Accepted Manuscript

Your neighbors are misunderstood: On modeling accurate similarity driven by data range to collaborative web service QoS prediction

Zhen Chen, Limin Shen, Feng Li

PII: S0167-739X(18)31611-X
DOI: https://doi.org/10.1016/j.future.2019.01.003
Reference: FUTURE 4696

To appear in: *Future Generation Computer Systems*

Received date : 6 July 2018
Revised date : 12 November 2018
Accepted date : 1 January 2019

Please cite this article as: Z. Chen, L. Shen and F. Li, Your neighbors are misunderstood: On modeling accurate similarity driven by data range to collaborative web service QoS prediction, *Future Generation Computer Systems* (2019), https://doi.org/10.1016/j.future.2019.01.003

Your Neighbors Are Misunderstood: On Modeling Accurate Similarity Driven by Data Range to Collaborative Web Service QoS Prediction

Zhen Chen[1], Limin Shen[1,*], Feng Li[2]

[1]College of Information Science and Engineering, Yanshan University, China

[2]College of Computer and Communication Engineering, Northeastern University, China

Corresponding author Email: shenllmm@ysu.edu.cn

Abstract Quality of service (QoS) is a set of non-functional attributes of Web services for differentiating enriched Web services with same or similar functionality. Predicting the unknown QoS of Web services for service users is often required for any QoS based service computing because QoS plays a fundamental role in reliable Web service recommendation, composition and selection. Existing collaborative filtering based QoS prediction methods suffer from a serious acclimatization issue caused by the difference of QoS data range, which dramatically degrades the prediction accuracy and even impedes its adaptability. The fact that Web service QoS data exhibit large service effect with different data ranges, is verified on public real-world datasets. In this study, we aim to tackle the problem of QoS prediction while considering the influences of QoS data range in the context of collaborative filtering. In particular, a simple yet effective similarity model called JacMinMax, which is driven by QoS data range, is designed. Furthermore, two neighborhood selection strategies using JacMinMax are proposed, and the obtained neighbors are systemically integrated into neighborhood- and model-based methods for collaborative QoS predictions. Experimental results show that the proposed method efficiently alleviates the influence of the concerned QoS data ranges, and performs better than many state-of-the-art approaches with respect to accuracy.

Keywords Web service; quality of service; collaborative prediction; data range

1 Introduction

1.1 Background of Web service QoS prediction

A Web service is an Internetware that describes a standardized way to integrate Web-based applications using SOAP, WSDL, and UDDI open standards over an Internet protocol backbone [1]. Web services extend the World Wide Web infrastructure to provide the means for software to connect to other software applications. Applications access Web services via ubiquitous Web protocols and data formats such as XML and JSON, with no need to know anything about the platform, object model, or programming language used to implement the Web services. With the deepening and development of service oriented computing concept, Web services have become one of the most important technologies for building distributed systems and connect software, applications, and systems together to create attractive user experiences [2-3].

At present, traditional Internet has gradually changed from being “data centric” to “service centric”, and increasing number of enterprises and organizations have joined in the ranks of software servitization [4-5]. For instance, Amazon Web Services and Microsoft Azure provide services that span a wide range including computing, storage, networking, and analytics, which provide developers with agility to leverage existing Web services [6]. However, care must be taken that Web services deployed on the Internet are constantly enriched, thereby causing a severe overload of Web services with the same or similar function in the network. For example, as of May 27, 2018, the number of short-message verifying services on the Baidu API store has reached 105. The richness of Web services in the network not only provides users with a wide range of choices but also causes difficulties for users when selecting the optimal Web services. It is our contention that the most important consideration in the

future is not to produce a larger amount of resources of Web services, but to study how to recommend the most suitable Web services to the most suitable service users, thereby ensuring an accurate match between the Web services and service users.

Quality of service (QoS) is defined as a set of non-functional attributes of Web services, where each attribute with an attribute value characterizes the quality information of a Web service on a certain aspect, such as response time, throughput, and cost[7]. QoS directly expresses the operability and computational nature of the Web services, and it is able to effectively differentiate the performance of Web services with the same or similar functionality. Thus, QoS has become a decisive factor in the selection, combination, and deployment of Web services for constructing high-quality distributed application systems. QoS data-driven Web services recommendation has become a hot research topic in the field of service computing [8-10]. However, predicting the unknown QoS of Web services for service users is often required for any QoS based service computing studies because of the following facts:

(1) *Large number of deployed and small number of used Web services*. The number of Web services deployed on the Internet is large and growing gradually. However, a single target user has used only a small number of Web services, while most of the QoS information of Web services are still unknown to target users.

(2) *Time and resource costs hinder service users to test the QoS of Web services*. As mentioned, the number of Web services with similar functionality is large. Owing to the large time and resource costs, testing all the candidate Web services before selecting and invoking the Web service with a certain function is impractical for users. In addition, charging also hinders users to test the QoS of candidate Web services because some Web services are subject to fees before every invocation.

Based on the aforementioned facts, personalized QoS prediction has become the basis and key issue to be resolved in QoS based service computing, including Web service recommendation, composition, and selection. Over the past 10 years, extensive research has been conducted on how to accurately predict the QoS values of Web services to users and various methods have been proposed [11,12]. As one of the most successful applications in modern e-commerce, collaborative filtering (CF) has become an outstanding representative of the recommendation system over the last two decades. For example, Amazon.com has been using CF to recommend products to its customers [13], and Netflix gained improvements to the CF technology underlying its movie rental service [14]. Motivated by the success of CF in e-commerce, CF based QoS prediction methods have been widely studied in both academia and industry

Generally, CF is the technique of making automatic predictions (filtering) about the interests of a user by collecting preferences or QoS information from many users (collaborating) based on the assumption that similar users prefer similar services, and similar services are preferred by similar users [15]. Intuitively, CF finds a group of similar neighbors based on the calculated similarities, such as Pearson correlation coefficient, Jaccard and cosine similarities. The final prediction is made on the basis of the selected neighbors. Thus, the core of CF is to determine who can be cooperated with and how to cooperate with them.

To improve the QoS prediction accuracy of Web services, context information besides similarity, such as geographical, social, and trust information [16-18] have been introduced to existing methods on how to find more similar neighbors. Moreover, efforts of existing methods are either a combination of different similarities [19], or a combination of different prediction methods [20] on how to cooperate with similar neighbors. Although numerous efforts have been devoted to improve the CF based QoS prediction accuracy, the fundamental reason from the adaptability of the CF method itself caused by the different data ranges on accurate QoS prediction has been ignored in the existing methods, thereby causing a limited improvement in QoS prediction accuracy.

1.2 Motivation example

It is our contention that the CF method does not work effectively on datasets that have large data ranges between users or services. To clarify our motivation, we present a toy example analysis and study how the data range can impair the accuracy of the CF method.

Assuming that there are four users and four Web services, every user submits the observed response time of Web services. Thus, the four users contribute to a 4×4 user-service response time rating matrix $RT=\{rt_{u,s}\}_{4\times4}$, and each entry $rt_{u,s}$ records the response time of Web service s accessed by user u, as shown in Table 1.

Table 1 Toy example: 4×4 user-service response time matrix(unit: *second*)

	s_1	s_2	s_3	s_4
u_1	0.1000	1.0000	1.5000	0.2000
u_2	0.3000	3.0000	4.4000	0.4000
u_3	0.2000	2.0000	*null*	0.4000
u_4	*null*	3.0000	4.000	0.8000

Different from the rating scores in MovieLens [21], they are *subjective data* in which ratings are in the same range [1, 5]. QoS data are *objective data* that vary in different ranges; for instance, the data range of response time is (0, +∞) in reality. Moreover, our previous research suggests that the response time data of Web services exhibit large service effects [22], that is, some Web services tend to exhibit a higher response time than others because of greater server load, and vice versa. Thus, in practical application, it often happens that s_2 and s_3 vary with data range [1, 10], and s_1 and s_4 vary with data range (0, 1]. Note that $rt_{4,1}$ and $rt_{3,3}$ are missing values that have to be predicted. The real value of $rt_{4,1}$ is 0.3. Without loss of generality, $rt_{4,1}$, which is marked red in Table 1, is taken as an example and service based neighborhood CF is used as the prediction method for analysis in the following part.

To accurately predict the missing response time $rt_{4,1}$ of u_4 on s_1, we first need to calculate the similarities between s_1 and three other Web services, namely, s_2, s_3, and s_4, for neighborhood selection based on the idea of CF. Here, we adopt the widely used similarity calculation method: Jaccard [23], PCC [24], COS [25], and COSA [26]. Then, we obtain their similarity matrices based on Table 1, as shown in Table 2.

Table 2 Calculated Jaccard, PCC, COS, and COSA similarities between Web services

Jaccard

	s_1	s_2	s_3	s_4
s_1	1.0000	0.7500	0.5000	0.7500
s_2		1.0000	0.7500	1.0000
s_3			1.0000	0.7500
s_4				1.0000

PCC

	s_1	s_2	s_3	s_4
s_1	1.0000	1.0000	1.0000	0.8660
s_2		1.0000	0.9919	0.7609
s_3			1.0000	0.6665
s_4				1.0000

COS

	s_1	s_2	s_3	s_4
s_1	1.0000	1.0000	1.0000	0.9800
s_2		1.0000	0.9988	0.9592
s_3			1.0000	0.9358
s_4				1.0000

COSA

	s_1	s_2	s_3	s_4
s_1	1.0000	-0.8829	-1.0000	0.9960
s_2		1.0000	0.9905	-0.7511
s_3			1.0000	-0.9538
s_4				1.0000

As shown in Table 1, services s_1 has high similarity with three other Web services, especially $PCC(s_1, s_2) = 1$, $PCC(s_1, s_3) = 1$, and $PCC(s_1, s_4) = 0.866$, indicating a more similar performance of the two Web services co-invoked by the same users. Then, as services $\{s_2, s_3, s_4\}$ have been accessed by u_4, and they are highly correlated with s_1, we can naturally select $\{s_2, s_3, s_4\}$ as the similar neighbors of service s_1 for the final prediction of $rt_{4,1}$. Finally, we make the collaborative prediction using the weighted average with mean offset method based on the selected neighbors. The calculation of $rt_{4,1}$ by using PCC similarity can be derived as follows:

$$rt_{4,1} = \overline{r_1} + \frac{PCC(s_1,s_2)(rt_{4,2}-\overline{r_2}) + PCC(s_1,s_3)(rt_{4,3}-\overline{r_3}) + PCC(s_1,s_4)(rt_{4,4}-\overline{r_4})}{PCC(s_1,s_2) + PCC(s_1,s_3) + PCC(s_1,s_4)}$$
$$= 0.2 + \frac{(3-2.25)\times 1 + (4-3.3)\times 1 + (0.8-0.45)\times 0.8660}{1+1+0.8660} \quad (1)$$
$$= 0.8117$$

Meanwhile, we calculate the final predicted values of $rt_{4,1}$ using Jaccard, COS, and COSA similarities, and calculate the errors between the predicted values and real value. The results are reported in Table 3.

