A Highly Accurate Prediction Algorithm for Unknown Web Service QoS Values

You Ma, Shangguang Wang, *Member, IEEE*, Patrick C.K. Hung, *Member, IEEE*, Ching-Hsien Hsu, *Member, IEEE*, Qibo Sun, and Fangchun Yang, *Senior Member, IEEE*

Abstract—Quality of Service (QoS) guarantee is an important component of service recommendation. Generally, some QoS values of a service are unknown to its users who has never invoked it before, and therefore the accurate prediction of unknown QoS values is significant for the successful deployment of Web service-based applications. Collaborative filtering is an important method for predicting missing values, and has thus been widely adopted in the prediction of unknown QoS values. However, collaborative filtering originated from the processing of subjective data, such as movie scores. The QoS data of Web services are usually objective, meaning that existing collaborative filtering-based approaches are not always applicable for unknown QoS values. Based on real world Web service QoS data and a number of experiments, in this paper, we determine some important characteristics of objective QoS datasets that have never been found before. We propose a prediction algorithm to realize these characteristics, allowing the unknown QoS values to be predicted accurately. Experimental results show that the proposed algorithm predicts unknown Web service QoS values more accurately than other existing approaches.

Index Terms—Web Service, QoS prediction, collaborative filtering, objective data, service recommendation

1 Introduction

Web services are software components designed to support interoperable machine-to-machine interaction over a network [1]. With the rapid growth in the number of such services, users are faced with a massive set of candidate services in their service selection, and thus effective and efficient service recommendation is essential to determining the most suitable service component. For most recommender systems, Quality of Service (QoS) is an important basis to judge whether a service is suitable to recommend.

QoS data compose a set of non-functional properties, with each property characterizing a certain aspect of service quality [2]. Some QoS properties are user-independent, having identical values for different users (e.g., price, popularity, availability) while other QoS properties are user-dependent (e.g., response time, invocation failure rate). If the recommender system suggests user a service that has not been used before, the user-dependent QoS values are unknown for this user. In this case, predicting the unknown QoS values is very important to determine whether this service is a suitable recommendation.

As the most popular and successful recommendation method, Collaborative Filtering (CF) have been widely used in many popular commercial recommender systems such as YouTube, Reddit, Last.fm and Amazon etc[1]. The core work of CF in such recommender systems is predicting the unknown rating values. For example, whether a book or a movie should be recommended to a user. It needs an accurate prediction of the future user-book or user-movie rating. In fact, Web service recommender systems should do the similar prediction works as YouTube or Amazon does, the difference is nothing but what they predict are the unknown QoS values of future user-service invocations. Inspired by the successes of CF achieved by existing commercial recommender systems, many studies have used CF-based methods to predict unknown QoS values [3], [4], [5], [6], [7], [8], [9], [45], [46], [47]. Such methods generally consist of two steps: 1) finding similar users or items and mining their similarities; and 2) calculating unknown QoS values according to the known data of similar users or items.

CF ideas originated from the processing of subjective data [10], [11]. Four famous datasets—Netflix [12], MovieLens[2] [13], CiteULike[3] [14], and Delicious [15]—that have often been used in CF studies are all composed of subjective data from users. Netflix and MovieLens consist of movie scores as given by users. CiteULike is a set of tags that are marked by users according to their favorite articles, and Delicious is a set of tags marked for different Web pages. The commercial recommender systems mentioned above mainly process subjective data too. Most existing QoS prediction methods are inspired by these CF ideas, for which we call *traditional CF* methods to distinguish our proposed algorithm.

However, QoS data related to Web services is objective, and we have found there are some significant differences between subjective and objective data, which may bring errors to the prediction of unknown QoS values with tra-

- *M. You, S. G. Wang, Q. B. Sun and F. C. Yang are with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: mayou0531@126.com; {sgwang; qbsun; fcyang}@bupt.edu.cn.*
- *P. Hung is with the Faculty of Business and IT, University of Ontario Institute of Technology, 2000 Simcoe Street North, Oshawa, ON, Canada L1H 7K4. E-mail: patrik.hung@uoit.ca*
- *CH Hsu is with the Computer Science and Information Engineering, Chung Hua University, Hsinch, Taiwan. E-mail: chh@chu.edu.tw*

[1] www.readwriteweb.com/archives/collaborative_filtering_social_web.php
[2] http://grouplens.org/datasets/movielens/
[3] http://www.citeulike.org/

ditional CF manners. To explain the reason easily we present two definitions as follow.

Definition 1: ***Subjective Data***—This refers to values associated with users' subjective evaluations, and comes directly from users. This type of data is influenced by users' cognitive level, interests, and other subjective factors. For example, the movie scores of MovieLens and the commodity ratings of Amazon are subjective data.

Definition 2: Objective Data—This includes values related to objective characteristics of a service for a user, such as the response time, the throughput, the reliability. This type of data does not come directly from users, but is determined as a result of some objective factors related to users. For example, a user will get a response time when he invokes a Web service. However he cannot control the value of the response time since it is determined as a result of the objective factors—network traffic, bandwidth, etc.

An important difference between subjective and objective data is that for the former two high similar users often give similar values for an item while it is not the case for the latter. We found this difference by a contrasting observation between MovieLens (subjective data) and WSDream[1] (objective data). We searched in MovieLens for all the user pairs wherein each user pair has Pearson Correlation Coefficient (PCC) similarity greater than 0.8 and shares at least 10 common movies, and 26671 such user pairs were found. We use UP_M to denote these user pairs. We also searched in the throughput dataset of WSDream for all the user pairs wherein each user pair has a PCC similarity greater than 0.8 and shares at least 10 common Web services, and 16320 such user pairs were found. We use UP_W to denote these user pairs. We wanted to see if two users have a high PCC similarity, whether they would have a similar experience[2] on the same item. If we let (u_a, u_b) denote a user pair where u_a and u_b are the two users of this pair. Let *CI* denote the common items that u_a and u_b share. Let M_a denote the mean value of all the user-item scores of *CI* observed from u_a. M_b is defined by the similar way. Let $GAP = \frac{|M_a - M_b|}{min(M_a, M_b)}$ denote the overall gap between u_a and u_b. Therefore a *GAP* can presents the overall experience difference between two users for the same items they all experienced. We have made a statistic of the proportions of the number of user pairs in different *GAP* ranges for UP_M and UP_W, shown as Fig. 1.

From Fig. 1 we can see that UP_M has many more user pairs than UP_W in *GAP*s of [0, 0.1] and (0.1, 0.2], while UP_W has more user pairs than UP_M in other *GAP*s where the two users of one pair may have very different experiences on one item. Also worth attention is there are 9.99% user pairs of UP_W have *GAP*s greater than 1, which means that these user pairs may bring errors greater than 100% in the predictions of traditional CF. Fig. 1 suggests us that if two users of MovieLens have a big similarity then they are likely to make similar evaluation for their common movies, while two users of WSDream may get very different QoS values on the same Web services even they have a big similarity.

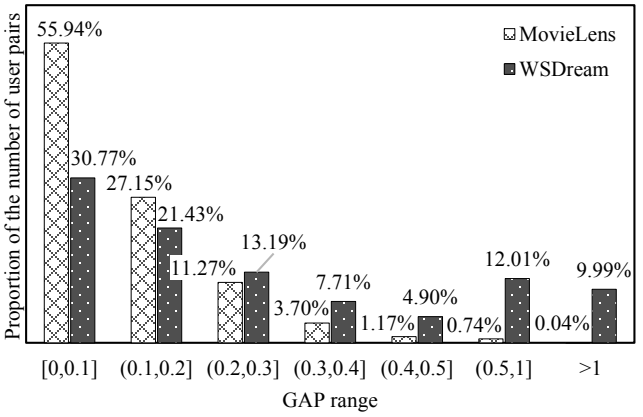


Fig. 1. Distribution proportions of user pairs in different *GAP*s. If two users of MovieLens have a big similarity, they are more likely to make similar evaluations for the same movies, while two users of WSDream may observe very different QoS values on their commonly invoked Web services even they have a high similarity.

We have also observed the item pairs in MovieLens and WSDream by the similar way above, which suggests us that even the high similar Web services may give very different QoS values to same users.

Why are MovieLens and WSDream such different as Fig. 1 showed? Our preliminary speculation suggests that, with subjective datasets, such as movie scores, if two audiences have seen many similar movies, then they have similar tastes, and therefore their evaluation of a movie will be similar. But it isn't true for objective data, which is not created by user. Objective data (e.g., Web service QoS values) is created by objective factors beyond user.

