calling service i, and $P(u,i)$ indicates that the missing value N that needs to be predicted is the total number of all the values to be predicted. The *MAE* value represents the average error between the predicted value and the true value; therefore, the smaller the *MAE*, the higher the prediction accuracy.

RMSE is defined as

$$RMSE = \sqrt{\frac{\sum\limits_{u,i} (R(u,i) - P(u,i))^2}{N}} \tag{9}$$

4.3 Experimental setup and result analysis

4.3.1 Experimental setting

We set up our experiments using cross-validation techniques that are widely used in research. For the user-project-time matrix in the WS-Dream dataset, we first randomly remove part of the data for use as the training set according to the matrix density. In the experiment, in order to better reflect the sparseness of historical data in the real world, the matrix density is increased from 0.2 to 0.4 at 0.05 intervals, and the weight is increased from 0.1 to 0.9 at 0.1 intervals. The value of k is increased from 1 to 6 at an interval of 1, and the number of time windows is increased from 5 to 25 at an interval of 5. Using this method, the ability of different algorithms to process datasets with different parameters is evaluated synthetically.

The environment for this part of the experiment is IntelliJ IDEA Community Edition 14.1.4, JDK 1.7, Scala-SDK 2.10.4, and the Scala programming language. The experimental machine is configured with 16 GB memory, a Core i7-4970 3.6 GHz processor, and an Ubuntu 14.04 LTS system.

4.3.2 Prediction accuracy

Table 1 shows the *MAE* and *RMSE* for different prediction methods at different matrix densities, where $\lambda = 0.6$, $T = 10$, and $k = 2$.

The experimental results reported in Table 1 show that the proposed TWQP method can achieve better prediction accuracy in terms of response time and throughput when compared with the UPCC, IPCC, UIPCC, CAPK, TOSEM, and MF methods. The results for the response time dataset show that TWQP achieves improvements in *MAE* of 25.6%, 28.7%, 22.5%, 8.7%, 11.9%, and 7.2%, respectively, and improvements in

Table 1 Comparison of MAE and RMSE between different methods.

Method		Matrix density											
		MAE						*RMSE*					
		0.2	**0.25**	**0.3**	**0.35**	**0.4**	**Average**	**0.2**	**0.25**	**0.3**	**0.35**	**0.4**	**Average**
RT	UPCC	0.926	0.895	0.849	0.781	0.713	0.832	2.115	1.989	1.943	1.868	1.824	1.947
	IPCC	0.954	0.926	0.874	0.825	0.768	0.869	2.328	2.196	2.062	1.973	1.896	2.091
	UIPCC	0.899	0.862	0.815	0.738	0.681	0.799	1.972	1.967	1.915	1.857	1.812	1.904
	CAPK	0.736	0.718	0.695	0.648	0.596	0.678	1.862	1.832	1.801	1.783	1.759	1.807
	TOSEM	0.758	0.735	0.714	0.689	0.623	0.703	1.915	1.899	1.865	1.837	1.798	1.862
	MF	0.727	0.705	0.683	0.639	0.585	0.667	1.854	1.821	1.793	1.776	1.753	1.799
	TWQP	**0.682**	**0.665**	**0.629**	**0.589**	**0.532**	**0.619**	**1.816**	**1.785**	**1.762**	**1.721**	**1.684**	**1.753**
TP	UPCC	9.658	9.296	9.034	8.768	8.531	9.057	39.856	38.626	37.361	36.459	36.026	37.665
	IPCC	9.689	9.322	9.211	9.015	8.646	9.176	41.673	40.652	39.871	38.737	37.895	39.766
	UIPCC	9.565	9.193	8.828	8.645	8.385	8.923	37.984	36.353	35.308	34.823	33.686	35.631
	CAPK	8.071	7.855	7.625	7.228	6.986	7.553	31.984	30.385	29.457	28.819	27.518	29.632
	TOSEM	8.418	8.212	8.068	7.856	7.673	8.045	35.123	34.942	33.706	32.752	31.215	33.547
	MF	7.892	7.726	7.524	7.179	7.057	7.475	30.563	28.859	27.528	26.243	25.586	27.755
	TWQP	**6.834**	**6.644**	**6.272**	**6.028**	**5.729**	**6.301**	**25.118**	**23.215**	**22.057**	**21.256**	**20.037**	**22.336**

RMSE of 9.9%, 16.2%, 7.9%, 2.9%, 5.8%, and 2.6%, respectively. The results for the throughput dataset show that TWQP achieves improvements in *MAE* of 30.4%, 31.3%, 29.3%, 16.5%, 21.7%, and 15.7%, respectively, and improvements in *RMSE* of 40.6%, 43.8%, 37.3%, 24.6%, 33.4%, and 19.5%, respectively. Moreover, with the increase of matrix density from 0.2 to 0.4, the prediction accuracy of the proposed method increases gradually, as with other methods.