Table 3 Predicted response time of $rt_{4,1}$

	Jaccard	PCC	COS	COSA
Predicted $r_{4,1}$	0.7875	0.8117	0.8017	1.3428
Prediction error	0.4875	0.5117	0.5017	1.0428

Table 3 shows larger errors between the predicted values and the real value 0.3. Why are there such large errors? Existing methods attribute the selected neighbors are not similar as the main reason, and they further introduce context information to filter the similar neighbors [16-18]. We argue that these similar neighbors are misunderstood, and we try to explain it as *a phenomenon of acclimatization in the CF method* caused by the differences in data ranges. Specifically, when we use Eq. (1) to perform collaborative prediction, services s_2, s_3, and s_4 are indeed highly correlated with s_1. However, the data ranges of s_2 and s_3 are both in [1, 10], whereas the data range of s_1 is only in (0, 1], which makes the contributions of neighbor s_2 and s_3 at a very different level with that of s_1 in Eq. (1), thereby leading to the large errors between the predicted values and real value.

It is our contention that an error of this type indicates that accurate prediction of the QoS value, based on similar neighborhood, must be calculated at a level where all the neighbors have the same level contributions as the target service. Therefore, a deeper analysis of the relationship between the target service and its neighbors must be performed prior to making predictions. The apparent cause is the problem of the selected neighbors, while the deeper reason is the problem of the adaptability of CF itself caused by differences in data ranges. The upshot is that services s_2 and s_3 in the toy example are misunderstood, and s_2 and s_3 are similar neighbors to s_1 because an underlying positively correlated relationship exists between them. However, the problem lies in the fact that the response time range of s_2 and s_3 is ten times that of s_1. Unlike the existing methods, which only focus on improving the quality of selected neighbors [16-20], our current study aims to design an accurate similarity calculation model to finely characterize the relationship of selected neighbors, which effectively alleviates the phenomenon of acclimatization in the CF method while improving the prediction accuracy.

1.3 Our contributions

In this study, we aim to address the problem of QoS prediction while considering the influences of the QoS data range in the CF method. We first discover the phenomenon of acclimatization in CF method and analyze why CF method does not work well on QoS data with large data ranges. Furthermore, to verify our hypothesis that Web services QoS data perform a large data range effect, we conduct empirical data analysis on real-world datasets. The results have demonstrated the fact that the QoS performance between Web services has large data change ranges, which challenge the adaptability of neighborhood based CF methods. Finally, to obtain more accurate Web service QoS prediction, unlike most neighborhood based CF systems that combine traditional similarities or prediction methods [19-20], our method presents a simple yet effective similarity model to select similar neighbors using two designed neighborhood filtering strategies, thereby alleviating the influence of QoS data range and improving the QoS prediction accuracy.

In a nutshell, the key contributions of our work are summarized are as follows:

(1) The fact that some Web services tend to show better performance than others to service users is verified on real-world datasets, demonstrating that Web service QoS data exhibit a large service effect with different data ranges.

(2) The phenomenon of acclimatization in CF method and the reason why neighborhoods are misunderstood have been discovered and analyzed in the context of QoS rating datasets with large data ranges among Web services.

(3) To alleviate the influence of the QoS data range, a simple yet effective similarity model driven by QoS data range is proposed. This similarity model is of importance as it can be useful not only for accurate Web service QoS prediction but also for those rating-oriented related studies that have a large data range influence, such as POI recommendation and social network analysis.

(4) To evaluate the advantages of our approach, detailed comparisons with many state-of-the-art counterparts and similarities are conducted. Some beneficial and interesting conclusions have also been found through a comprehensive analysis of the experimental results.

This paper is organized as follows. Related works are reviewed in Section 2. In Section 3, we describe the research problem and motivation based on real-world QoS data range analysis. In Section 4, we provide details of collaborative QoS prediction method considering the influence of QoS data range including similarity calculation, neighborhood selection, and collaborative prediction. We present an evaluation of our method using WSDream dataset in Section 5. Finally, we conclude the paper in Section 6.

2 Related work

2.1 Similarity in collaborative filtering

Similarity is an essential measurement of data mining, and a good metric often leads to a good performance [27]. In the CF based method, similarity plays a double role: to filter dissimilar neighbors and obtain similar neighbors for the target users or services, and to weight the importance of similar neighbors for collaborative prediction. Therefore, one of the most important design decisions in CF is the choice of similarity computation method. The following similarity methods have been proposed and evaluated in the literature:

Jaccard (Jaccard). Jaccard similarity is defined as the size of the intersection divided by the size of the union of the two sets [23]. The Jaccard similarity between Web services s and t can be calculated by:

$$Jaccard(s,t) = \frac{|U_s \cap U_t|}{|U_s \cup U_t|} = \frac{|U_s \cap U_t|}{|U_s| + |U_t| - |U_s \cap U_t|} \tag{2}$$

where U_s and U_t are the sets of users who have invoked Web services s and t, respectively. $|.|$ denotes the size of a given set. Jaccard similarity measure uses information based on the distribution and the number of ratings obtained by each pair of services to be compared. Its use has been a determining factor in achieving the quality of the results obtained, which confirms that combining this information with traditional information based on numerical values of the QoS ratings is appropriate when we intend to design a cold-start similarity measure. In addition, we observe that the proportion of common ratings represents the credibility of the similarity between two services. Therefore, we use Jaccard similarity as the basis of our proposed similarity in Section 4.2.

Pearson correlation coefficient (PCC). PCC similarity is defined as the covariance of the two variables divided by the product of their standard deviations [24]. The PCC similarity between Web services s and t can be calculated by:

$$PCC(s,t) = \frac{\sum_{u \in U_{st}} (r_{u,s} - \overline{r}_s)(r_{u,t} - \overline{r}_t)}{\sqrt{\sum_{u \in U_{st}} (r_{u,s} - \overline{r}_s)^2} \sqrt{\sum_{u \in U_{st}} (r_{u,t} - \overline{r}_t)^2}} \tag{3}$$

where $r_{u,s}$ represents the QoS value that is observed by user u on the service s; $U_{st} = U_s \cap U_t$ is the subset of users who have invoked both services s and t previously; and $\overline{r}_s$ and $\overline{r}_t$ denote the average QoS values of service s and t observed by different service users, respectively. $PCC(s, t)$ has a value between +1 and −1, where 1 is total positive linear correlation, 0 is no linear correlation, and −1 is total negative linear correlation.

Although PCC can provide accurate similarity computation, it suffers from co-accessed user problem. Specifically, the effect of the number of co-accessed user on the PCC similarity is not considered, which will overestimate the similarity of two Web services that are actually not similar but happen to perform similar QoS observed by a few co-accessed users. Moreover, PCC outputs high similarity even if a significant difference exists in the ratings, that is, the rating range is ignored. For example, in Table 1, s_1={0.1, 0.3} and s_3={1.5, 4.4}, while *PCC* (s_1, s_3) =1. Our data analysis in Eq. (1) shows that the rating range will dramatically impair the prediction accuracy. Therefore, designing a more accurate similarity method to resolve the aforementioned problems is necessary.

Cosine (COS). COS similarity is defined as an inner product between two non-zero vectors that measures the cosine of the angle between them [25]. The COS similarity between Web service s and t can be calculated as follows:

$$COS(s,t)=\frac{\sum_{u\in U_{st}} r_{u,s}r_{u,t}}{\sqrt{\sum_{u\in U_{st}} r_{u,s}^{2}}\sqrt{\sum_{u\in U_{st}} r_{u,t}^{2}}} \tag{4}$$

Note that COS similarity also suffers the co-accessed user problem that exists in PCC. Furthermore, COS similarity is more about differentiating the difference from direction and is insensitive to the range of rating value. Therefore, it is impossible to measure the difference between the values of each dimension, which leads to a situation where *COS* (s_1, s_3) =0.9988 (Table 1), indicating a very similar relationship between services s_1 and s_3. However, judging from the rating values, we do not observe a similar response time between s_1 and s_3. Thus, in this study, we try to amend this irrationality by incorporating the influence of QoS data range in similarity calculation.

Adjusted cosine (COSA). COSA similarity is a modified form of COS similarity where we consider that different users have different rating schemes [26]. The COSA similarity between Web services s and t can be calculated as follows:

$$COSA(s,t)=\frac{\sum_{u\in U_{st}} (r_{u,s}-\bar{r}_u)(r_{u,t}-\bar{r}_u)}{\sqrt{\sum_{u\in U_{st}} (r_{u,s}-\bar{r}_u)^2}\sqrt{\sum_{u\in U_{st}} (r_{u,t}-\bar{r}_u)^2}} \tag{5}$$

COSA aims to remove the drawback from the COS similarity, by subtracting the average rating for each user from each user's rating for the pair of Web services. Although COSA can alleviate the disadvantage of COS similarity and ensure accurate similarity, our experimental results show that large prediction errors also exists, demonstrating that the prediction accuracy cannot be improved significantly.

2.2 Neighborhood based Web service QoS prediction

According to the classification of traditional CF methods, Web service QoS prediction methods can be categorized into two general classes: neighborhood based and model based. In neighborhood based methods, the QoS values of similar neighbors are directly used to predict the QoS values for new Web services. This task can be performed in two ways known as user-based and service-based prediction. Shao et al. [28] first introduced the CF technology into Web services QoS prediction in 2007. This method was used in collaborative prediction by finding similar neighbors of users. In addition, to improve the accuracy of the prediction results, the researchers calculated the positive and negative similarity separately when determining the predicted value of QoS. Zhang et al. [29] improved the similarity calculation of two services by combing the similarity with the common users and COSA similarity. Zheng et al. [20] proposed a combined user-based CF and service-based CF based on an improved PCC similarity, which employs a significant weight to reduce the influence of a small number of similar neighbors. E et al. [30] also proposed a hybrid QoS prediction method based on weighted Adamic-Adar, in which location context was incorporated to optimize the neighbor searching.