For unknown Web service QoS values prediction, most existing CF methods are based the following two equations:

$$r'_{u,i} = \bar{u} + \frac{\sum_{v \in SU} sim(u,v) \times (r_{v,i} - \bar{v})}{\sum_{v \in SU} sim(u,v)} \quad (1)$$

$$r'_{u,i} = \bar{i} + \frac{\sum_{j \in SI} sim(i,j) \times (r_{u,j} - \bar{j})}{\sum_{j \in SI} sim(i,j)} \quad (2)$$

where (1) is user-based CF, (2) is item-based CF, $r_{v,i}$ is the QoS value of service i observed by user v, $r'_{u,i}$ denotes the prediction of the unknown QoS value $r_{u,i}$, SU is the set that consists of all users similar to user u, $\bar{u}$ is the average QoS value of all the services accessed by u, SI is the set of all services similar to service i, $\bar{i}$ is the average QoS value of i according to all users, $sim(u,v)$ is the similarity between users u and v, and $sim(i,j)$ is the similarity between services i and j. The above two equations (or their improved transformations) play an important role in existing CF-based methods.

However, (1) and (2) work well only if there is no GAP problems. Take (1) for example, if user u and v have a

[1]WSDream consists of values of response time and throughput of real world Web service invocations by different users. Detailed information can be found at http://www.wsdream.net/dataset.html

[2]In this paper, the user-item experience only denotes the user-item value of that dataset.

large GAP, then $\bar{u}$ and $\bar{v}$ would be very different, and the prediction value $r'_{u,i}$ would be unreasonable. Most existing CF-based methods either just employ this two equations directly [4], [6], [8], [47] or make some reforms but without substantial improvements [3], [5], [7], [9], [45], [46].

Generally, normalization methods are often used to avoid the GAP problem, but we cannot do this for Web service QoS data because another important difference between subjective and objective data. Normalization methods are used mainly if we know the maximum and minimum values of the data. In most subjective datasets, all the values are fixed in a known value scope. For example, all the rating scores of MovieLens are fixed in the scope of [1, 5]. However, it is difficult to determine the QoS values scope of Web services since the constantly appearing of new added users and Web services, and the scope of QoS value often changes. We have made statistics on the changed QoS value scope for WSDream. There are 1831253 and 1873838 QoS values (got from 339 users on 5825 Web services, and the missing values are eliminated) in the two of WSDream datasets, i.e., throughput and response time dataset respectively. We surveyed how much the new added QoS values exceed the old QoS value scopes before it added. If a Web service give a suddenly changed QoS value for its new added user (from $user_0$ to $user_{338}$) and the new QoS value exceed its old value scope more than 100% then this value is recorded. There are total 12827 and 12962 such QoS values for all the Web services in throughput and response time datasets, respectively. This means that each user would encounter about 40 Web services on average who give them a suddenly changed QoS value. Notice that the original WSDream datasets are very dense which reduces the changes of QoS values scope. In practice, the datasets are often spare and the changes of QoS value scope are more serious with the appearing of new users and Web services. Since we cannot known the fixed QoS value scope, the normalization methods are difficult to be employed to avoid the GAP problem.

From what is mentioned above, traditional CF may not work well for Web service QoS prediction, due to the GAP problem which is difficult to avoid by normalization methods. Table 1 illustrates how traditional CF lose efficacy in such case.

TABLE 1
SOME THROUGHPUT VALUES IN WSDREAM

	WS_0	WS_1	WS_2	WS_3	WS_4	WS_5	WS_6	WS_7
U_{110}	0.798	5.134	7.623	2.583	2.684	4.065	0.337	47.24
U_{122}	0.318	4.347	6.437	2.171	2.322	0.52	0.321	43.816
U_{123}	0.318	4.489	6.795	2.309	2.293	0.571	0.324	43.337
U_{124}	0.328	4.362	7.009	2.229	2.247	0.579	0.321	43.673
U_{242}	0.965	13.513	19.672	7.042	7.017	2.676	1.661	112.277

Table 1 presents some throughput values in WSDream observed by 5 users on 8 real-world Web services from the WSDream dataset. U_i denotes the i-th user and WS_j denotes the j-th Web service. If we assume user U_{242} has never invoked service WS_7, then the throughput value is unknown, and it is predicted to be 49.33 by (1) and 49.22 by (2). The prediction errors are caused by the GAPs between U_{242} and other users, also the GAPs between WS_7 and other Web services. However, it is difficult to avoid the GAP problem by normalizing the QoS values since the U_{242} - WS_7 invocation will give a throughput value exceedingly beyond the old QoS value scopes of U_{242} ([0.965, 19.672]) and WS_7 ([43.337, 112.277]).

Web service QoS values are objective and have no subjective features, therefore we cannot predict the unknown objective QoS values as for movie scores.

The existing CF based prediction methods for unknown QoS values have not realized the above differences between subjective and objective data and therefore cannot predict objective QoS values accurately. In allusion to this problem this paper presents a highly accurate prediction algorithm (HAPA) for unknown Web service QoS values. HAPA is also CF-based, i.e. also use similar users and similar items to make prediction, but with fundamental changes from traditional CF approaches to adapt to the characteristics of objective QoS data. Fig.1 points out one of the characteristics of objective QoS data, i.e. a high similarity does not means that two users or items present similar QoS values, but this characteristic doesn't tell us how predict unknown QoS values. Therefore we have to explore other characteristics of objective QoS data which are the theoretical basis of HAPA. These characteristics will tell us what the similarity means to objective QoS data, which can be summarized as: 1) if two users share a high similarity, then their similarity will hardly fluctuate with their future invocations of more numbers of Web services, and 2) in the same way, if two items share a high similarity, then their similarity will hardly fluctuate with their future invocations by more numbers of users. We built our HAPA for the prediction of the unknown Web service QoS values based on these characteristics which were found by many experiments.

The main contributions of this paper are threefold. First, based on real Web service QoS data and a number of experiments, we find some important characteristics of objective QoS datasets that have never been found before. Second, we propose a Web service QoS value prediction algorithm HAPA to realize these characteristics, allowing the unknown QoS values to be predicted accurately. Finally, we conduct several real world experiments to verify our prediction accuracy.

The rest of this paper is organized as follows. Section 2 presents our highly accurate prediction algorithm for unknown Web service QoS values and also analyze its computational complexity. Section 3 describes our experiments. Section 4 shows related work, and Section 5 concludes the paper.

2 PROPOSED HAPA

HAPA is a CF-based algorithm, i.e., it concretely consists of user-based and item-based HAPAs. Each type of HAPA can make predictions, however we always use the combination of the two HAPAs to make predictions more

accurate.

This section first presents the theoretical foundation of HAPA, then introduce the mechanisms of user-based and item-based HAPAs, and finally describe how make predictions employing the combination of the two HAPAs.

2.1 Theoretical Foundation of HAPA

In this section we carry out some analysis of the characteristics of QoS data based on which we propose our HAPA.

We use PCC to measure the similarity between users or items. For user-based CF methods, PCC uses the following equation to calculate the similarity between two users u and v based on the Web services they commonly invoke:

$$sim(u,v)=\frac{\sum_{i\in I}\left(r_{u,i}-\overline{r_u}\right)\left(r_{v,i}-\overline{r_v}\right)}{\sqrt{\sum_{i\in I}\left(r_{u,i}-\overline{r_u}\right)^2}\sqrt{\sum_{i\in I}\left(r_{v,i}-\overline{r_v}\right)^2}} \tag{3}$$

where $I=I_u\cap I_v$ is the subset of Web service previously invoked by both user u and v. $r_{u,i}$ is the QoS value of Web service i observed by user u, and $\overline{r_u}$ represents the average QoS value of different Web services observed by user u.

Similarly, for item-based CF, the PCC between two service items is calculated as:

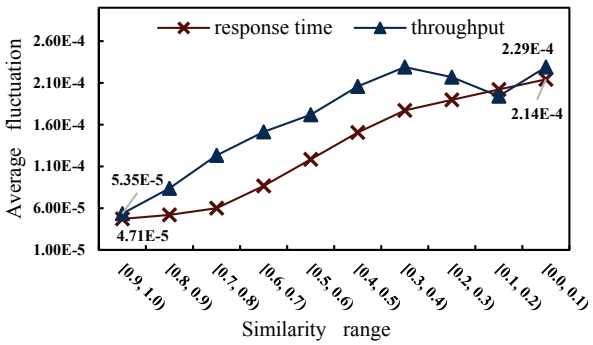
$$sim(i,j)=\frac{\sum_{u\in U}\left(r_{u,i}-\overline{r_i}\right)\left(r_{u,j}-\overline{r_j}\right)}{\sqrt{\sum_{u\in U}\left(r_{u,i}-\overline{r_i}\right)^2}\sqrt{\sum_{u\in U}\left(r_{u,j}-\overline{r_j}\right)^2}} \tag{4}$$

where $U=U_i\cap U_j$ is the subset of users who have previously invoked both Web services i and j, and $\overline{r_i}$ represents the average QoS value of Web service i observed by different users.