4.4 Influence of parameters on prediction accuracy

4.4.1 Matrix density

Table 1 shows a comparison between the proposed method and other methods at different matrix densities. In this section, we discuss the effect of matrix density on the TWQP method. Figures 6a and 6b show the effect of matrix density on prediction accuracy on the response time dataset, while Figs. 6c and 6d show the effect of matrix density on prediction accuracy on the throughput dataset. From the experimental results reported in Fig. 6, it can be seen that matrix density has a significant effect on the prediction accuracy of the TWQP method. In Fig. 6, *MAE* and *RMSE* decrease with an increase in matrix density. This is because the higher the matrix density of the dataset, the more information the users and services contain, which leads to the TWQP obtaining a higher accuracy in the prediction process.

4.4.2 *k* value

Parameter k is the number of similar users and similar services in the prediction process. In general, if the number of similar users and similar services selected in the prediction process is higher, then more information can be used to predict the missing values and further improve the prediction accuracy. In fact, however, this does not necessarily hold.

Figures 7a and 7b show the effect of k on prediction accuracy on the response time dataset, while Figs. 7c and 7d show the effect of k on prediction accuracy on the throughput dataset. The experimental results show that with an increase in the k value, the prediction accuracy improves, but the prediction accuracy decreases continuously after reaching an optimal value. If the value of k is too small in TWQP method, then insufficient information about similar users and similar services is taken account of in the prediction process. As shown in Fig. 7, both *MAE* and *RMSE* begin to decrease as k increases from 1.