To improve Web service QoS prediction accuracy, existing neighborhood-based methods have been improved mainly from the following two aspects: (1) Combination of different similarities. Researchers have proposed PCC, COS, and COSA similarities to accurately measure the similarity between users or services. To further enhance the similarity calculation, the number of co-invoked services and the location information are introduced to the traditional similarity calculation. (2) Combination of different prediction results. As user- and service-based methods may achieve different prediction accuracies, many existing methods have attempted to introduce different weights to obtain a weighted result of different CF methods. In summary, the improvements of existing methods are focused on combination and they do not consider the situation sensitivity of QoS data; in other words, some services might exhibit a high response time in general, whereas others might have lower response time. Our empirical analysis shows that CF method suffers the acclimatization issue and does not work well in this scenario. Ning et al. [31] pointed out that considering the rating range has a significant benefit on improving CF accuracy, and they proposed mean-centering and z-score normalization schemes to alleviate the influence of rating range. As normalization makes the experimental dataset become normalized data, the predicted results are also normalized results. However, these normalized results are unsuitable for the application of Web service QoS prediction because Web service real QoS values are not only used for Web service ranking but also for quantitative assessment of the reliability of systems that invoked the Web services. Chen et al. [32] mapped QoS data in different ranges into a uniform interval using Gaussian normalization theory to improve prediction accuracy; they designed a restoration algorithm to restore the real value, and the final result was generated by combining the results of user- and service- neighborhood based CF methods. Different from previous works, our study improves the similarity calculation by capturing the influence of QoS data range between Web services, which can accurately characterize their similarity relationship. Moreover, besides the top-K neighbor selection strategy, a similarity threshold filtering strategy is designed to obtain more similar neighbors.

2.3 Model based Web service QoS prediction

In contrast to neighborhood-based Web services QoS prediction methods, which leverage the stored QoS values directly for the prediction, model based approaches use these known QoS values to learn a predictive model. The latent features of users and services are learned from training data based on matrix factorization (MF) technique and later used to predict unknown QoS values. Lo et al. [33] proposed an extensible matrix factorization model called EMF, which employed user and service PCC similarities for user and service neighbor selection, respectively; these neighbors are integrated into the MF model using two designed regularization terms. Similar to EMF with PCC for neighbor selection, NIMF was proposed by Zheng et al. [34] to integrate user similar neighbors directly into MF to improve the accuracy of user feature learning. To further improve the prediction accuracy, additional context information is introduced to the MF. Xu et al. [35] introduced reputation information to address the inaccuracy caused by unreliable users. Chen et al. [36] proposed GeoMF, which incorporates the knowledge of geographical neighbors into MF; their experimental findings demonstrated the effectiveness of geographical information in improving QoS prediction accuracy.

Generally, model based Web services QoS prediction methods leverage all the existing ratings to learn a prediction model, whereas neighborhood based methods only use the similar neighbors' ratings for prediction. By contrast, model based methods obtain better prediction results than neighborhood-based methods. However, the underlying reason for the low accuracy of neighborhood based methods has not been identified. Thus, most of the studies focus on model based methods. Note that the results obtained using the model-based methods are difficult to explain, thereby affecting user acceptance of the recommendation results. In this paper, we introduce a QoS data range based method to accurately estimate the similarity of services. The experimental results show that our proposed neighborhood based method obtains better prediction accuracy than the model based methods with respect to RMSE in response time dataset and yield intuitive explanations for the prediction results of Web service recommendation.

3 Problem description and motivation

3.1 Problem description

In practical application, when service user u has accessed a Web service s, user u submits the observed QoS information called *rating*, which is frequently represented as a triple (*User*, *Service*, *Rating*) to the UDDI system [37]. The set of all rating triples form a sparse matrix referred to as the rating matrix. Triple (*User*, *Service*, *null*) indicates that the user has not accessed the service, which are unknown value in this matrix. Table 1 shows an example response rating matrix for four users and four Web services in a Web service recommender system; cells marked "*null*" indicate unknown values (the user has not accessed that service). Let $U=\{u_1,u_2,\ldots, u_m\}$ and $S=\{s_1,s_2,\ldots, s_m\}$ be the user set and Web service set, where m and n are the number of users and Web services, respectively. Then, m users generate an $m \times n$ user-service QoS rating matrix $R=\{r_{u,s}\}_{m\times n}$ based on the submitted set of all rating triples, and each entry $r_{u,s}$ in R represents the value of a certain QoS attribute of Web service s observed by service user u.

As discussed in Section 1, the large number of deployed and small number of used Web services indicates that R will have many *null* values. To recommend the most appropriate Web services to the most suitable service users, our first task is to predict the unknown ratings in R. With the notations defined above, the main task of this study is formally defined as follows: given a sparse QoS rating matrix R, and for a given user u and service s, what is user u's likely observed QoS value for the Web service s?

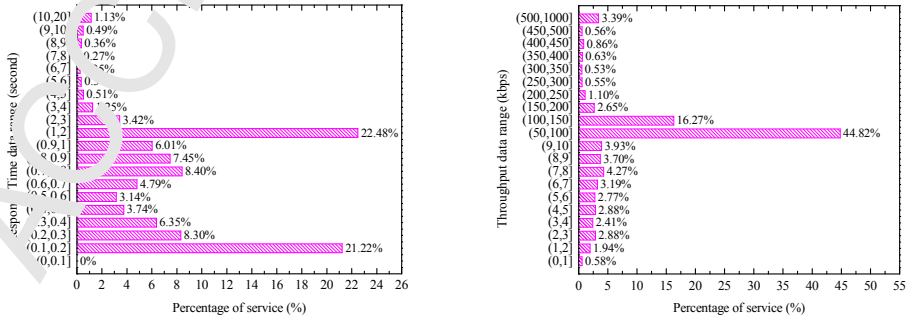
3.2 Analysis of QoS data range

In Section 1.2, we discover the phenomenon of acclimatization in CF method caused by the difference of data ranges based on a given toy example in Table 1. The empirical analysis has demonstrated that this phenomenon dramatically harms the prediction accuracy of CF method. To examine whether the fact that Web services QoS data vary in very large different ranges exists in reality, we conduct detailed data analysis based on public real-world dataset WSDream [38]. WSDream contains 1,974,675 (*User*, *Service*, *Rating*) triples contributed by 339 service users on 5,825 Web services, including two QoS attributes: response time and throughput. If the large data range in WSDream is confirmed, then it will become our research motivation for designing an effective method to resolve the influence of QoS data range while improving accuracy. To study the characteristics of QoS data with response time and throughput, we pose the following two questions:

(1) What is the QoS data distribution of Web services?

(2) Does a large range of QoS data exist between Web services?

To answer the first question, we count the number of average response time of Web services that vary in the data ranges from (0, 0.1] to (10, 20] due to the maximum response time is 20s and count the number of average throughput of Web services that vary in the data ranges from (0, 1] to (500, 1000] due to the maximum throughput is 1000s. Figs. 1(a) and 1(b) plot the proportions of Web services with different data ranges in response time and throughput, respectively.



(a) Response Time (b) Throughput

Fig.1. Distribution of Web service QoS data

From Fig. 1(a), we observe that the average response time of Web services is distributed widely from (0, 0.1]

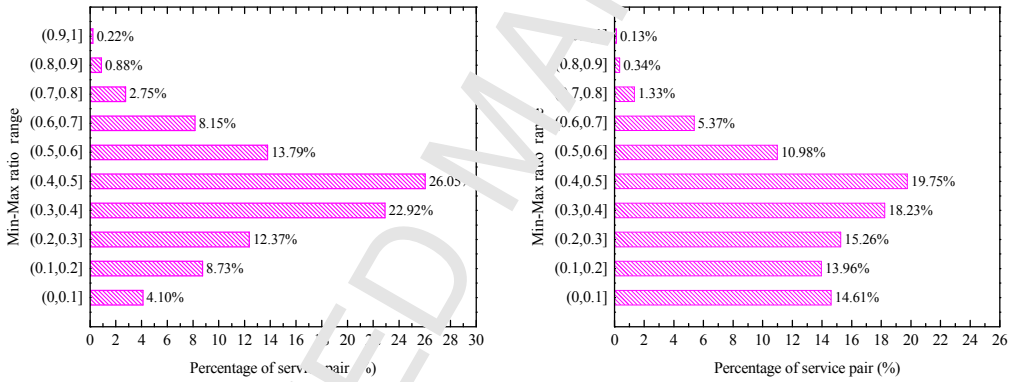
to (10, 20], and the proportions of average response time in ranges (0.1, 0.2] and (1, 2] are 21.22% and 22.48%, respectively. From Fig. 1(b), the average throughput of Web services is also distributed in different data ranges, and the proportions of the average throughput in ranges (50, 100] and (100, 150] are 44.82% and 16.27%, respectively. These results suggest that the QoS data of Web services are not focused on a specified data range but are spread discretely in different data ranges, demonstrating that the QoS data of Web services exhibit large service effects. Thus, we draw the following **Observation 1** of QoS data in relation to data range.

Observation 1 *The QoS data of Web services are distributed in different ranges discretely.*

Observation 1 contributes to the possibility of a large difference of QoS data among Web services. To evaluate the rating gap among Web services, we define the ratio *MinMax*(*s*, *t*) of Web service *s* and *t* as follows:

$$MinMax(s,t)=\frac{\sum_{u\in U_s\cap U_t}\frac{min(r_{u,s},r_{u,t})}{max(r_{u,s},r_{u,t})}}{|U_s\cap U_t|} \tag{6}$$

where $r_{u,s}$ and $r_{u,t}$ are the ratings of service *s* and *t* observed by user *u*, respectively; *min*() and *max*() are the functions of solving the maximum and minimum values, respectively; and U_s and U_t represent the set of users who have invoked services *s* and *t*, respectively. The value of *MinMax*(*s*, *t*) is between 0 and 1, and a larger value indicates a closer QoS rating between service and *s* and *t*, and vice versa. Therefore, *MinMax*(*s*, *t*) can present the overall experience difference between two Web services for the same users that have been have accessed. The proportions of the number of service pairs in different *MinMax*(*s*, *t*) ranges using response time and throughput attributes in WSDream are shown in Fig.2.



(a) Response Time (b) Throughput

Fig.2. Distribution of Min-Max ratio among Web service

According to Fig. 2, 0.22% service pairs have the values of *MinMax*(*s*, *t*) in (0.9, 1] with response time dataset and only 0.13% service pairs have the values of *MinMax*(*s*, *t*) in (0.9, 1] with throughput dataset, which means that very few of Web services have close QoS data values, while the QoS values of majority of Web services have a wide data range. Note that is *MinMax*(*s*, *t*) has 4.1% and 14.61% service pairs in (0, 0.1] with response time and throughput datasets, respectively, which indicates a very large difference between the data range of two Web services. Thus, we draw the **Observation 2** of QoS data in relation to *MinMax*(*s*, *t*).

Observation 2 *A large range gap of QoS data exists among Web services in reality.*

Positive answers with **Observations 1** and **2** to the two questions provide evidence that Web service QoS data vary with large different data ranges, thereby causing the acclimatization in CF method, that is, two Web services in WS-Dream may obtain very different QoS values for the same users even if they have a large similarity. Our analysis in Section 1.2 has demonstrated that it will significantly deteriorate the prediction accuracy of CF method. With this validation, we next investigate how to model an accurate similarity calculation and alleviate the influence of QoS data range in the CF method.

4 Method

4.1 Collaborative QoS prediction framework

Our collaborative Web service QoS prediction framework is presented in Fig. 3. The workflow of our proposed framework can be separated into three stages: similarity calculation, neighborhood selection, and collaborative prediction.