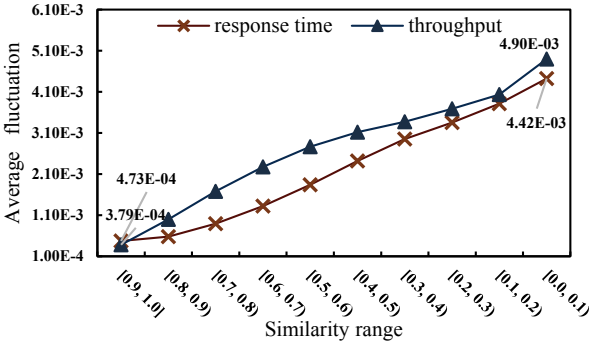
We adopt WSDream datasets to carry out analysis. WSDream contains two 339×5825 user-item matrixes. Each value is the response time for a user invoking a Web service in a matrix, and the throughput for a user invoking a Web service in another matrix.

We want to study how the similarity between two users or Web services changes with the growing user-service invocations. In this study, we found that if two users have a high similarity, then their similarity will hardly fluctuate with their growing invocations of Web services. On the contrary, if two users have only a low similarity, then their similarity would fluctuate notably with their growing invocations of Web services. The same phenomenon can be found for the similarity between Web services. If two Web services have a high similarity, then their similarity will hardly fluctuate with their growing invocations by users. Conversely, if two Web services have only a low similarity, then their similarity would fluctuate notably with their growing invocations by users. We have made statistics of all user and Web service pairs on the entire WSDream datasets to verify these phenomenon, which is shown as Fig. 2. We got Fig. 2 as the following five steps:

1) Randomly remove some invocation values from the two datasets of WSDream until only 5% invocation entries left. (As we can't know the actual situations of the future user-service invocations, we deleted some invocations from the dataset. Then the dataset with more deletions denotes "the past", and the dataset with less or no deletions denotes "the future".)
2) We divide the two datasets of WSDream (it is very sparse now) on the similarity basis, i.e., [0, 0.1), …, [0.9, 1]. For each similarity range, we find out all the user pairs whose similarity belong to this range.
3) For each user pair (u, v), we randomly add a Web service to $I_u\cap I_v$. We can know this service's QoS values (for user u and v) from the original dataset, and then the change of similarity $sim(u,v)$ is calculated out. The absolute value of the difference in similarity is used to denote the fluctuation. This step is repeated until all Web services are added to $I_u\cap I_v$.
4) We calculated the average fluctuation of similarity of all user pair for each similarity range.
5) We treat all the service pairs of each similarity range in the similar way above.



(a) For all user pairs of WSDream



(b) For all service pairs of WSDream

Fig. 2. Average fluctuation in each similarity range, which suggests us a characteristic of PCC similarity: the higher the similarity the steadier the similarity and vice versa.

Since Step 1 make the datasets sparse of 5% density, Fig. 2 can represents the feature of WSDream in different matrix density from 5% to about 90% (The density of original response time and throughput datasets are 94.9% and 92.7% respectively), because the density increases with the adding process.

From Fig. 2 we can see that if user u and v have a high similarity, then this similarity will hardly change with their future invocations of more Web services, and in the

same way, the higher the similarity two services have, the steadier the similarity with their future invocations by more numbers of users.

Then we propose the following two characteristics of QoS data:

Characteristic 1: For two pairs of service users (u,v) and (u',v'), if $sim(u,v)$ is considerably greater than $sim(u',v')$, then $sim(u,v)$ will fluctuate less than $sim(u',v')$ when they invocate more Web services.

Characteristic 2: For two pairs of services (i,j) and (i',j'), if $sim(i,j)$ is considerably greater than $sim(i',j')$, then $sim(i,j)$ would fluctuate less than $sim(i',j')$ when they are invocated by more users.

We can use Characteristics 1 and 2 to derive Corollaries 1 and 2, respectively:

Corollary 1: If we use a high similar user *v* to predict the unknown QoS value $r_{u,i}$, then $sim(u,v)$ should hardly change after the prediction of $r_{u,i}$. This can be explained by Characteristic 1, i.e., predicting $r_{u,i}$ is equivalent to adding a service to $I_u \cap I_v$.

Corollary 2: If we use a high similar service *j* to predict the unknown value $r_{u,i}$, then $sim(i,j)$ should hardly change after the prediction of $r_{u,i}$. This can be explained by Characteristic 2, i.e., predicting $r_{u,i}$ is equivalent to adding a user to $U_i \cap U_j$.

These corollaries are the theoretical foundation of our proposed algorithm HAPA. HAPA includes user-based and item-based prediction according to Corollaries 1 and 2, respectively. A detailed explanation of the prediction mechanism is given in Sections 2.2 and 2.3, respectively. To aid the following descriptions, we first give some definitions.

Definition 3: Active User—The active user is one who requires a prediction for an unknown QoS value.

Definition 4: Similar User—If an active user *u* requires the unknown QoS value $r_{u,i}$ and $r_{v,i}$ is already known, then service user *v* is a *similar user* to service user *u*.

Definition 5: Similar Service—If an active user *u* requires the unknown QoS value $r_{u,i}$ and $r_{u,j}$ is already known, then service *j* is a *similar service* to service *i*.

2.2 User-based HAPA

User-based HAPA consists of three steps: 1) a *user-based prediction equation* is constructed based on each similar user to mathematically implement Corollary 1; 2) these prediction equations are proved to be quadratic, that produce two prediction values, in which only one value is appropriate, and therefore linear regression is used to select the appropriate prediction value for each prediction equation; 3) the unknown QoS value is calculated from all the prediction values made by every similar user.

2.2.1 User-based Prediction Equation

The *user-based prediction equation* is a mathematical formulation of Corollary 1. Before constructing these equations, we first find the top similar users with big similarities. Let *user_topK* denote the number of these top similar users, and let *KU* represent the set of these top similar users. In Section 3.4 we study how big the similarity is appropriate by experiments and it determines *user_topK*.

Suppose $r_{u,i}$ is the unknown QoS value, and let *x* represent the predicted value of $r_{u,i}$. User *su* is a similar user to *u*. To establish the prediction equation based on user *su*, we assume that user *u* has invoked service *i*, and let *x* be the QoS value for this invocation. When the invocation has finished, we recalculate the similarity between user *u* and *su*. Let the new similarity be $sim'(u,su)$. Inspired by Corollary 1, we know that the new similarity $sim'(u,su)$ should be approximately the same as the old similarity $sim(u,su)$. Thus, we have:

$$sim'(u,su) = sim(u,su) \tag{5}$$

where

$$sim'(u,su) = \frac{\sum_{e\in I'}\left(r_{u,e}-\overline{r_u'}\right)\left(r_{su,e}-\overline{r_{su}'}\right)}{\sqrt{\sum_{e\in I'}\left(r_{u,e}-\overline{r_u'}\right)^2}\sqrt{\sum_{e\in I'}\left(r_{su,e}-\overline{r_{su}'}\right)^2}} \tag{6}$$

where $I' = I \cup \{\text{service } i\}$, because we have assumed that user *u* has invoked service *i*. Let *x* be the QoS value of service *i* for this invocation by user *u*. $\overline{r_u'}$ and $\overline{r_{su}'}$ are the average QoS values of different services observed by users *u* and *su*, respectively. Obviously, $\overline{r_u'}$ is a function of *x*, written as:

$$\overline{r_u'} = \frac{\overline{r_u}\times|I|+x}{|I|+1} \tag{7}$$

Therefore, (5) is a function of *x*, and *x* is the only variable in (5). Hence, to predict $r_{u,i}$, we solve for *x* to satisfy (5).

Substituting the data of users *u* and *su* into (6) and substituting (6) into (5) give:

$$\frac{e\cdot x+f}{d\sqrt{a\cdot x^2+b\cdot x+c}} = sim(u,su) \tag{8}$$

where the constant coefficients *a, b* and *c* are determined by user *u*, the constant coefficient *d* is determined by the similar user *su*, constant coefficients *e* and *f* are commonly determined by user *u* and the similar user *su*, and the original similarity $sim(u,su)$ is already known. (8) is a *user-based prediction equation.*

We can easily transform (8) to a quadratic equation. The predicted value of $r_{u,i}$ must be one of the two roots of (8), and we use linear regression to select the appropriate root.

2.2.2 Linear Regression for User-based Prediction Equation

Linear regression can be used to predict a vector based on another linearly related vector. As we are using PCC to measure the similarity and choosing highly similar users to make predictions, user *u* and the similar user *su* should

have a strong linear relationship. Therefore, we can use linear regression to make a rough prediction, and then use the rough prediction to select the appropriate root.

We establish the linear regression model as:

$$\begin{cases} b_1 = \dfrac{\sum_{s\in I} r_{u,s}\cdot r_{su,s} - |I|\cdot \overline{r_u}\cdot \overline{r_{su}}}{\sum_{s\in I} r_{su,s}^{\ 2} - |I|\cdot \overline{r_{su}}^{2}} \\ b_0 = \overline{r_u} - b_1\cdot \overline{r_{su}} \end{cases} \tag{9}$$

where b_0 and b_1 represent coefficients of the linear relationship between user u and similar user su. Using the two coefficients, we obtain a rough prediction of $r_{u,i}$ as:

$$rp_{su}\left(r_{u,i}\right) = b_1\cdot r_{su,i} + b_0 \tag{10}$$

where $rp_{su}\left(r_{u,i}\right)$ is the rough prediction of $r_{u,i}$ made by similar user su.