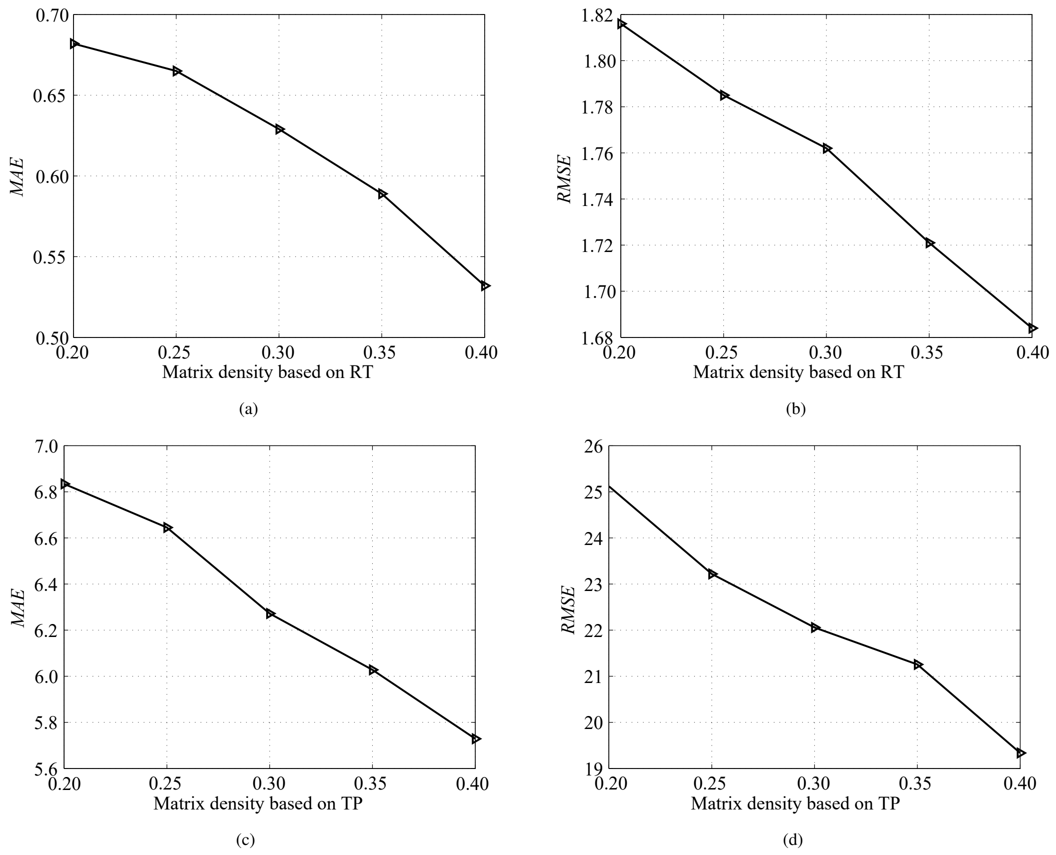


Fig. 6 Effect of matrix density on prediction accuracy.

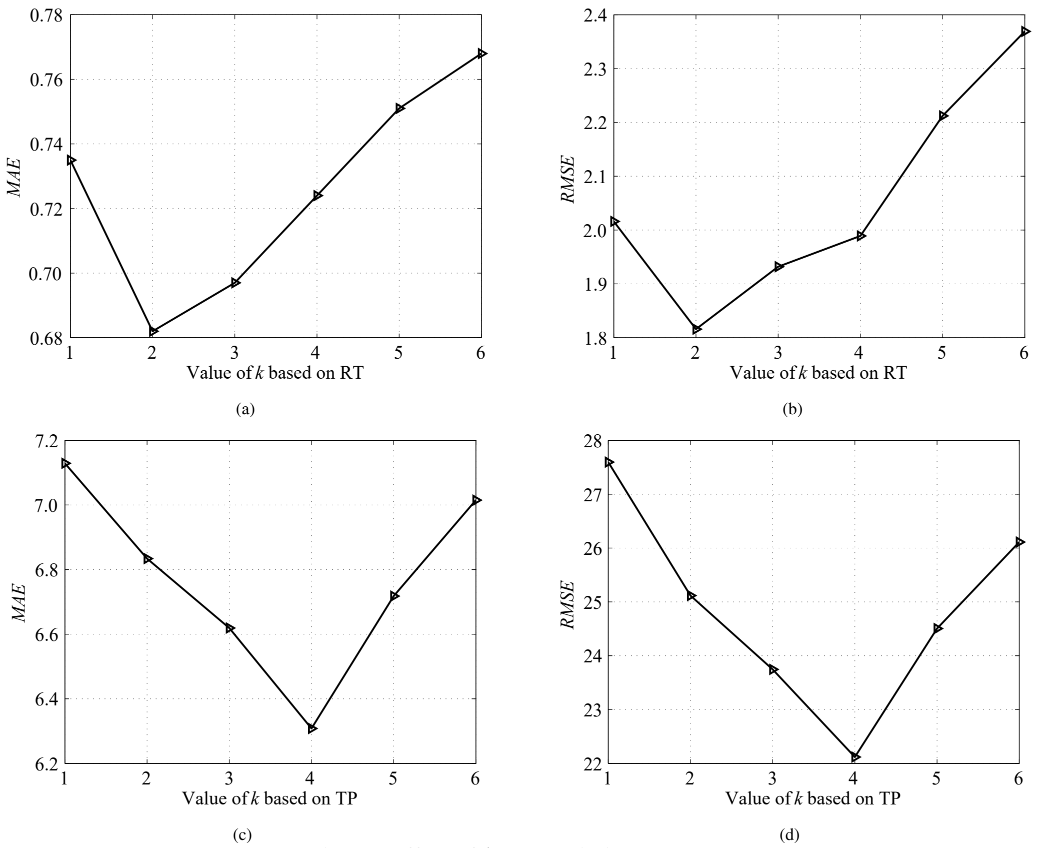


Fig. 7 Effect of *k* on prediction accuracy.

On the other hand, if the value of k is too large, the prediction process begins to take account of users who are not very similar to the target users and services that are not similar to the target services, these dissimilar users and services are noise data and will reduce the prediction accuracy to some extent. Taking Fig. 7a as an example, with k increases from 2 to 6, the predicted *MAE* increases from 0.682 to 0.768. In addition, different datasets can have different optimal k values.