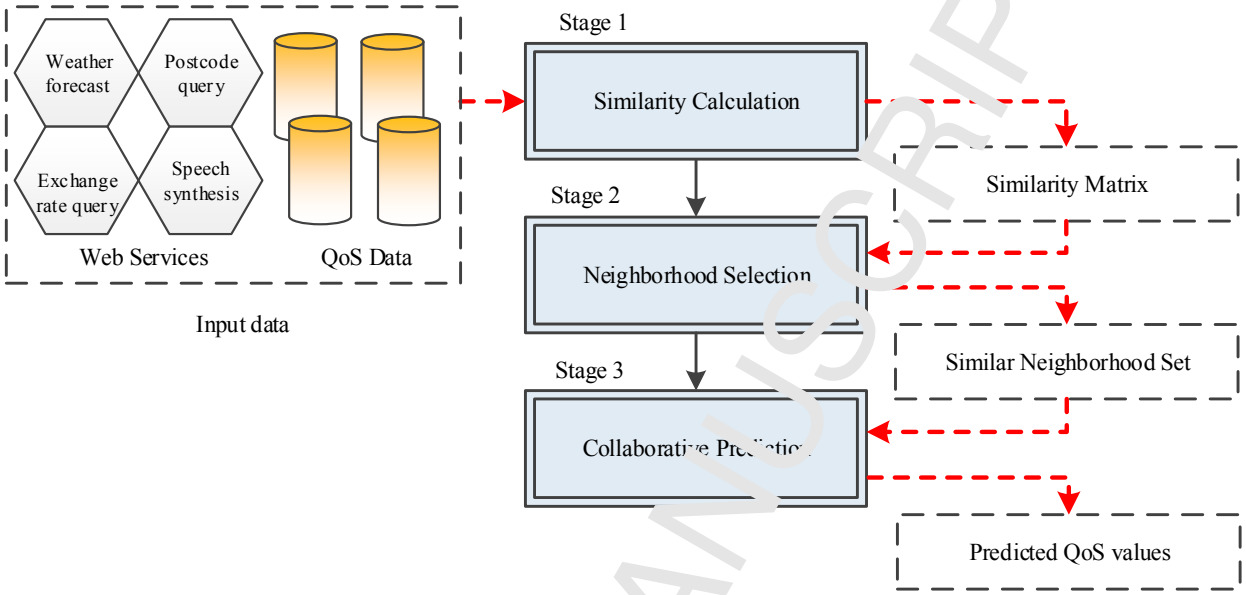


Fig. 3. Framework of collaborative Web service QoS prediction

Stage 1 *Similarity calculation*. The observed QoS rating triples as input data undergo preprocessing to form the desired user-service rating matrix R. This matrix R is then calculated based on the designed similarity calculation model to generate the similarity matrix among Web services.

Stage 2 *Neighborhood selection*. The obtained similarity matrix is used to select the similar neighbors, where we design two feasible strategies, namely, top-K and threshold based strategies, for neighborhood selection. In the practical application, a strategy-switching mechanism can be introduced for the neighborhood selection strategy switching to obtain the desired neighbors.

Stage 3 *Collaborative prediction*. The selected neighbors can either be used directly by the neighborhood based CF method for final collaborative prediction, or be integrated into the MF model to learn a predictive model for prediction. Furthermore, a prediction switching mechanism can be introduced for neighborhood and model based CF methods switching to obtain the best result in a particular context.

4.2 Similarity calculation

The aforementioned framework shows that the first task of collaborative Web service QoS prediction is to calculate the similarity between services. Obtaining a proper similarity calculation is important because similarity is used not only in neighbor selection but also in final result prediction.

Our previous review in Section 2.1 indicate that traditional similarities, such as PCC, COS, and COSA, suffer from the co-accessed user problem and ignore the effect of data range. In general, if two Web services have a very large similarity, then they are likely to exhibit similar QoS performance for the same users, while our **Observations 1** and **2** demonstrated that two Web services may have very different QoS values for the same users even if they have a large similarity. Therefore, we have to consider the influence of data range when modeling our similarity metric *JacMinMax*. The *JacMinMax* similarity between Web services s and t is defined as follows:

$$JacMinMax(s,t) = Jaccard(s,t) \cdot MinMax(s,t) = \frac{|U_s \cap U_t|}{|U_s| + |U_t| - |U_s \cap U_t|} \cdot \frac{\sum_{u \in U_s \cap U_t} \frac{min(r_{u,s}, r_{u,t})}{max(r_{u,s}, r_{u,t})}}{|U_s \cap U_t|} = \frac{\sum_{u \in U_s \cap U_t} \frac{min(r_{u,s}, r_{u,t})}{max(r_{u,s}, r_{u,t})}}{|U_s| + |U_t| - |U_s \cap U_t|} \quad (7)$$

JacMinMax captures the influence of data range using *MinMax*(*s*, *t*) for the similarity calculation between services *s* and *t*. In addition, to emphasize the significance of co-accessed users, we have added Jaccard similarity to our model. The proposed *JacMinMax* similarity is reasonable because of the following conditions: (1) *Jaccard*(*s*, *t*) and *MinMax*(*s*, *t*) have a value between 0 and 1, respectively, and the product of *Jaccard*(*s*, *t*) and *MinMax*(*s*, *t*) also has a value between 0 and 1; thus, the essential nature of similarity is not lost when the larger value indicates a more similar QoS performance between services *s* and *t* and vice versa. (2) Introducing of the *Jaccard* similarity resolves the co-accessed user problem, thereby preventing the overestimation of the similarity between services when they are actually not similar but happen to have similar QoS performance observed by a few co-accessed users. (3) Introducing *MinMax* alleviates the influence of data range, thereby avoiding the inaccurate characterization of the similarity relationship between services when they are actually similar but have very different QoS values.

4.3 Neighborhood selection

Leveraging the wisdom of neighborhood is the core idea of CF method. The accuracy of CF method strongly relies on the selected neighbors. Generally, Web services know only the users who invoke them, while they are not clear who have the similar QoS performance. To filter the dissimilar neighbors and obtain the similar neighbors, we design two strategies: top-K and threshold filtering based neighborhood selection.

4.3.1 Top-K based neighborhood selection

Top-K based neighborhood selection strategy selects neighbors based on the idea that limiting neighborhood size can result in more accurate predictions as the neighbors with low correlation may introduce more noise than signal into the process. Thus, the neighbors of Web service *s* using top-K based neighborhood selection strategy can be obtained as follows:

$$Nei(s) = \{t \mid t \in Top-K(s), JacMaxMin(s,t) > 0, t \neq s\} \quad (8)$$

For a target service, only a list of the *K* nearest neighbors and their respective *JacMinMax* similarity are kept. A tradeoff exists between efficiency and accuracy when deciding the value of *K*. If *K* is too large, then an excessive amount of memory would be needed to store the neighbors and the prediction calculation will be slow. On the other hand, selecting an extremely value for *K* may filter some similar neighbors and reduce the prediction accuracy. To avoid problems with efficiency and accuracy, we will tune the value of *K* in the experiment and obtain the optimal one that ensures the prediction accuracy with a smaller number of neighbors.

4.3.2 Threshold filtering based neighborhood selection

Instead of keeping a fixed number of similar neighbors, threshold filtering based neighborhood selection strategy selects the neighbors whose similarity is greater than a given positive threshold value θ. Thus, the neighbors of Web service *s* using the threshold filtering based neighborhood selection strategy can be obtained as follows:

$$Nei(s) = \{t \mid JacMinMax(s,t) > \theta, \ \ \theta > 0 \ \ and \ \ t \neq s\} \quad (9)$$

As only the most significant neighbors are kept, threshold filtering strategy is more flexible than the top-K strategy. Similar to selecting the value of *K*, the right value of θ can be tuned experimentally.

4.4 Collaborative prediction

Once a list *Nei*(*s*) of similar neighbors has been selected for each service, the missing QoS values can be

predicted by combing the QoS values of services in *Nei*(*s*). To accurately generate predictions and leverage the wisdom of the selected neighbors, we adopt two widely used methods: neighborhood based and model based collaborative prediction.

4.4.1 Neighborhood based collaborative prediction

To predict unknown Web service QoS values, neighborhood based collaborative prediction is typically performed by computing the weighted average of the QoS values of the neighboring services and using similarity as the weight through the following equation:

$$\hat{r}_{u,s} = \overline{r}_s + \frac{\sum_{t \in Nei(s)} JacMinMax(s,t)(r_{u,t} - \overline{r}_t)}{\sum_{t \in Nei(s)} JacMinMax(s,t)} \tag{10}$$

where $\hat{r}_{u,s}$ is the predicted QoS value of the missing entry $r_{u,s}$ in user-service matrix R, and $\overline{r}_t$ is the average QoS value of Web services t observed by different users.

4.4.2 Model based collaborative prediction

Different from neighborhood based prediction that directly uses the known QoS values of neighbors to make predictions, the model based method applies the overall known QoS values to learn a predictive model for predictions. In the model based approach, dimensionality reduction technique is mainly used, which compresses user-service matrix R into a low-dimensional representation in terms of latent factors. Specifically, the model factorizes the user-service rating matrix R by a multiplicative of two d-rank matrices, U and S, where $U \in \mathbb{R}^{d \times m}$ and $S \in \mathbb{R}^{d \times n}$ are user-specific and service-specific feature matrices, respectively. Thus, a missing value $\hat{r}_{u,s}$ can be restored it by the inner product of user feature vector p_u and service feature vector q_s.

$$\hat{r}_{u,s} = \alpha(\mu + b_u + b_s) + (1-\alpha) p_u q_s \tag{11}$$

where μ is the global average of ratings in R, b_u and b_s are the biases of user and service from the average, respectively; α is used to control the importance of user and service biases. Thus, $\mu + b_u + b_s$ can be a bias baseline predictor to account for the influence of the QoS data range, and our previous works have demonstrated the effectiveness of the bias baseline predictor in improving prediction accuracy [22, 36].

To model the significance of selected neighbors *Nei*(*s*), we systematically integrate the *Nei*(*s*) into the MF model by capitalizing on the advantages of neighborhood and latent feature approaches. Thus, with the benefit of similar neighbors in *Nei*(*s*), $\hat{r}_{u,s}$ can be obtained as follows:

$$\hat{r}_{u,s} = \alpha(\mu + b_u + b_s) + (1-\alpha) p_u (q_s + |Nei(s)|^{-\frac{1}{2}} \sum_{t \in Nei(s)} JacMinMax_{s,t} q_t) \tag{12}$$

where a rating frequency based strategy $|Nei(s)|^{-\frac{1}{2}}$ is introduced to avoid the over-fitting problem, and $JacMinMax_{s,t}$ is the relative weight inside the selected neighborhood, which is computed as follows:

$$JacMinMax_{s,t} = \frac{JacMinMax(s,t)}{\sum_{t \in Nei_s} JacMinMax(s,t)} \tag{13}$$

The main intuition behind the model in Eq. (12) is that similar neighbors not only have similar QoS values, but also have similar latent features. Thus, the latent feature of service s is modeled as $(q_s + |Nei(s)|^{-\frac{1}{2}} \sum_{t \in Nei(s)} JacMinMax_{s,t} n_t)$ by capturing the significance of its neighbors.