Let x_1 and x_2 denote the two roots of the user-based prediction equation (8). We select the appropriate root as follows:

$$\begin{cases} pre_{su}(r_{u,i}) = x_1, & \text{if } \left|x_1 - rp_{su}\left(r_{u,i}\right)\right| < \left|x_2 - rp_{su}\left(r_{u,i}\right)\right| \\ pre_{su}(r_{u,i}) = x_2, & \text{else} \end{cases} \tag{11}$$

where $pre_{su}(r_{u,i})$ is the predicted value of $r_{u,i}$ made by the the similar user su.

2.2.3 Final Prediction for User-based HAPA

After each similar user in KU calculating out a prediction value using its prediction equation and linear regression, we would have $user\_topK$ prediction values. Now we should determine the final accurate prediction by comprehensively considering all the similar users of KU as follow:

$$\hat{r}_{u,i} = \sum_{su\in KU} pre_{su}(r_{u,i})\cdot con(su) \tag{12}$$

where $\hat{r}_{u,i}$ is the final prediction of $r_{u,i}$, and $con(su)$ denotes the confidence in the prediction given by su. Because a higher similarity generally gives greater stability, we will obtain a more accurate prediction from users with a higher similarity. Therefore, we have more confidence in users who have a higher similarity to user u. This can be written as:

$$con(su) = \frac{sim(u,su)}{\sum_{v\in KU} sim(u,v)} \tag{13}$$

2.3 Item-based HAPA

In the paper, an item is namely a Web service. Item-based HAPA uses similar services to make predictions. Its mathematical principle is very similar to that of user-based HAPA.

2.3.1 Item-based Prediction Equation

Let $r_{u,i}$ represent the unknown QoS value, and let x denote the predicted value of $r_{u,i}$. To predict $r_{u,i}$ in an item-based manner, we should first determine the $item\_topK$ most similar service. Let KI represent the set of $item\_topK$ most similar services.

For an arbitrary similar service si ($si \in KI$), if we make a prediction for $r_{u,i}$ based on si, we find an equivalent to (5) based on Corollary 2:

$$sim'(i,si) = sim(i,si) \tag{14}$$

where $sim(i,si)$ is the original similarity between service i and its similar service si, and $sim'(i,si)$ is the similarity after the prediction of $r_{u,i}$:

$$sim'(i,si) = \frac{\sum_{v\in U'}\left(r_{v,i} - \overline{r_i'}\right)\left(r_{v,si} - \overline{r_{si}'}\right)}{\sqrt{\sum_{v\in U'}\left(r_{v,i} - \overline{r_i'}\right)^2}\sqrt{\sum_{v\in U'}\left(r_{v,si} - \overline{r_{si}'}\right)^2}} \tag{15}$$

where $U' = U \bigcup \{\text{service user } u\}$, as we have assumed that user u has invoked service i and that x is the QoS value of service i for this invocation by user u. $\overline{r_i'}$ and $\overline{r_{si}'}$ are the average QoS values of service i and si observed by different users. $\overline{r_i'}$ is a function of x, which can be written as:

$$\overline{r_i'} = \frac{\overline{r_i}\cdot|U| + x}{|U| + 1} \tag{16}$$

Substituting the data of service i and si into (15) and substituting (15) to (14) give:

$$\frac{e\cdot x + f}{d\sqrt{a\cdot x^2 + b\cdot x + c}} = sim(i,si) \tag{17}$$

Equation (17) is an *item-based prediction equation,* to which linear regression is applied to determine the appropriate root.

2.3.2 Linear Regression for Item-based Prediction Equation

We establish the linear regression model as:

$$\begin{cases} b_1 = \dfrac{\sum_{v\in U} r_{v,i}\cdot r_{v,si} - |U|\cdot \overline{r_i}\cdot \overline{r_{si}}}{\sum_{v\in U} r_{v,si}^{\ 2} - |U|\cdot \overline{r_{si}}^{2}} \\ b_0 = \overline{r_i} - b_1\cdot \overline{r_{si}} \end{cases} \tag{18}$$

The rough prediction of $r_{u,i}$ made by similar service si can be obtained from:

$$rp_{si}\left(r_{u,i}\right) = b_1\cdot r_{u,si} + b_0 \tag{19}$$

Let x_1 and x_2 denote the two roots of the item-based prediction equation. The appropriate root is selected according to:

$$\begin{cases} pre_{si}(r_{u,i}) = x_1, & \text{if } \left|x_1 - rp_{si}\left(r_{u,i}\right)\right| < \left|x_2 - rp_{si}\left(r_{u,i}\right)\right| \\ pre_{si}(r_{u,i}) = x_2, & \text{else} \end{cases} \tag{20}$$

where $pre_{si}(r_{u,i})$ is the predicted value of $r_{u,i}$ given by the similar service si.

2.3.3 Final Prediction for Item-based HAPA

After every similar item in KI has predicted the unknown QoS value $r_{u,i}$, we have the final prediction as:

$$\hat{r}_{u,i} = \sum_{si\in KI} pre_{si}(r_{u,i})\cdot con(si) \tag{21}$$

where $\hat{r}_{u,i}$ is the final prediction of $r_{u,i}$, and $con(si)$ de-

notes the confidence in the prediction given by si :

$$con(si)=\frac{sim(i,si)}{\sum_{j\in KI} sim(i,j)} \tag{22}$$

2.4 Make Prediction via HAPA

Though each of user-based and item-based HAPAs can make prediction, HAPA combines this two kinds of HAPAs to further improve the prediction accuracy.

Let $r_{u,i}$ be the unknown QoS value, pv_{user} and pv_{item} denote the predicted values given by user-based and item-based HAPA respectively. We use the following three indexes to evaluate the quality of these predicted values:

1) *Max Similarity (MS)*: For pv_{user} and pv_{item}, MS represents the maximum similarity between users in *KU* and services in *KI*, denoted by $ms(pv_{user})$ and $ms(pv_{item})$, respectively.
2) *Average Similarity (AS)*: For pv_{user} and pv_{item}, AS represents the average similarity between users in *KU* and services in *KI*, denoted by $as(pv_{user})$ and $as(pv_{item})$, respectively.
3) *Reciprocal of Standard Deviation (RSD)*: For pv_{user} and pv_{item}, RSD represents the reciprocal of the standard deviation of the similarities between users in *KU* and services in *KI*, denoted by $rsd(pv_{user})$ and $rsd(pv_{item})$, respectively.

These indexes can be written as:

$$ms(pv_{user})=\max\{sim(u,v)\mid v\in KU\} \tag{23}$$

$$ms(pv_{item})=\max\{sim(i,j)\mid j\in KI\} \tag{24}$$

$$as(pv_{user})=\frac{\sum_{v\in KU} sim(u,v)}{|KU|} \tag{25}$$

$$as(pv_{item})=\frac{\sum_{j\in KU} sim(i,j)}{|KI|} \tag{26}$$

$$rsd(pv_{user})=1\Big/\sqrt{\frac{\sum_{v\in KU}\left[sim(u,v)-as(pv_{user})\right]^2}{|KU|}} \tag{27}$$

$$rsd(pv_{item})=1\Big/\sqrt{\frac{\sum_{j\in KI}\left[sim(i,j)-as(pv_{item})\right]^2}{|KI|}} \tag{28}$$

If we decompose the prediction quality into each index, then we have:

$$Q_{ms}(pv_{user})=\frac{ms(pv_{user})}{ms(pv_{user})+ms(pv_{item})} \tag{29}$$

$$Q_{ms}(pv_{item})=\frac{ms(pv_{item})}{ms(pv_{user})+ms(pv_{item})} \tag{30}$$

$$Q_{as}(pv_{user})=\frac{as(pv_{user})}{as(pv_{user})+as(pv_{item})} \tag{31}$$

$$Q_{as}(pv_{item})=\frac{as(pv_{item})}{as(pv_{user})+as(pv_{item})} \tag{32}$$

$$Q_{rsd}(pv_{user})=\frac{rsd(pv_{user})}{rsd(pv_{user})+rsd(pv_{item})} \tag{33}$$

$$Q_{rsd}(pv_{item})=\frac{rsd(pv_{item})}{rsd(pv_{user})+rsd(pv_{item})} \tag{34}$$

The prediction quality of pv_{user} and pv_{item} can then be obtained as:

$$Q(pv_{user})=Q_{ms}(pv_{user})+Q_{as}(pv_{user})+Q_{rsd}(pv_{user}) \tag{35}$$

$$Q(pv_{item})=Q_{ms}(pv_{item})+Q_{as}(pv_{item})+Q_{rsd}(pv_{item}) \tag{36}$$

Hence, the final predicted value of $r_{u,i}$ is:

$$pv=\frac{pv_{user}\times Q(pv_{user})+pv_{item}\times Q(pv_{item})}{Q(pv_{user})+Q(pv_{item})} \tag{37}$$

2.5 Computational Complexity Analysis

We now discuss the computational complexity of predicting one unknown QoS value using our prediction algorithm. Suppose the dataset is an $m\times n$ matrix containing *m* service users and *n* Web services, and that each entry in this matrix is a QoS value for a user invoking a service.