4.4.3 Parameter λ

Parameter λ represents the proportion of similar users and similar services that the TWQP method depends on when predicting missing values in a combined manner. As can be seen from the experimental results in Table 1, the TWQP method outlined in this paper combines the information from similar users and similar services, and improves prediction accuracy. This section evaluates the influence of parameters λ (in the TWQP algorithm). As discussed above, when $\lambda = 0$, TWQP considers only the information from similar services; when $\lambda = 1$, TWQP considers only the information from similar users. In the experiment, the matrix density is 0.2, $k = 2$, and $T = 10$.

Figures 8a and 8b show the effects of λ values on prediction accuracy on the response time dataset, while Figs. 8c and 8d show the effects of λ values on prediction accuracy for the throughput dataset. The experimental results show that the λ value parameter has a significant effect on the prediction accuracy of TWQP. At first, with an increase in the λ value, prediction accuracy increases; However, once the optimal value of prediction accuracy is reached, with an increase in the λ value, the prediction accuracy decreases gradually. This indicates that setting the appropriate λ values allows the algorithm to make full use of the information from similar users and similar services to generate optimal prediction accuracy. The results of the comparison experiment in Fig. 8 also show that the optimal values λ is not fixed and will vary from dataset to dataset. Taking *RMSE* for example, for the response time attribute, the *RMSE* of the predicted result is at its smallest when $\lambda = 0.5$, while for throughput, the predicted *RMSE* is at its smallest when $\lambda = 0.3$.

4.4.4 Number of time windows

In the phase of prediction based on historical time

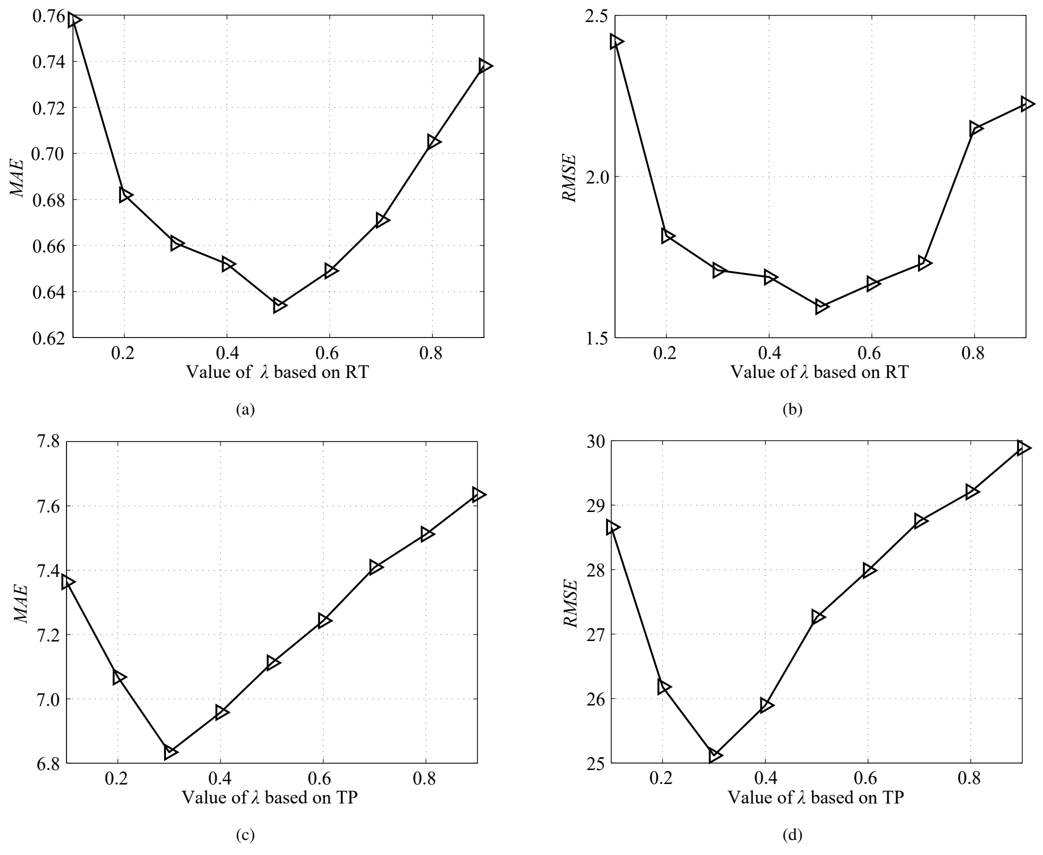


Fig. 8 Effect of weight λ on prediction accuracy.

slices, the missing value of a historical time slice should be predicted by referring to the nearest T time slices' QoS data. Setting a suitable number of time windows T will further improve the prediction accuracy. To study the effect of the number of time windows on prediction accuracy, the matrix density is set to 0.2, $k = 2$, and $\lambda = 0.4$.