To calculate the latent feature vectors of b_u, b_s, p_u, q_s, and q_t, with the known QoS values and obtained neighbors, our solution can be mathematically formulated as solving the following optimization problem:

$$\ell = \min \frac{1}{2}\sum_{u=1}^{m}\sum_{s=1}^{n} I_{u,s}(r_{u,s} - \hat{r}_{u,s})^2 + \frac{\lambda}{2}(|S_u|^{-\frac{1}{2}}\|b_u\|_F^2 + |U_s|^{-\frac{1}{2}}\|b_s\|_F^2 + |S_u|^{-\frac{1}{2}}\|p_u\|_F^2 + |U_s|^{-\frac{1}{2}}\|q_s\|_F^2 + \sum_{t\in Nei(s)}|Nei(t)|^{-\frac{1}{2}}\|q_t\|_F^2) \quad (14)$$

where $I_{u,s}$ is an indicator equal to 1 when user u has accessed service s, and is equal to 0 otherwise. $\|\cdot\|_F^2$ is the Frobenius norm. The regularization term parameters λ are positive to control the over fitting issue.

The optimization problem defined in Eq. (14) is not jointly convex in vector $V=\{b_u, b_s, p_u, q_s, q_t\}$, but they are convex in each of them individually with the other fixed. Therefore, to find a local solution, we design a stochastic gradient descent algorithm to derive the relevant parameters in vector V using an iterative manner with user-service matrix R until convergence. The details are shown in Algorithm 1.

Algorithm 1 Model learning based on stochastic gradient decent

Input: training matrix R, similar neighbors matrix *NeiList*, and respective *JacMinMax* similarity *SimList*

Output: learned user and service feature vectors p_u, q_s, and bias vectors b_u, b_s

initialize b_u, b_s, p_u, q_s, q_t randomly;
while not convergent **do**
  calculate $\hat{r}_{u,s}$ with Eq. (12);
  $err = r_{u,s} - \hat{r}_{u,s}$; //calculate the prediction error
  b_u += γ*(err*α - λ$|S_u|^{-\frac{1}{2}} b_u$); //update user bias b_u, γ is learn rate used for each iteration
  b_s += γ*(err*α - λ$|U_s|^{-\frac{1}{2}} b_s$); //update service bias b_s
  p_u += γ*(err*(1-α) (q_s + $|Nei(s)|^{-\frac{1}{2}} \sum_{t\in Nei(s)} JacMinMax_{s,t} q_t$) - λ$|S_u|^{-\frac{1}{2}} p_u$); //update user feature
  q_s += γ*(err*(1-α)p_u - λ$|U_s|^{-\frac{1}{2}} q_s$); //update service feature
  foreach (neighbor t in *NeiList*(s))
    q_t += γ*(err*(1-α) p_u * $|Nei(s)|^{-\frac{1}{2}}$ * $JacMinMax_{s,t}$ - λ$|Nei(t)|^{-\frac{1}{2}} q_t$); //update neighborhood feature
  end foreach
end while

4.5 Complexity analysis

To evaluate the feasibility of the proposed method in large datasets, we analyze the computation complexity of the proposed QoS prediction method. We assume there are m users and n Web services in the training matrix R.

4.5.1 Complexity of neighborhood based collaborative filtering

The computation complexity of neighborhood based CF is mainly determined by calculating the similarity between Web services in Section 4.2 and collaborative QoS prediction in Section 4.4.1. As there are at most m intersecting users between Web service s and Web service t, the computation complexity of *JacMinMax*(s,t) is $O(m)$. To obtain the most similar neighbors for the target Web service based on the *JacMinMax* similarity, we need to calculate the similarities of the target Web service with all n Web services in the training matrix R. Therefore, the computation complexity of similarity is $O(m*n)$. When we predict the missing values in the training matrix using Eq.(10), the computational complexity is $O(m*n*K)$, where K is the number of neighbors. Thus, the overall computational complexity of the neighborhood based CF can be relaxed to $O(m*n*K)$. As K is a limited small constant in our experiment, showing that the computational complexity of the neighborhood based CF depends on the number of users and services. Moreover, the similarity calculation for the training matrix R can be pre-computed and re-computation is required only when the training matrix R is updated, thereby enabling real-time prediction for active user.

4.5.2 Complexity of model based collaborative filtering

The computation complexity of model based CF is determined primarily by evaluating the feature variables

b_u, b_s, p_u, q_s, and q_t using Algorithm 1. We first initialize b_u, b_s, p_u, q_s, and q_t with small random values. Then, every non-empty entry in R is iterated to update these feature variables with a limited number of iterations *IterNum*. Parameter γ is the iteration step size used to control the speed of iteration. Thus, the computation complexities of variables b_u, b_s, p_u, q_s, and q_t are $O(\rho_R * IterNum)$, $O(\rho_R * IterNum)$, $O(d * \rho_R * IterNum + d * \rho_R * K * IterNum)$, $O(d * \rho_R * IterNum)$, and $O(d * \rho_R * K * IterNum)$, respectively, where ρ_R is number of non-empty entries in training matrix R, d is the dimensionality of the feature space, and K is the number of neighbors. Thus, the overall computation complexity of Algorithm 1 can be relaxed to $O(d * \rho_R * (K+1) * IterNum)$. As the values of K, d, and *IterNum* are small constants in our experiment, thereby insuring that our algorithm is linear with respect to the size of non-empty entries in R, with the result that our proposed method is scalable to large datasets.

5 Experiment

The main goal of this experimental evaluation is to show that our proposed approach alleviates the influence of QoS data range, thereby improving the Web service QoS prediction accuracy. In particular, our experiments are designed to answer the following questions:

(1) How does the proposed method perform compared with many state-of-the-art counterparts and similarity calculation methods?
(2) What is the performance of the proposed method under different matrix densities?
(3) How does the neighborhood size affect the performance of the proposed method?
(4) What is the performance of the proposed method using threshold filtering strategy?

5.1 Preparation

Data set. We used the Web service QoS dataset provided by WSDream [38] to evaluate the performance of the proposed approach. This dataset includes 1,974,675 response time (RT) and throughput (TP) invocation records with values varying in (0, 20] and (0, 1000], respectively.

To enable our experiments to follow the reality in which service users only access a few Web services, the user-service rating matrix is very sparse. We randomly remove some QoS data in WSDream to sparse R, with different matrix densities: 2%, 4%, 6%, and 8%. Specifically, a 2% matrix density means that 98% known QoS data are randomly removed from the dataset, while the remaining 2% data are kept as the training data to build the prediction model and the removed 98% data are used to evaluate our approach.

Evaluation metrics. We adopt the most widely used rating-oriented evaluation metrics of mean absolute error (MAE) and root mean square error (RMSE) to measure the accuracy of predicted values with respect to the held-out QoS values [39]. If N held-out values exist, then MAE and RMSE are computed as follows:

$$MAE = \frac{1}{N}\sum \left| r_{u,s} - \hat{r}_{u,s} \right| \tag{15}$$

$$RMSE = \sqrt{\frac{1}{N}\sum \left| r_{u,s} - \hat{r}_{u,s} \right|^2} \tag{16}$$

where $r_{u,s}$ and $\hat{r}_{u,s}$ are the true and predicted ratings, respectively; and N is the number of test ratings. The smaller values of MAE and RMSE indicate better accuracy. MAE provides equal weights to all the individual differences. RMSE is a good measure of accuracy when detecting relatively large errors.

Parameter settings. We investigated how accurate results can be influenced by selecting various alternatives for data preprocessing and setting the parameters involved. Then, we assessed the performance of our approach and the other methods according to their respective best settings. Table 4 presents the parameters used to make prediction in our approach.

Table 4 Experimental parameter settings

K	α	λ	learning rate for RT dataset γ_{rt}	learning rate for TP dataset γ_{tp}	number of factors d	maximum iteration
20	0.3	1	0.012	0.0004	12	16

All the experiments have been implemented on Windows7 64bit OS with Intel Core i7-3370 3.4GHz processor and 8GB RAM.

5.2 Comparison with different prediction methods

To verify the effectiveness of the JacMinMax similarity both in neighborhood and model based CF prediction, we name these predictions as JacMinMax-N and JacMinMax-M, respectively. We compare JacMinMax-N and JacMinMax-M with the following well-known six methods.

(1) GMEAN, which uses the global average QoS value over the overall training data to perform prediction.

(2) UMEAN, which uses the average QoS value of the target user to perform prediction.

(3) SMEAN, which uses the average QoS value of the target service to perform prediction.

(4) WSRec, which is a neighborhood based method that uses PCC to calculate similarity and combines the user and item based CF to perform prediction [20].

(5) MF, which is the basic model based method that learns the latent features merely based on known ratings [40].

(6) GeoMF extends the MF method and integrates the geographical neighborhood to build the predictive model [36].

To measure the ability of our approach using JacMinMax similarity and the above competing methods in predicting QoS values effectively, matrix density is conducted at 2%, 4%, 6%, and 8%, and the neighborhood size is 20. The experimental results are reported in Table 5.