For most CF-based methods, including HAPA, the similarity calculations must be performed prior to the prediction. The complexity of $sim(u,su)$ is $O(n)$, because there are at most *n* Web services invoked by both user *u* and *su*. Therefore the complexity of all similarities calculations for user *u* is $O(mn)$ since there are *m* users in the dataset, and for all user is $O(m^2n)$. Similarly, the complexity of all similarities calculations for all services is $O(mn^2)$. All similarities are generally calculated in advance since it is very time-consuming with a large dataset. Therefore similarities calculations are not included in the complexity of our prediction algorithm.

In the prediction of $r_{u,i}$, the complexity of determining *KU* is $O(f(m))$, as we have to sort all the similarities between user *u* and the other *m*-1 users to find the $user\_TopK$ most similar users. $f(m)$ is the complexity of this sorting process. A similar argument implies that the complexity of determining *KI* is $O(f(n))$. HAPA uses heap sorting, so $f(m)=m\cdot log_2 m$ and $f(n)=n\cdot log_2 n$.

In user-based HAPA, for every similar user, we must complete two steps: establish the user-based prediction equation, and then use linear regression to select the appropriate root. The coefficients of the two steps can be calculated simultaneously in the same loop. Therefore, the complexity for the prediction of one similar user is $O(n)$, because there are at most *n* Web services invoked by both active user and a similar user. This means the total complexity of user-based HAPA is $O(user\_TopK\cdot n)+O(m\cdot log_2 m)$, as we use the *user_topK* most similar users to form the final prediction.

Similarly, the computational complexity of item-based HAPA is $O(item\_TopK\cdot m)+O(n\cdot log_2 n)$.

HAPA is a linear combination of user-based and item-based HAPAs, and thus the total computational complexity is $O(user\_TopK\cdot n)+O(m\cdot log_2 m)+O(item\_TopK\cdot m)+O(n\cdot log_2 n)$

3 Experiments

In this section, we perform experiments to validate our HAPA, and compare the results with those from other CF

methods. Our experiments are intended to: 1) verify the rationality of our proposed characteristics and corollaries; 2) compare HAPA with other CF methods; and 3) parameterize HAPA to achieve optimum performance.

3.1 Experiments Setup

We adopted WSDream to validate our prediction algorithm. We implemented the experiments employing JDK 7.0 and eclipse 4.3 on Windows Server 2008 R2 Enterprises with Inter Xeon E5-2670 eight-core 2.60 GHz CPU and 32G RAM. Our experiments mainly consists of two parts: 1) Section 3.3 compares our algorithm with other 7 well known prediction methods based on the throughput as well as response time dataset; 2) Section 3.4 and 3.5 study the optimal parameters *user_topK* and *item_topK* of our algorithm.

To validate the accuracy of our algorithm, we predict only the known values, so that we can evaluate the error between the predicted values and real values.

3.2 Accuracy Metrics

The mean absolute error (MAE) and root mean squared error (RMSE) are frequently used to measure the difference between values predicted by a model or estimator and observed values. We adopted MAE and RMSE to measure the prediction accuracy of our algorithm through comparisons with other methods. MAE is defined as:

$$MAE = \frac{\sum \left| R_{ij} - \hat{R}_{ij} \right|}{N} \tag{38}$$

where R_{ij} denotes the QoS value of service j observed by user i, $\hat{R}_{ij}$ is the predicted QoS value, and N is the number of predicted values.

RMSE is defined as:

$$RMSE = \sqrt{\frac{\sum \left(R_{ij} - \hat{R}_{ij} \right)^2}{N}} \tag{39}$$

This is a good measure of accuracy when comparing prediction errors from different models for a particular variable [16].

3.3 Comparisons

We compare the predictive performance of our HAPA with other 7 well known prediction methods. The 7 compared methods are follows:

1) UMEAN (user mean): UMEAN employs the average QoS value of a user to predict the unknown QoS values.
2) IMEAN (item mean): IMEAN uses the average QoS value of the Web service from other users to predict the unknown QoS values.
3) UPCC (user-based collaborative filtering method using PCC): UPCC employs similar users for the QoS value prediction [9], [17].
4) IPCC (item-based collaborative filtering method using PCC): IPCC employs similar Web services (items) for the QoS value prediction [18].
5) UIPCC: This method combines UPCC and IPCC for the QoS value prediction [4].
6) NMF (nonnegative matrix factorization): NMF was proposed by Lee and Seung [19], [20]. It is different from other matrix factorization methods, as it enforces the factorized matrices to be nonnegative.
7) PMF (probabilistic matrix factorization): PMF was proposed by Salakhutdinov and Minh [21], and employs a user-item matrix for recommendations. PMF is based on probabilistic matrix factorization.

Sparse matrixes are common in the real word, and accurate predictions for sparse matrixes are very important in many practical applications, such as recommender systems, social networking, and so on. In this experiment, we made the datasets sparser by removing entries, enabling us to compare HAPA with the other methods using various dataset densities. The experimental results are shown in Table 2.

From Table 2, we can see that HAPA is more accurate than all of the other methods for all the two datasets. As the matrix density increases from 5% to 50%, the MAE and RMSE values become smaller, because the similarities between different users or different services become steadier as the amount of data increases.

TABLE 2
ACCURACY COMPARISON

	Methods	Matrix Density=5%		Matrix Density=10%		Matrix Density=20%		Matrix Density=50%	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Response Time	UMEAN	0.8844	1.8323	0.8721	1.8739	0.8676	1.8544	0.8324	1.8412
	IMEAN	0.7110	1.5799	0.6898	1.5378	0.6714	1.5231	0.6536	1.5098
	UPCC	0.6211	1.4012	0.5520	1.3148	0.4875	1.2353	0.3115	1.0754
	IPCC	0.6889	1.4288	0.5917	1.3245	0.4461	1.2094	0.2899	1.1732
	UIPCC	0.6246	1.4067	0.5367	1.3045	0.4498	1.2036	0.2112	1.0988
	NMF	0.6179	1.5798	0.6013	1.5487	0.5994	1.5238	0.4877	1.4856
	PMF	0.5690	1.4794	0.4987	1.2843	0.4495	1.1862	0.4013	1.0823
	HAPA	**0.3825**	**0.9476**	**0.3076**	**0.7602**	**0.2276**	**0.5628**	**0.1237**	**0.3156**
Throughput	UMEAN	55.1105	111.1489	54.4325	111.1732	53.1626	110.1478	52.9224	109.9912
	IMEAN	27.1347	68.2233	27.6745	65.2373	26.0415	64.1352	26.8736	63.4048
	UPCC	27.0011	62.3245	22.3521	52.3367	17.2317	49.3235	17.0515	46.9474
	IPCC	30.2561	66.3471	28.2158	62.0015	25.0561	56.1084	25.1887	54.4332
	UIPCC	26.3578	61.1231	21.5121	55.1045	17.2998	47.3046	15.9782	46.8578
	NMF	27.5612	66.2975	17.2365	52.2187	15.0994	51.0288	14.9870	47.8876
	PMF	20.4132	55.3479	16.3214	47.2143	14.1478	42.1162	14.9013	41.8863
	HAPA	**18.0112**	**52.2345**	**13.2578**	**42.7602**	**10.1276**	**36.5748**	**10.0037**	**35.4379**

3.4 Impact of user_topK

3.4.1 Is the Bigger user_topK the Better?

We employed user-based HAPA, i.e., only similar users, to predict 1000 known values of the dataset with different values of *user_topK*. Fig. 3 shows the resulting prediction accuracies. Note that smaller values of MAE and RMSE are better. Fig. 3 indicates that *user_topK* = 10, 50, 100, 150, 200, 250, 300, 330 gives relatively accurate predictions. However, when *user_topK* exceeds 150, the prediction accuracy declines quickly. We believe that this is because, when *user_topK* becomes too large, users with very low similarity are employed to make predictions. To avoid this problem, a good *user_topK* should exclude small similarities. As stated in Characteristic 2, if active user *u* and the similar user *v* are not very similar, then their similarity may fluctuate noticeably when they invoke more services, and the prediction equation no longer be valid. Hence, when *user_topK* = 330 (in a dataset containing 339 users), the high number of low similarity between users causes large prediction errors (see Fig. 3).