Figures 9a and 9b show the effect of the number of time windows on the prediction accuracy on the response time dataset, while Figs. 9c and 9d show the effect of the number of time windows on the prediction accuracy on the throughput dataset. It can be seen from Fig. 9 that the number of time windows has a significant effect on prediction accuracy. With an increase of the number of windows, the prediction accuracy of TWQP method increases gradually. This is because the QoS data in a given historical time slice is limited. Because if the number of time windows increases, information from a greater number of users and services can be fully utilized, which is conducive to further improving the prediction accuracy of the algorithm.

5 Conclusion

In this paper we propose TWQP, a novel time-aware QoS prediction method for services. The TWQP process is made up of two phases: first, a QoS prediction based on historical time slices, and second, a QoS prediction based on current time slice. The procedure for the prediction stage based on historical time slices depends on whether the user has invoked the service in a previous time slice. If so, the QoS value for the user invoking the service on the next time slice is predicted based on the previous time slice; otherwise, when the user has not invoked the service in the previous time slice, then the CA is applied to predict the missing values. In the prediction phase based on the current time slice, the QoS values of all users and services for the current time slice are predicted according to the data from historical time slices. The results of experiments on the real-world WS-Dream datasets demonstrate that TWQP significantly outperforms classical prediction methods for services. Therefore, the paper shows that the prediction problem stemming from the dynamic