Table 5 Comparisons with different prediction methods

Dataset	Metric	MAE				RMSE			
	Density	2%	4%	6%	8%	2%	4%	6%	8%
Response Time (seconds)	GMEAN	1.1524	1.0680	1.0372	1.0202	1.9822	1.9619	1.9692	1.9689
	UMEAN	0.8788	0.8726	0.8707	0.8674	1.8608	1.8545	1.8549	1.8546
	SMEAN	0.9293	0.7874	0.7267	0.7234	1.7312	1.5709	1.5620	1.5458
	WSRec	0.7155	0.7031	0.6847	0.6688	1.6501	1.6323	1.6105	1.5956
	MF	0.6838	0.5896	0.5031	0.5798	1.6873	1.4192	1.3796	1.3531
	GeoMF	*0.6817*	*0.5545*	*0.5192*	*0.4951*	*1.6220*	*1.3356*	*1.2868*	*1.2566*
	JacMinMax-N	0.7054	0.5615	0.5154	0.4968	**1.5233**	**1.3341**	**1.2839**	**1.2496**
	JacMinMax-M	**0.6477**	**0.5387**	**0.5109**	**0.4944**	1.6549	1.3973	1.3317	1.2804
	Improvement	5.0%	2.93%	1.59%	0.14%	6.08%	0.11%	0.22%	0.55%
Throughput (kbps)	GMEAN	54.4526	54.3440	54.3348	53.5546	110.8809	110.8550	110.7489	110.7156
	UMEAN	54.3053	54.2663	54.1374	53.8965	110.9331	110.5548	110.4366	110.5302
	SMEAN	40.8816	37.8098	37.479	36.6806	88.7501	86.3266	86.2388	85.2898
	WSRec	34.0657	32.9123	32.8649	31.6322	74.1994	68.9989	67.1462	66.4901
	MF	27.9710	24.8403	22.9333	21.8176	71.8208	61.6144	57.1301	55.9187
	GeoMF	*28.8920*	*22.0557*	*21.6523*	*20.8053*	*70.9999*	*57.9733*	*56.3709*	*54.6836*
	JacMinMax-N	28.4276	23.1130	21.6254	20.0820	77.6437	65.5997	62.2390	59.6284
	JacMinMax-M	**27.3356**	**20.9673**	**18.6742**	**17.3659**	**69.0212**	**54.7006**	**50.6070**	**48.2456**
	Improvement	5.38%	4.93%	13.75%	16.53%	2.78%	5.64%	10.22%	11.7732%

From Table 5, the following observations can be made:

(1) JacMinMax-N obtains smaller RMSE values consistently under different matrix densities compared with all methods in response time dataset, JacMinMax-M obtains the smaller MAE and RMSE values consistently with different configurations in throughput dataset, thereby indicating a better prediction accuracy. This is because QoS data among Web services have a large data range, and our analysis in Section 1.2 has shown that CF method does not work well in this scenario. However, our proposed JacMinMax similarity can effectively alleviate the influence of the QoS data range while improving the prediction accuracy.

(2) The MAE and RMSE values are decreased with the increase of matrix density, which indicates that the denser is the QoS rating matrix, the more accurate are the prediction results. This finding can be explained by the fact that a denser matrix can contribute more significant information and learn a more accurate prediction model. This phenomenon suggests that we should encourage more service users to contribute their observed QoS data to the UDDI to improve the prediction accuracy. Conversely, this approach can also enhance user acceptability of the prediction results, because a large amount of data indicates a more reliable conclusion.

(3) Generally, the model based CF method outperforms the neighborhood based CF method because the former uses the entire training data to learn the predictive model whereas the latter only uses the information of similar neighbors in prediction. However, in the response time dataset, when JacMinMax-N with JacMinMax-M are compared, the former obtains much better prediction accuracy with respect to RMSE, thereby demonstrating a stronger ability to detect relatively large errors in the response time dataset. The performance of the neighborhood based CF heavily relies on the similarity and can be greatly improved by designing an appropriate similarity calculation model based on the characteristic of the QoS dataset, thus leveraging the limited information of similar neighbors to obtain more interpretable and better results is possible.

(4) An interesting observation is that our JacMinMax method does not obtain much better improvements with the increase of available QoS ratings in the response time dataset. However, JacMinMax-M obtains much better predictions with the increase of QoS ratings available in the throughput dataset. A possible reason for this phenomenon is that only 4.1% service pairs have $MinMax(s, t)$ in (0, 0.1] with the response time dataset, while this percentage reaches 14.61% with throughput dataset, as shown in Fig. 2. This phenomenon indicates that when the QoS dataset has a larger data range among Web services, our JacMinMax can effectively perceive these QoS data differences between Web services with the increase of available QoS data, thereby effectively improving the accuracy of our method.

5.3 Comparison with different similarity calculations

As mentioned, similarity plays a double role in CF methods. On the one hand, it allows the selection of neighbors whose QoS values are used directly in the prediction; and on the other hand, it provides the means to give more or less importance to these neighbors in the prediction. The computation of similarity is one of the most important aspects of the in CF method because it can have a considerable effect on both accuracy and performance. To study the impact of similarity calculation in the CF method, we compare our improved similarity JacMinMax with four classic similarity measures described in Section 2.1. Comparison results with different similarity calculations using response time and throughput datasets are shown in Table 6.

Table 6 Comparisons with different similarity calculations

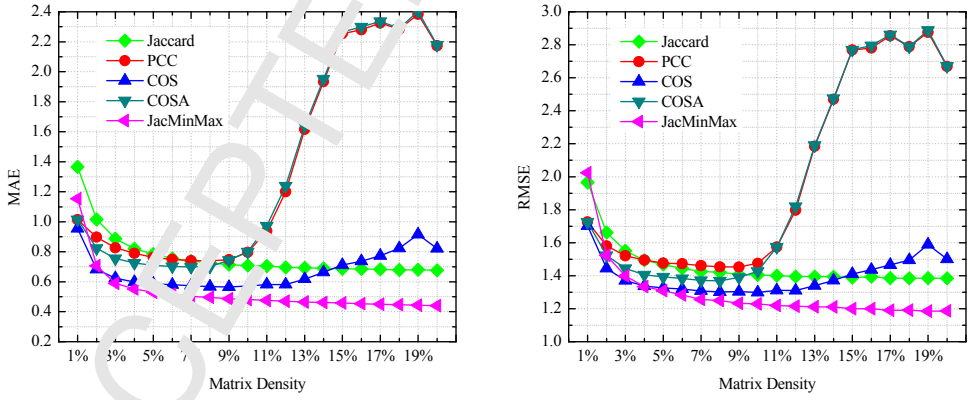
Dataset	Collaboration Type	Metric	MAE				RMSE			
		Density	2%	4%	6%	8%	2%	4%	6%	8%
Response Time (seconds)	Neighborhood based	Jaccard	1.0158	0.8204	0.7551	0.726	1.6627	1.4942	1.4451	1.4217
		PCC	0.8980	0.7903	0.7488	0.7398	1.5810	1.4953	1.4724	1.4541
		COSA	0.8236	0.7252	0.7018	0.7008	1.5237	1.4062	1.3841	1.3692
		COS	***0.6830***	*0.5987*	*0.5807*	*0.5678*	***1.4452***	*1.3376*	*1.3199*	*1.3023*
		JacMinMax	0.7054	**0.5515**	**0.5154**	**0.4968**	1.5233	**1.3341**	**1.2839**	**1.2496**
		Improvement	-3.28%	7.88%	11.24%	12.50%	-.5.40%	0.26%	2.72%	4.04%
	Model based	Jaccard	0.6605	0.5413	0.5147	0.4970	1.6933	1.4019	1.3362	1.2911
		PCC	0.6517	0.5401	0.5131	0.4959	1.6701	1.4026	1.3345	1.2869
		COSA	0.6512	0.5437	0.5127	0.4989	1.6799	1.4101	1.3336	1.2941
		COS	*0.6566*	*0.5412*	*0.5167*	*0.5000*	*1.6862*	*1.4035*	*1.3414*	*1.2982*
		JacMinMax	**0.6475**	**0.5382**	**0.5109**	**0.4944**	**1.6549**	**1.3973**	**1.3317**	**1.2804**
		Improvement	1.38%	0.55%	1.12%	1.12%	1.8%	0.44%	0.72%	1.37%

		Jaccard	43.3319	39.7636	35.7484	33.5232	80.0765	74.4868	70.3463	68.2022
		PCC	35.2079	33.8197	34.5898	37.4917	74.6539	69.0983	68.649	70.5257
	Neighborhood	COSA	34.1701	33.2133	33.5202	35.9448	73.7055	68.3219	67.5586	68.7657
	based	COS	*30.9104*	*30.3753*	*30.0824*	*31.5469*	***71.3633***	*67.1337*	*67.3324*	*71.8961*
Throughput		JacMinMax	**28.4276**	**23.1130**	**21.6254**	**20.0820**	77.6437	**69.3297**	**62.2390**	**59.6284**
(kbps)		Improvement	8.03%	23.09%	28.11%	36.34%	-8.80%	2.71%	7.56%	17.06%
		Jaccard	27.8485	22.3298	20.0535	18.9499	70.4657	57.3815	52.5709	50.8031
		PCC	27.7979	22.3562	19.7848	18.7831	70.4267	57.6064	51.9206	50.1298
	Model	COSA	27.6532	21.9055	19.6432	18.9460	70.1485	57.5841	51.801	50.1967
	based	COS	*28.1033*	*22.2349*	*19.8610*	*18.7106*	*70.9166*	*58.3337*	*52.5142*	*49.9001*
		JacMinMax	**27.3356**	**20.9673**	**18.6742**	**17.3659**	**69.8112**	**56.7006**	**50.6070**	**48.0456**
		Improvement	2.73%	5.70%	5.97%	7.18%	2.07%	2.79%	3.63%	3.71%

As shown in Table 6, JacMinMax is more accurate than other compared similarity computations for the neighborhood and model based CF methods under all settings except the neighborhood based method when the matrix density is 2% compared with COS similarity. The results indicate that JacMinMax is capable of improving the prediction accuracy of both neighborhood and model based CF methods, and the designed JacMinMax enables the CF method to adapt to the QoS data with a large data range. When the matrix density increase from 2% to 8%, the MAE and RMSE become smaller because the similarities between services become steadier as the available of QoS values increase. Moreover, JacMinMax has much larger improvement under throughput dataset than under response time dataset because the former has a larger has lager data range among Web services, which is consistent with the observation in Table 5.

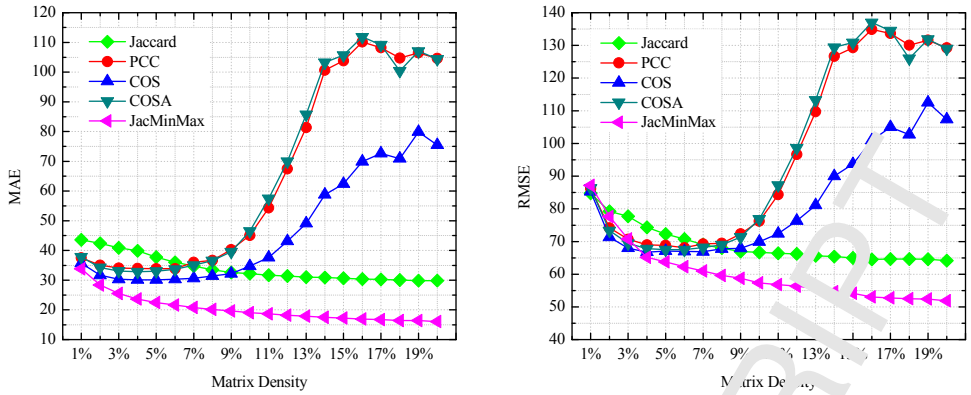
5.4 Impact of matrix density

Matrix density is the percentage of the kept ratings in the user-service matrix, which indicates how much available information can be used in making predictions. To study the impact of matrix density, we vary the matrix density from 1% to 20% with a step value of 1%. We set the neighborhood size at 20. The experimental results using response time and throughput datasets are illustrated in Figs. 4 and 5, respectively.