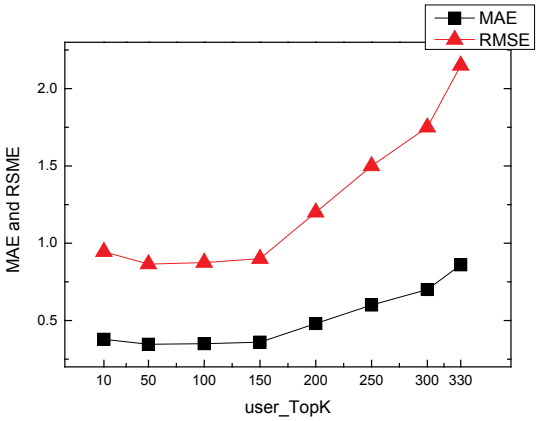


Fig. 3. The user_topK isn't the bigger the better, since a big user_topK may employ the users with low similarity to make predictions.

This experiment suggests us that *user_TopK* is not the bigger the better. To find the optimum *user_topK*, we designed the following experiment.

3.4.2 The Optimum Setting of user_topK

To study the optimum value of *user_topK*, i.e., the number of similar users needed to give a relatively accurate prediction, we studied one active user at a time.

We have speculated that the optimum *user_topK* is related to the similarity distribution. Therefore, we first studied the distribution of similarities for active users before making predictions, and observed the similarity ranges of the *user_topK* most similar users. Table 3 shows the results for four users selected from the dataset. User ID in Table 3 denotes the row number of the dataset, i.e., a user can be recognized as a row of the matrix. For each of the four users, we calculated how many similar users exist in different similarity ranges.

TABLE 3
SIMILARITY DISTRIBUTIONS

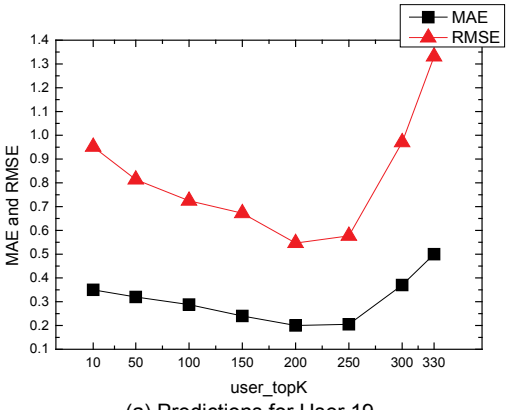
	Ranges of Similarities of Similar Users					
User ID	0.9-1	0.8-0.9	0.7-0.8	0.6-0.7	0.5-0.6	<0.5
19	0	0	47	151	69	71
63	1	0	3	140	133	61
140	0	4	11	6	16	301
312	0	0	63	146	53	76

Based on these similarity distributions, we examined which similarity ranges provided the most accurate predictions.

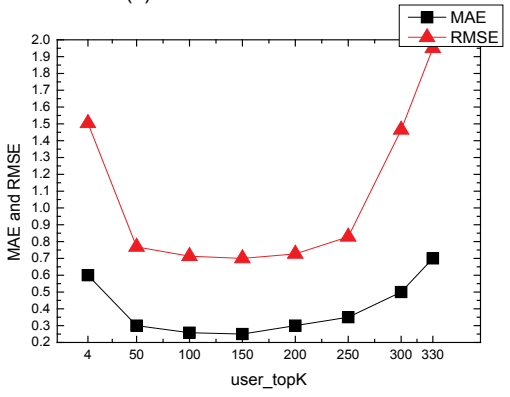
For each user of Table 3, we predicted 500 values shown as Fig. 4. Combining Table 3 and Fig. 4, we obtain a clear picture of the similarity ranges that contribute to accurate predictions.

From Fig. 4a, the most accurate prediction for user 19 comes from *user_topK* = 200. User 19 has 198 similar users whose similarity is greater than 0.6. Therefore, it is these users who contribute to the prediction accuracy.

Fig. 4b shows that very few, highly similar users can give a relatively accurate prediction. When *user_topK* = 4, we obtained a prediction accuracy for user 63 with MAE = 0.6 and RMSE = 1.5. As *user_topK* increased from 4 to 150, most similar users appear in the similarity range [0.6, 0.7], and these give the best prediction accuracy. Increasing *user_topK* to 200 introduced similar users in the similarity range [0.5, 0.6], and these harmed the prediction accuracy.



(a) Predictions for User 19



(b) Predictions for User 63

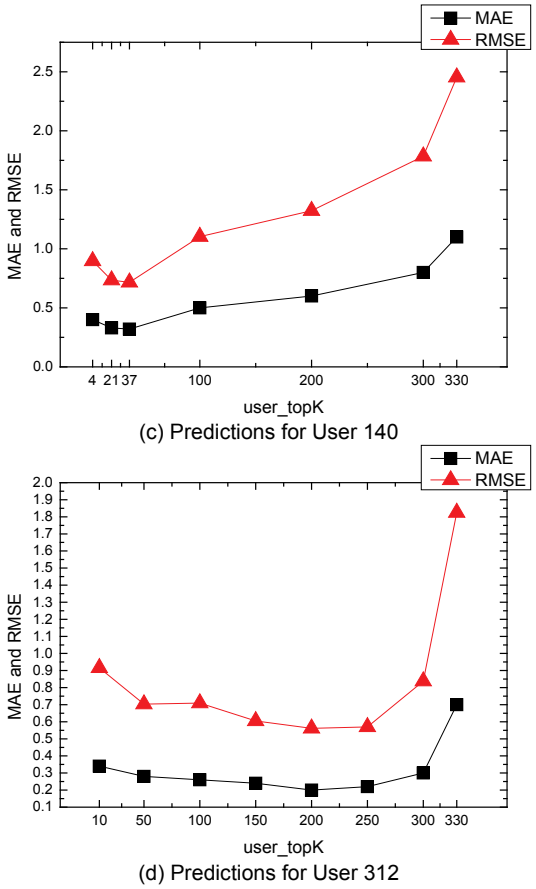


Fig. 4. Impact of user_topK for Different Active Users. The best user_topK employs PCC similarity greater than 0.6 which contributes to accurate prediction.

User 140 has only 21 similar users whose similarity is greater than 0.6. The additional 16 users in the similarity range [0.5, 0.6] improved the prediction accuracy slightly(see Fig. 4c), suggesting that users with a similarity value above 0.5 have an uncertain effect on prediction accuracy, as they caused the prediction accuracy of users 4 and 63 to deteriorate.

Similarly, user 312 obtained good prediction accuracy from *user_topK* = 200 with almost all the similar users whose similarity is greater than 0.6(see Fig. 4d).

Fig. 4 tells us that a similarity greater than 0.6 contributes to accurate prediction, and that similarities around 0.5 are unreliable.

Thus, this experiment suggests that the optimum value of *user_topK* is determined by the similarity distribution of similar users. For different active users, the best *user_topKs* are different. High performance can be achieved by setting *user_topK* to introduce all similar users whose similarity value is above 0.6.

3.5 Impact of item_topK

The mathematical principle of user-based HAPA is identical to that of item-based HAPA, as both users and services are processed as arrays of the two-dimensional QoS dataset. However, most services in the dataset have many very similar services (similarity greater than 0.9), unlike users, for whom similarity greater than 0.8 are rare. Hence, we studied how high similarity scores affect the *item_topK* setting. When using item-based HAPA to make predictions, we find it interesting that relatively few (about 10) highly similar services (similarity greater than 0.9) can give an accurate prediction.

As shown in Table 4, we selected four services that have a number of highly similar services. Service ID denotes the column occupied by that service in the dataset. We predicted 100 known QoS values for each of the four services under different *item_topK* values. The prediction accuracies are shown in Table 5. From Table 5, we can see that the prediction accuracy barely changed as *item_topK* increased. These results conform to Characteristic 2, as two very similar services have a steady similarity, and therefore, after predicting a QoS value for a service, the predicted value should not affect this similarity. Thus, we do not need many highly similar services to make accurate predictions.

TABLE 4
SIMILARITY DISTRIBUTIONS

	Ranges of Similarities of Similar Items					
Service ID	0.9-1	0.8-0.9	0.7-0.8	0.6-0.7	0.5-0.6	0-0.5
671	10	9	206	876	992	3732
736	23	307	812	675	540	3468
5277	127	661	712	595	419	3311
5096	90	653	907	580	395	3200

This is in contrast to user-based HAPA, where we generally required a large value of *user_topK*. To illustrate this, if we let $ANU[a, b)$ denote the average number of similar users whose similarity is in the interval $[a, b)$ for each user, and let $ANS[a, b)$ denote the average number of similar services whose similarity is in the interval $[a, b)$ for each service, then $ANU[0.9,1)=0.41$, $ANS[0.9,1)=41.35$, and $ANU[0.8,0.9)=4.07$, $ANS[0.8,0.9)=104.89$.