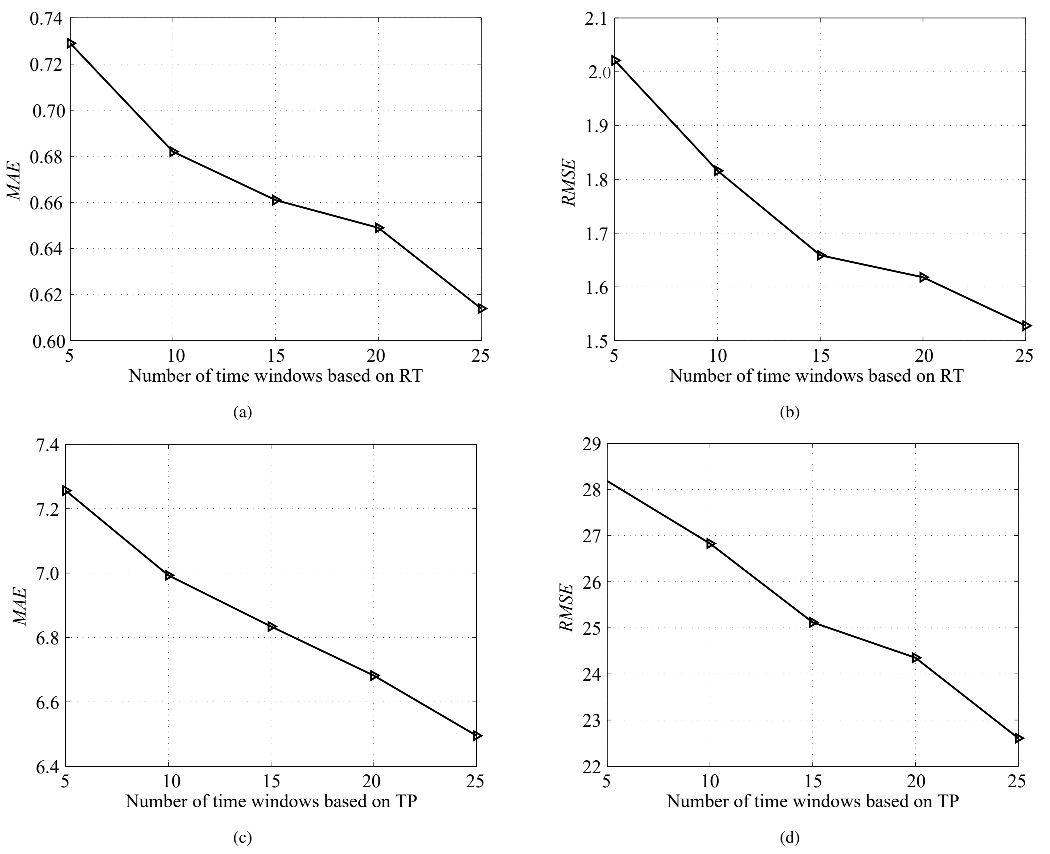


Fig. 9 Effect of the number of time windows on prediction accuracy.

nature of QoS can be solved effectively.

In future work, we will consider the variability of the effects of QoS values at different historical time slices on the QoS values to be predicted. We will seek to identify the most suitable and highest performing algorithm to obtain the weighting of the QoS values of different time slices relative to the final predicted QoS values, and thereby to facilitate more accurate service recommendations.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 61872002), the National Natural Science Foundation of Anhui Province of China (No. 1808085MF197), and the Philosophy and Social Science Planned Project of Anhui Province (No. AHSKY2015D67).

References

[1] J. W. Yin, Y. Tang, W. Lo, and L. Lo, From big data to great services, in *IEEE International Congress on Big Data (Big Data Congress 2016)*, San Francisco, CA, USA, 2016, pp. 165–172.

[2] Y. W. Zhang, G. M. Cui, S. G. Deng, F. F. Chen, Y. Wang, and Q. He, Efficient query of quality correlation for service composition, *IEEE Transactions on Services Computing*, doi: 10.1109/TSC.2018.2830773.

[3] Q. He, J. Han, F. F. Chen, Y. C. Wang, R. Vasa, Y. Yang, and H. Jin, QoS-aware service selection for customizable multi-tenant service-based systems: Maturity and approaches, in *IEEE 8th International Conference on Cloud Computing*, New York, NY, USA, 2015, pp. 237–244.

[4] Y. W. Zhang, G. M. Cui, Y. Wang, X. Guo, and S. Zhao, An optimization algorithm for services composition based on an improved FOA, *Tsinghua Science and Technology*, vol. 20, no. 1, pp. 90–99, 2015.

[5] Y. W. Zhang, G. M. Cui, S. Zhao, and J. Tang, IFOA4WSC: A quick and effective algorithm for QoS aware service composition, *International Journal of Web and Grid Services*, vol. 12, no. 1, pp. 81–108, 2016.

[6] S. G. Deng, H. Y. Wu, D. N. Hu, and J. L. Zhao, Service selection for composition with QoS correlations, *IEEE Transactions on Services Computing*, vol. 9, no. 2, pp. 291–303, 2016.

[7] Y. Yu, J. Chen, S. Q. Lin, and Y. Wang, A dynamic QoS aware logistics service composition algorithm based on social network, *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 4, pp. 399–410, 2014.

[8] F. Dahan, K. El Hindi, and A. Ghoneim, Enhanced artificial bee colony algorithm for QoS-aware web service selection problem, *Computing*, vol. 99, no. 5, pp. 507–517, 2017.

[9] Y. Guo, S. G. Wang, K. S. Wong, and M. H. Kim, Skyline service selection approach based on QoS prediction, *International Journal of Web and Grid Services*, vol. 13,

no. 4, pp. 425–447, 2017.

[10] Y. S. Xu, J. W. Yin, S. G. Deng, N. N. Xiong, and J. B. Huang, Context-aware QoS prediction for web service recommendation and selection, *Expert Systems with Applications*, vol. 53, pp. 75–86, 2016.

[11] K. Su, B. Xiao, B. P. Liu, H. Q. Zhang, and Z. S. Zhang, TAP: A personalized trust-aware QoS prediction approach for web service recommendation, *Knowledge-Based Systems*, vol.115, pp. 55–65, 2017.

[12] S. G. Deng, D. G. Wang, Y. Li, B. Cao, J. W. Yin, Z. H. Wu, and M. C. Zhou, A recommendation system to facilitate business process modeling, *IEEE Transactions on Cybernetics*, vol. 47, no. 6, pp. 1380–1394, 2017.