(a) MAE (b) RMSE

Fig.4. Impact of matrix density using response time dataset



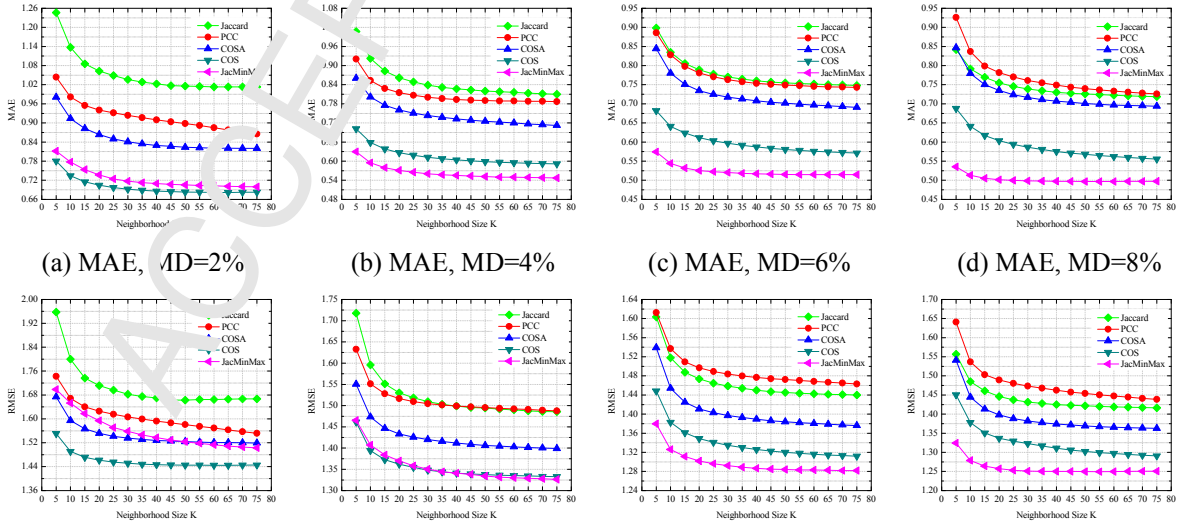
(a) MAE (b) RMSE

Fig.5. Impact of matrix density using throughput dataset

Figs. 4 and 5 show that with the increasing matrix density, both MAE and RMSE values decrease rapidly at first, and our JacMinMax does not produce better results than COS when the matrix density is less than 3% because extremely small available ratings cannot enable JacMinMax to effectively capture the influence of data range among Web services. However, when the matrix density is greater than 3%, JacMinMax can effectively alleviate the impact of data range and show its superiority on improving the prediction accuracy of CF compared with other similarity calculation models. Moreover, when the matrix density grows larger, the decline of MAE and RMSE slows down, thereby suggesting that improving the accuracy of CF methodologies by adding available QoS ratings is effective when available ratings are very small. When there more QoS ratings are available, we should improve the method itself to further improve the performance of CF. Care must be taken when matrix density is larger than 10%, we observed that MAE and RMSE values with PCC, COSA and COS similarities increase. A possible reason for this phenomenon is that QoS data vary in different data ranges, PCC, COSA, and COS have large similarity weights between services although they have considerably different QoS values, thereby degrading the prediction accuracy. In summary, our proposed JacMinMax works well as the matrix density increases, indicating that this method can be applied to both sparse and dense datasets.

5.5 Impact of neighborhood size on neighborhood based CF

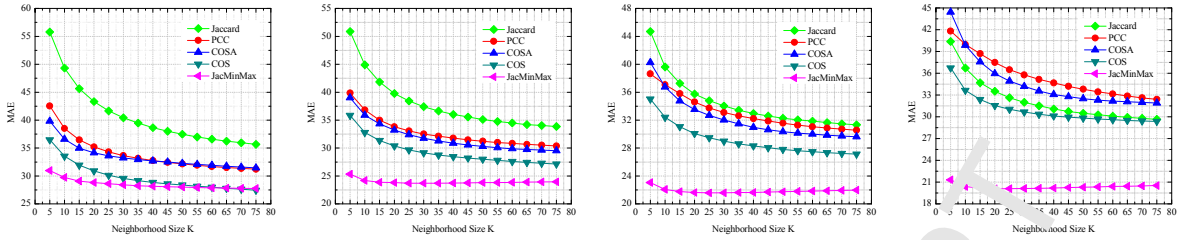
Neighborhood size K determines the number of neighbors used to help us make predictions. An extremely small or large value of K will affect the prediction accuracy and efficiency. To study the impact of neighborhood size, we tune it from 5 to 75 with a step value of 5. Figs. 6 and 7 illustrate the experimental results of the impact of neighborhood size on neighborhood based CF using response time and throughput datasets, respectively.



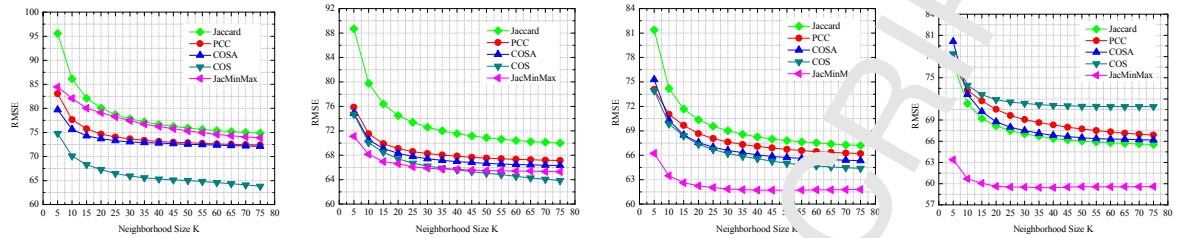
(a) MAE, MD=2% (b) MAE, MD=4% (c) MAE, MD=6% (d) MAE, MD=8%

(e) RMSE, MD=2% (f) RMSE, MD=4% (g) RMSE, MD=6% (h) RMSE, MD=8%

Fig.6. Impact of neighborhood size on neighborhood based CF using response time dataset



(a) MAE, MD=2% (b) MAE, MD=4% (c) MAE, MD=6% (d) MAE, MD=8%



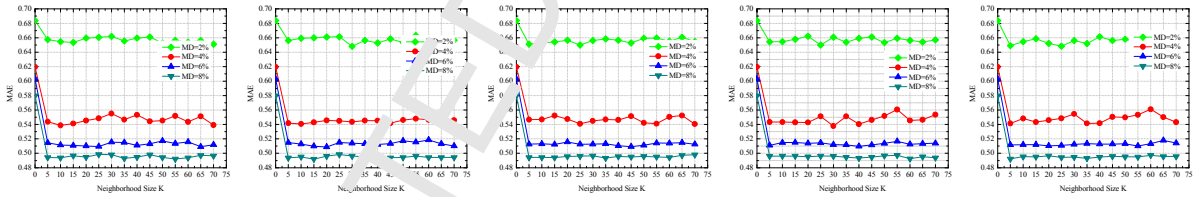
(e) RMSE, MD=2% (f) RMSE, MD=4% (g) RMSE, MD=6% (h) RMSE, MD=8%

Fig.7. Impact of neighborhood size on neighborhood based CF using throughput dataset

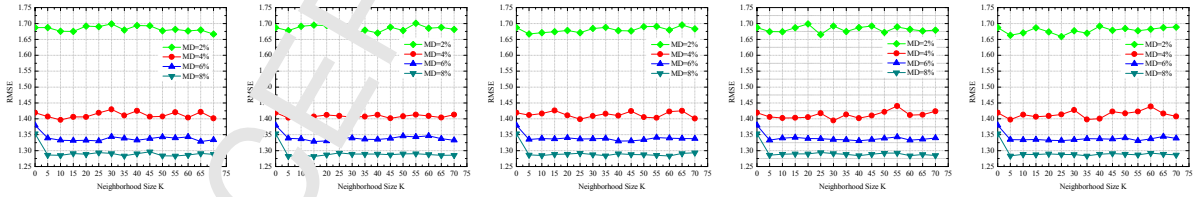
Figs. 6 and 7 show that both MAE and RMSE decrease rapidly at first as the neighborhood size increases from 5 to 20. This result indicates that leveraging the wisdom of similar neighbors can effectively improve the prediction accuracy. However, when the neighborhood size exceeds 20, the decline of MAE and RMSE under all settings slows down because too many similar neighbors cannot effectively improve the performance of CF. In addition, our complexity analysis of neighborhood based CF in Section 4.5.1 shows that the large number of neighbors indicates greater computation time to make prediction. This condition explains why we have selected K = 20 in our experimental settings.

5.6 Impact of neighborhood size on model based CF

To investigate the effect of neighborhood size on model based CF, we tune the neighborhood size from 0 to 70 with a step value of 5. Figs. 8 and 9 present the experimental results.

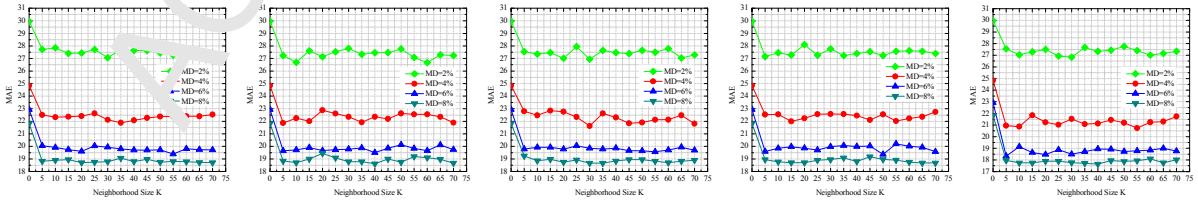


(a) MAE, Jaccard (b) MAE, PCC (c) MAE, CosA (d) MAE, COS (e) MAE, JacMinMax

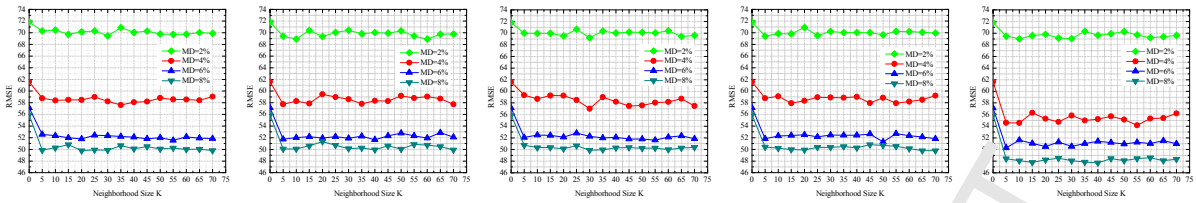


(f) RMSE, Jaccard (g) RMSE, PCC (h) RMSE, CosA (i) RMSE, COS (j) RMSE, JacMinMax

Fig.8. Impact of neighborhood size on model based CF using response time dataset



(a) MAE, Jaccard (b) MAE, PCC (c) MAE, CosA (d) MAE, COS (e) MAE, JacMinMax



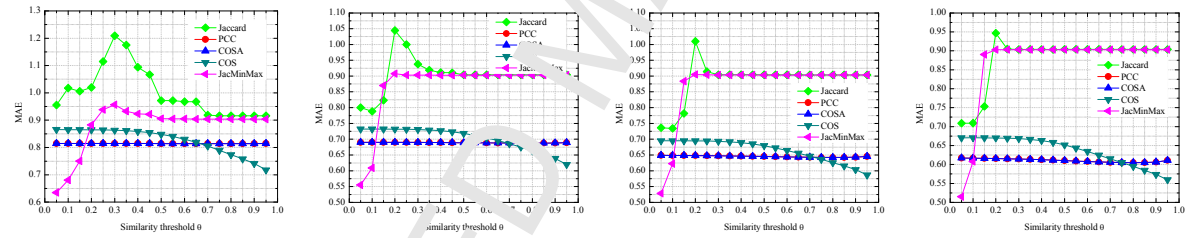
(f) RMSE, Jaccard (g) RMSE, PCC (h) RMSE, CosA (i) RMSE, COS (j) RMSE, JacMinMax