TABLE 5
ACCURACIES UNDER DIFFERENT PARAMETERS

	item_topK=5		item_topK=10		item_topK=20		item_topK=200		item_topK=1000	
Service ID	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
671	0.4303	1.087	0.2013	0.5062	0.1916	0.4863	0.1838	0.457	0.1802	0.4448
736	0.4687	1.1485	0.2103	0.5225	0.2033	0.5168	0.1974	0.4956	0.194	0.4798
5277	0.4796	1.184053	0.2806	0.6978	0.2803	0.7042	0.2718	0.6841	0.2714	0.6830
5096	0.4201	1.0525	0.2673	0.6550	0.2614	0.6516	0.2517	0.6230	0.2421	0.5992

3.6 Combining User-based and Item-based HAPAs Can Make Prediction More Accurate

HAPA use the combination of user-based and item-based HAPAs to make prediction more accurate. Based on the experiments of sections 3.4 and 3.5, we have determined good values for *user_topK* and *item_topK*. These parameters values are dynamic with the number of the similarities greater than 0.6. For example, if we need to predict $r_{u,i}$, *user_topK* should be set as follows:

1) For user u, if there exist more than 10 users with similarities greater than 0.9, then we can directly set *user_topK* = 10. Of course, we can set a larger value, but this will have a small effect on the prediction accuracy.
2) If there are not enough similar users with similarities greater than 0.9, we set the value of *user_topK* to the number of users whose similarity is at least 0.6.

The setting of item_topK is similar. Using the user-based HAPA, item-based HAPA and their combination, we predicted 10000 QoS values. A comparison of the prediction accuracy is shown in Fig. 5.

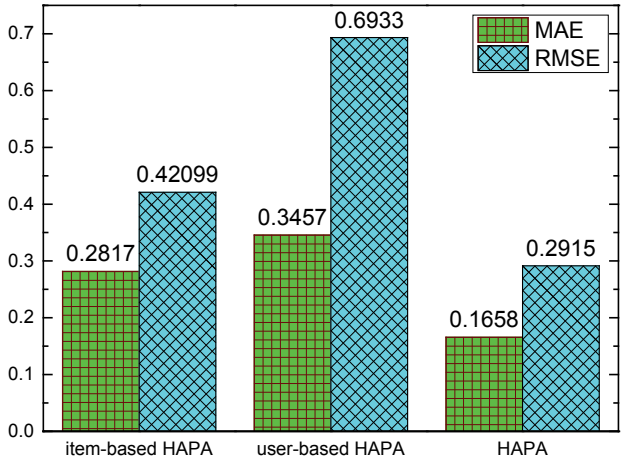


Fig. 5. Accuracy comparison of three HAPAs. HAPA use the combination of user-based and item-based HAPAs to make the prediction more accurate.

From Fig. 5, the HAPA clearly gives the most accurate predictions. HAPA employed user-based and item-based HAPAs to obtain two prediction values, and then, using the three indexes MS, AS, and RSD proposed in Section 2.4, evaluated the quality of these predicted values to form the final prediction. Therefore, HAPA gives more accurate predictions than either user-based or item-based HAPA.

4 Related Work

Web services QoS has been widely studied by many researches [22], [23], [24], [25], [26]. The previous QoS-based approaches such as service composition [27], [28], [29], [30] and service selection [31], [32][33], [34], etc. have limitations in some respect. Most of them assume that we can get the service QoS values from third-party organizations or service providers easily, but it always difficult to do so since the values we get may be not reliable and the more troublesome—the values we need may be unknown. Therefore, many researchers have studied how to predict the unknown QoS values. Especially for service recommender system, unknown QoS values prediction is more significant since the prediction's quality determines the recommendation's quality. In this paper, we focus on how to accurately predict the unknown client-side QoS values for service users.

We have witnessed the wide adoption of CF methods in the commercial recommender systems [10], [11], [35], [36]. There are two types of collaborative filtering approaches—the model-based and the neighborhood-based. The neighborhood-based collaborative filtering approach was mainly implemented by user-based methods [17], [37], item-based methods [18], [38], and their hybridization [39]. In user-based method, the unknown values are predicted by employing the values of similar users. In item-based methods, the unknown values are predicted by employing the values of similar items. Similarity calculation between items or users is an important part of neighborhood-based collaborative filtering. Multiple mechanisms such as Pearson Correlation and Vector Cosine based similarity are used for this.

Model-based CF employs training data to find the user interest pattern and then predict user's interest in haven't accessed items [48]. There are many methods including Bayesian networks, clustering models [40], latent semantic models [41], probabilistic latent semantic analysis [42], multiple multiplicative factor, latent dirichlet allocation and markov decision process used for finding user interest pattern [43]. This kind of CF handles sparse data better than memory based ones. It is good at processing large scale data set and improves the prediction performance. It can presents intuitive rationales for recommendations. But model building in model-based CF is very time-consuming.

CF methods originated from the processing of subjective data such as movie scores and satisfaction rating etc. In subjective data processing, there is a potential premise—similar user have similar interests in same items, but QoS values of Web services are objective and this premise no longer work for QoS prediction.

In the research field of unknown QoS values prediction, CF methods are widely borrowed from commercial recommender systems. But most of this kinds of QoS prediction methods are not aware that traditional CF doesn't work well for objective data. Zheng et al. [4] improved the computation method of PPC similarity. Little amount of data may cause an unreliable similarity. For example if two service user both invoked only one same service then their PPC similarity is 1 while it is useless for prediction. Therefore this work proposed a reliability factor of similarity based on the data amount and always employs the reliable similarities to make prediction. But this work's prediction method still similar to traditional CF, i.e., equation (1) and (2) which are applicable only for subjective data. Jiang et al. [5] was also committed to finding the appropriate similarities. This work thought different services contribute differently to the similarity between two users. If a service provided similar QoS for all service us-

ers including user *u* and *v*, then this service contribute little to the similarity between *u* and *v*. Conversely, if a service provided very different QoS for different service users but similar QoS for user *u* and *v*, then this service contribute a lot to the similarity between user *u* and *v*. However this work still use equations similar to (1) and (2) to make prediction, which are not applicable to objective data—Web service QoS. Shao et al. [9] did not find similarity as other researchers who only find the similarity greater than zero. He thought no matter a PPC similarity is positive or negative but only if its absolute value is big enough then this similarity can ensure a strong linear relationship, and then this similarity can be used for making prediction. But this work still use equations (1) and (2) to make prediction.

The main hindrance to verifying prediction methods is the lack of real-world Web service QoS datasets for experimental studies. It is difficult to mine the peculiarities of Web service QoS values, and the prediction accuracy of previous algorithms cannot be trusted without believable and sufficient real-world Web service QoS data. Experiments with a user-based PCC method reported by Shao et al. [9] for Web service QoS value prediction contain only 20 Web services. The Web service QoS dataset released by Al-Masri and Mahmoud [44] consists of 2,507 Web services evaluated by one user, but the usability of this dataset is limited because the QoS values of different users tend to vary widely. Therefore, we have utilized the dataset released by Zheng et al. [3], which contains information from 339 users for 5,825 Web services.

5 Conclusion and Future Work

We have utilized CF to predict unknown QoS values. Strictly speaking, our approach is essentially different from traditional CF which is not applicable to objective data prediction.

Traditional CF is based on the premise that similar users have similar subjective experience on the same items, but this premise no longer applies for the objective data. By the way of experiments, we determined some important characteristics of PCC similarity first. These characteristics show that the magnitude of PCC similarity determines its stability. Based on these characteristics, we proposed our HAPA. The prediction accuracy of HAPA was shown to outperform that of many of existing QoS prediction methods.

As the definition of *Objective Data*, Web service QoS is determined as a result of some objective factors, such as network traffic, bandwidth, when and where a user accessed a Web service. Our proposed HAPA does not predict unknown QoS values by these objective factors, but directly by the known QoS values. We can make predictions even more accurately if we know the relationship between these objective factors and the final QoS. To work out this relationship, we still have some important problems to solve, such as finding the core objective factors, how observe these objective factors, how probe user context and how learn this relationship. We are trying to solve these problems and will propose our approaches in the future work.

Acknowledgments

The work was supported by the NSFC (61202435); Beijing Municipal Natural Science Foundation (4132048); and the Fundamental Research Funds for the Central Universities (2014ZD01). Shangguang Wang is the corresponding author.

References

[1] L.-J. Zhang, J. Zhang, and H. Cai, *Services Computing*. Springer and Tsinghua Univ, 2007.

[2] S. Ran, "A model for Web services discovery with QoS," *ACM Sigecom exchanges*, vol. 4, no. 1, pp. 1-10, 2003.

[3] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware Web service recommendation by collaborative filtering," *IEEE Trans, Services Computing*, vol. 4, pp. 140-152, 2011.

[4] X. Chen, Z. Zheng, X. Liu, Z. Huang, and H. Sun, "Personalized qos-aware web service recommendation and visualization," Services Computing, *IEEE Trans, Services Computing*, vol. 6, pp. 35-47, 2013.

[5] Y. Jiang, J. Liu, M. Tang, and X. Liu, "An effective Web service recommendation method based on personalized collaborative filtering," *Proc. Ninth Int'l Conf. Web Services (ICWS '11)*, pp. 211-218, 2011.