[13] Y. W. Zhang, Y. Y. Zhou, F. T. Wang, Z. Sun, and Q. He, Service recommendation based on quotient space granularity analysis and covering algorithm on Spark, *Knowledge-Based Systems*, vol. 147, pp. 25–35, 2018.

[14] X. Chen, Z. B. Zheng, Q. Yu, and M. R. Lyu, Web service recommendation via exploiting location and QoS information, *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1913–1924, 2014.

[15] Z. B. Zheng, H. Ma, M. R. Lyu, and I. King, QoS-aware web service recommendation by collaborative filtering, *IEEE Transactions on Services Computing*, vol. 4, no. 2, pp. 140–152, 2011.

[16] L. Si and R. Jin, Flexible mixture model for collaborative filtering, in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, Washington, DC, USA, 2003, pp. 704–711.

[17] X. Luo, M. C. Zhou, Y. N. Xia, and Q. S. Zhu, An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems, *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1273–1284, 2014.

[18] L. S. Shao, J. Zhang, Y. Wei, J. F. Zhao, B. Xie, and H. Mei, Personalized QoS prediction for web services via collaborative filtering, in *IEEE International Conference on Web Services (ICWS 2007)*, Salt Lake City, UT, USA, 2007, pp. 439–446.

[19] G. Linden, B. Smith, and J. York, Amazon.com recommendations: Item-to-item collaborative filtering, *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[20] Z. B. Zheng, H. Ma, M. R. Lyu, and I. King, WSRec: A collaborative filtering based web service recommender system, in *IEEE International Conference on Web Services (ICWS 2009)*, Los Angeles, CA, USA, 2009, pp. 437–444.

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

[22] M. Zhang, X. D. Liu, R. C. Zhang, and H. L. Sun, A web service recommendation approach based on QoS prediction using fuzzy clustering, in *2012 IEEE Ninth International Conference on Services Computing (SCC)*, Honolulu, HI, USA, 2012, pp. 138–145.

[23] Z. B. Zheng, H. Ma, M. R. Lyu, and I. King, Collaborative web service QoS prediction via neighborhood integrated matrix factorization, *IEEE Transactions on Services Computing*, vol. 6, no. 3, pp. 289–299, 2013.

[24] C. Y. Yu and L. P. Huang, CluCF: A clustering CF algorithm to address data sparsity problem, *Service Oriented Computing and Applications*, vol. 11, no. 1, pp. 33–45, 2017.

[25] L. Zhang and B. Zhang, A geometrical representation of McCulloch-pitts neural model and its applications, *IEEE Transactions on Neural Networks*, vol. 10, no. 4, pp. 925–929, 1999.

[26] T. Wu, L. Zhang, and Y. P. Zhang, Kernel covering algorithm for machine learning, *Chinese Journal of Computers*, vol. 28, no. 8, pp. 1295–1301, 2005.

[27] L. Zhang and B. Zhang, A geometrical representation of M-P neural model and its applications, *Chinese Journal of Software*, vol. 9, no. 5, pp. 334–338, 1998.

[28] Y. L. Zhang, Z. B. Zheng, and M. R. Lyu, WSPred: A time-aware personalized QoS prediction framework for services, in *2011 22nd IEEE International Symposium on Software Reliability Engineering (ISSRE)*, Hiroshima, Japan, 2011, pp. 210–219.

[29] Z. Zheng and M. R. Lyu, Personalized reliability prediction of web services, *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 22, no. 2, pp. 1–25, 2013.

[30] C. Wu, W. Qiu, Z. Zheng, X. Wang, and X. Yang, QoS prediction of web services based on two-phase k-means clustering, in *IEEE International Conference on Web Services (ICWS 2015)*, New York, NY, USA, 2015, pp. 161–168.

Ying Jin received the master degree in 2009 from Hefei University of Technology. She is an associate professor in the Department of Management of Hefei University. Her research interests include service computing, data mining, and web service.

Weiguang Guo received the master degree in 2008 from Heifei University of Technology. He is an associate professor in the Department of Management of Hefei University. His research interests include data mining and data analysis.

Yiwen Zhang received the master degree in 2006 and the PhD degree in 2013 from Hefei University of Technology. He is a professor in the School of Computer Science and Technology at Anhui University. His research interests include service computing, cloud computing, and e-commerce.