[6] M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for qos-based service recommendation," *Proc. 10th Int'l Conf. Web Services (ICWS '12)*, pp. 202-209, 2012.

[7] J. Wu, L. Chen, Y. Feng, Z. Zheng, M. C. Zhou, and Z. Wu, "Predicting quality of service for selection by neighborhood-based collaborative filtering," *IEEE Trans, Systems, Man, and Cybernetics: Systems*, vol. 43, pp. 428-439, 2013.

[8] X. Chen, X. Liu, Z. Huang, and H. Sun, "Regionknn: A scalable hybrid collaborative filtering algorithm for personalized Web service recommendation," *Proc. Eighth Int'l Conf. Web Services (ICWS '10)*, pp. 9-16, 2010.

[9] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized qos prediction for Web services via collaborative filtering," *Proc. Fifth Int'l Conf. Web Services (ICWS '07)*, pp. 439-446, 2007.

[10] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, pp. 61-70, 1992.

[11] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: an open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, pp. 175-186, 1994.

[12] J. Bennett and S. Lanning, "The netflix prize," in *Proceedings of KDD Cup and Workshop*, pp. 35, 2007.

[13] J. L. Herlocker, J. A. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pp. 241-250, 2000.

[14] K. Emamy and R. Cameron, "Citeulike: A Researcher's Social Bookmarking Service," *Ariadne, No. 51*, 2007.

[15] S. A. Golder and B. A. Huberman, "Usage patterns of collaborative tagging systems," *Journal of Information Science*, vol. 32, pp. 198-208, 2006.

[16] R. J. Hyndman and A. B. Koehler, "Another look at measures of

forecast accuracy," *International Journal of Forecasting,* vol. 22, pp. 679-688, 2006.

[17] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," *Proc. 14th Int'l Conf. Uncertainty in Artificial Intelligence (UAI'98)*, pp. 43-52, 1998.

[18] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," *Proc. 10th Int'l Conf. World Wide Web (WWW '01)*, pp. 285-295, 2001.

[19] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature,* vol. 401, pp. 788-791, 1999.

[20] D. Seung and L. Lee, "Algorithms for non-negative matrix factorization," *Advances in Neural Information Processing Systems,* vol. 13, pp. 556-562, 2001.

[21] A. Mnih and R. Salakhutdinov, "Probabilistic matrix factorization," in *Advances in Neural Information Processing Systems*, pp. 1257-1264, 2007.

[22] M. C. Jaeger, G. Rojec-Goldmann, and G. Muhl, "Qos aggregation for Web service composition using workflow patterns," *Proc. Eighth Int'l Conf. Enterprise Distributed Object Computing (EDOC '04)*, pp. 149-159, 2004.

[23] D. A. Menascé, "QoS issues in Web services," *Internet Computing, IEEE,* vol. 6, pp. 72-75, 2002.

[24] M. Ouzzani and A. Bouguettaya, "Efficient access to Web services," *Internet Computing, IEEE,* vol. 8, pp. 34-44, 2004.

[25] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of Real-World Web Services," *IEEE Trans, Services Computing*, vol. 7, no. 1, pp. 32-38, 2014.

[26] S. Rosario, A. Benveniste, S. Haar, and C. Jard, "Probabilistic qos and soft contracts for transaction-based Web services orchestrations," *IEEE Trans, Services Computing,* vol. 1, no. 4, pp. 187-200, 2008.

[27] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition," in *Proceedings of the 18th international conference on World wide Web*, pp. 881-890, 2009.

[28] D. Ardagna and B. Pernici, "Adaptive service composition in flexible processes," *IEEE Trans, Software Engineering,* vol. 33, pp. 369-384, 2007.

[29] V. Cardellini, E. Casalicchio, V. Grassi, F. Lo Presti, and R. Mirandola, "Qos-driven runtime adaptation of service oriented architectures," in *Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, pp. 131-140, 2009.

[30] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng, "Quality driven Web services composition," in *Proceedings of the 12th international conference on World Wide Web*, pp. 411-421, 2003.

[31] P. A. Bonatti and P. Festa, "On optimal service selection," in *Proceedings of the 14th international conference on World Wide Web*, pp. 530-538, 2005.

[32] V. Cardellini, E. Casalicchio, V. Grassi, and F. Lo Presti, "Flow-based service selection forWeb service composition supporting multiple qos classes," *Proc. Fifth Int'l Conf. Web Services (ICWS '07)*, pp. 743-750, 2007.

[33] L. Mei, W. K. Chan, and T. Tse, "An adaptive service selection approach to service composition," *Proc. Sixth Int'l Conf. Web Services (ICWS '07)*, pp. 70-77, 2008.

[34] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints," *ACM Transactions on the Web (TWEB),* vol. 1, p. 6, 2007.

[35] R. Burke, "Hybrid recommender systems: Survey and experiments," *User modeling and user-adapted interaction,* vol. 12, pp. 331-370, 2002.

[36] X. Su, T. M. Khoshgoftaar, X. Zhu, and R. Greiner, "Imputation-boosted collaborative filtering using machine learning classifiers," in *Proceedings of the 2008 ACM symposium on Applied computing*, pp. 949-950, 2008.

[37] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 230-237, 1999.

[38] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *Internet Computing, IEEE,* vol. 7, pp. 76-80, 2003.

[39] J. Wang, A. P. De Vries, and M. J. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 501-508, 2006.

[40] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, *et al.*, "Scalable collaborative filtering using cluster-based smoothing," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 114-121, 2005.

[41] T. Hofmann, "Latent semantic models for collaborative filtering," *ACM Transactions on Information Systems (TOIS),* vol. 22, pp. 89-115, 2004.

[42] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence,* vol. 2009, p. 4, 2009.

[43] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P. Kriegel, "Probabilistic memory-based collaborative filtering," *IEEE Trans, Knowledge and Data Engineering,* vol. 16, pp. 56-69, 2004.

[44] E. Al-Masri and Q. H. Mahmoud, "Investigating Web services on the world wide Web," *Proc. 17th Int'l Conf. World Wide Web (WWW '08)*, pp. 795-804, 2008.

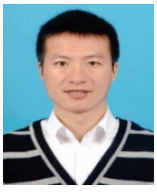
[45] L. Chen, Y. Feng, and J. Wu, "Collaborative QoS Prediction via Feedback-Based Trust Model," *Proc.6th Int'l Conf. Service-Oriented Computing and Applications (SOCA'13)*, pp. 206-213, 2013.

[46] 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.

[47] Q. Yu, "QoS-aware service selection via collaborative QoS evaluation," *World Wide Web*, vol. 17, pp. 33-57, 2014.

[48] Z. Zheng, H. Ma, and M. R. Lyu, "Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization," *IEEE Trans, Services Computing*, vol. 6, no. 3, pp. 289-299, 2013.

You Ma received the BEng and MEng degrees from Qufu Normal University and North China University of Warter Resource and Electric Power, in 2002 and 2006, respectively. He is currently working toward the PhD degree in the Beijing University of Posts and Telecommunications. His research interests are in service computing, and recommender systems.



Shangguang Wang is an assistant professor at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT). He received his Ph.D. degree at BUPT in 2011. His PhD thesis was awarded as outstanding doctoral dissertation by BUPT in 2012. He has served as editorial board member of International Journal of Services Computing.He has served as PC memberof IEEE SCC, IEEE BigData, IEEE MS, etc. His research interests include Service Computing, Cloud Services, and QoS Management.

Patrick C.K. Hung is an associate professor at University of Ontario Institute of Technology, Canada. He is a founding committee member of the IEEE International Conference of Web Services, IEEE International Conference on Services Computing, IEEE Congress on Services and IEEE Congress on BigData. He is Associate Editor of the IEEE Transactions on Services Computing, International Journal of Web Services Research and International Journal of Business Process and Integration Management, as well as an Executive Group Member and Co-ordinating Editor of the Information Systems Frontiers by Springer.

Ching-Hsien Hsuis a professor in department of computer science and information engineering at Chung Hua University, Taiwan. His research includes high performance computing, cloud computing, parallel and distributed systems, ubiquitous/pervasive computing and intelligence.He has been involved in more than 100 conferences and workshops as various chairs and more than 200 conferences/workshops as a program committee member. He is the editor-in-chief of international journal of Grid and High Performance Computing, and serving as editorial board for around 20 international journals.

Qibo Sunreceived his Ph.D. degree in communication and electronic system from the Beijing University ofPosts and Telecommunication in 2002. He iscurrently anassociate professor at the Beijing University ofPosts and Telecommunicationin China. He is a member of the China computer federation. His researchinterests include services computing, Internet of things, and network security.

Fangchun Yangreceived his Ph.D. degree in communication and electronic system from the Beijing University of Posts and Telecommunication in 1990. He is currently a professor at the Beijing University of Posts and Telecommunication, China. His research interests include network intelligenceand communications software.He is a fellow of the IET.