Fig.9. Impact of neighborhood size on model based CF using throughput dataset

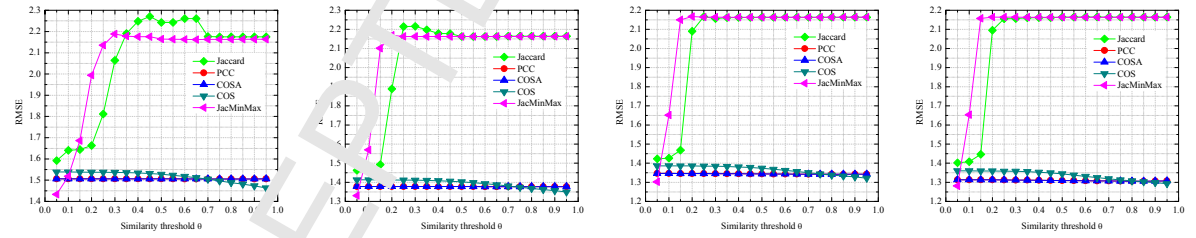
As shown in Figs. 8 and 9, when the neighborhood size is zero, the MAE and RMSE values are relatively large. However, when the neighborhood size is greater than zero, the MAE and RMSE values drop rapidly. Thus, we can conclude that a similar neighborhood has a beneficial influence on improving the prediction accuracy of model based CF. We have also observed that when the number of neighbors increases from 5 to 70 in both response time and throughput datasets, the accuracy of different similarity computation methods under different matrix density fluctuates because the few higher similar neighbors are diluted by the weak ones. Therefore, to enhance the accuracy of model based CF, we should not hope to find more neighbors but to improve the model itself. Moreover, according to our discussion of the complexity of the model based CF in Section 4.5.2, too many neighbors will result in greater time cost to learn the predictive model, thus, we set the same neighborhood size as that of the neighborhood based CF in our experiments.

5.7 Performance with threshold filtering strategy

Instead of keeping a fixed number of top-*K* neighbors, threshold filtering strategy adopts a more flexible way of selecting neighbors whose similarity weight is greater than a given threshold value θ. To study the impact of similarity threshold, in this experiment, we tune θ from 0.05 to 0.95 with a step value of 0.05.

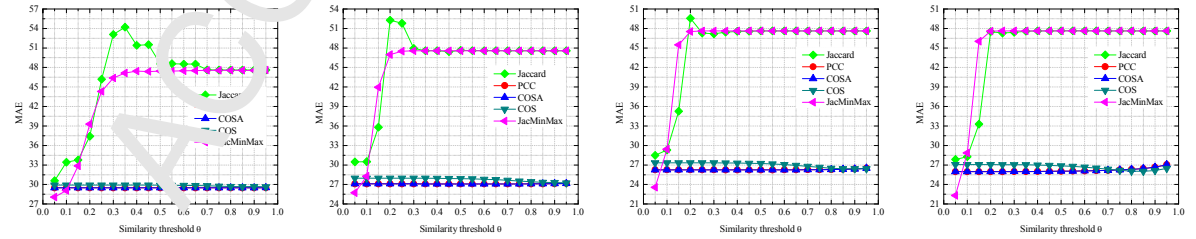


(a)MAE, MD=2% (b)MAE, MD=4% (c)MAE, MD=6% (d)MAE, MD=8%

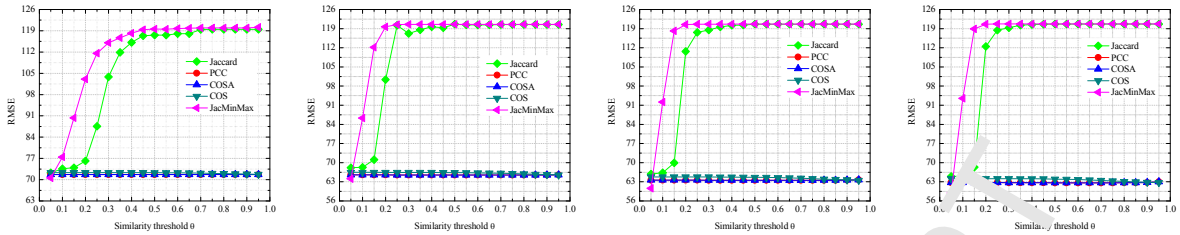


(e)RMSE, MD=2% (f)RMSE, MD=4% (g)RMSE, MD=6% (h)RMSE, MD=8%

Fig.10 Impact of similarity threshold using response time dataset



(a)MAE, MD=2% (b)MAE, MD=4% (c)MAE, MD=6% (d)MAE, MD=8%



(e)RMSE, MD=2% (f)RMSE, MD=4% (g)RMSE, MD=6% (h)RMSE, MD=8%

Fig.11. Impact of similarity threshold using throughput dataset

Figs.10 and 11 show that MAE and RMSE in JacMinMax have the smallest values when θ is set at 0.05 under different matrix densities in both response time and throughput datasets, thereby validating the effectiveness of our proposed similarity computation with threshold filtering strategy. In general, the value of Jaccard similarity is relatively small within a sparser matrix density. Thus, as we gradually increase the value of θ, many dissimilar neighbors occur in the neighborhood, which harms the performance of CF method.

5.8 Case study

To demonstrate the effectiveness of our proposed JacMinMax similarity on alleviating the influences of QoS data range in practical application, we then solve the problem of predicting r_{14} in Table 1. According to the workflow of our proposed method, we first calculate the JacMinMax similarities between Web services using Eq. (7). Table 7 reports the calculated similarity matrix.

Table 7 JacMinMax similarities between Web services

	s_1	s_2	s_3	s_4
s_1	1.0000	0.0750	0.0337	0.4374
s_2		1.0000	0.5246	0.2000
s_3			1.0000	0.0877
s_4				1.0000

As shown in Table 7, *JacMinMax* (s_1, s_2) = 0.0750, *JacMinMax* (s_1, s_3) = 0.0337, *JacMinMax* (s_1, s_4) = 0.4374, indicating that s_2, s_3, and s_4 are positively correlated with s_1. Furthermore, user u_4 has accessed these three services. Thus, we select $\{s_2, s_3, s_4\}$ as the neighborhood of service s_1 for the final prediction of $rt_{4,1}$.

For collaborative prediction, we adopt our neighborhood based prediction method in Eq.(10) and calculate $rt_{4,1}$ as follows:

$$\begin{aligned} rt_{4,1} &= \bar{r}_1 + \frac{JacMinMax(s_1,s_2)(rt_{4,2}-\bar{r}_2)+JacMinMax(s_1,s_3)(rt_{4,3}-\bar{r}_3)+JacMinMax(s_1,s_4)(rt_{4,4}-\bar{r}_4)}{JacMinMax(s_1,s_2)+JacMinMax(s_1,s_3)+JacMinMax(s_1,s_4)} \\ &= 0.2+\frac{0.075\times(3-2.25)+0.0337\times(4-3.3)+0.4374\times(0.8-0.45)}{0.075+0.0337+0.4374} \\ &= 0.6265 \end{aligned} \quad (17)$$

Table 8 Comparison of predicted response time for r_{14}

	Jaccard	PCC	COS	COSA	JacMinMax	Improvement
Predicted $r_{4,1}$	*0.7875*	0.8117	0.8017	1.3428	**0.6265**	
Prediction error	*0.4875*	0.5117	0.5017	1.0428	**0.3265**	33.02%

Table 8 presents the predicted values of $rt_{4,1}$ with different similarity calculations. The results show that JacMinMax has the smallest prediction error, which illustrates that this method can effectively handle the impact of data range while improving the prediction accuracy. It is worth noting that the predicted values of $rt_{4,1}$ are all calculated using the same neighbors, namely, s_2, s_3, and s_4, indicating that they are indeed the neighbors of s_1. And a finer measurement is required to characterize the relationship between s_1 and these neighbors for accurate QoS prediction. Moreover, JacMinMax has an improvement of 33.02% with respect to prediction error compared with

the best one. Although the percentage of relative improvements is small, Koren [41] has pointed out that even minor improvements in prediction error may lead to significant differences of recommendations in practice.

6 Conclusion and future work

In this paper, we have presented improvements to a data range driven similarity calculation model in collaborative filtering approach that appears to produce better prediction of Web service QoS than existing techniques. Particularly, we discover a phenomenon of acclimatization in CF methods faced with QoS data that have large data ranges, which degrades the prediction performance and impedes the applicability of CF. To address this issue, we conducted a detailed analysis of the public real-world WSDream dataset and verified the fact that Web service QoS data exhibit a large service effect when large different data ranges exist. Motivated by this fact, we designed a Min-Max function model as part of the similarity calculation, which not only explains why neighbors are generally misunderstood in many CF methods but also yields a more accurate Web service QoS prediction. Furthermore, two effective neighborhood selection strategies were introduced for selecting more reliable similar neighbors. Then, these obtained neighbors were systematically integrated into the neighborhood and model based collaborated filtering methods, and a switching mechanism could be used in neighbor selection and collaborative prediction switching. Our evaluation shows that the proposed approach can effectively solve the acclimatization problems in CF and achieve completive prediction accuracy as demonstrated on the set of benchmarks. We believe that this work demonstrates the potential impact of large data range on accurate Web service QoS prediction of CF. In the near future, we plan to collect emerging Web services and Web APIs, and monitor and obtain their QoS data for constructing a new real-world QoS dataset. We can further validate the universality of our proposed model by using the newly collected QoS datasets.

Acknowledgments

We thank the anonymous reviewers for their insightful comments and suggestions. This work is supported by National Natural Science Foundation of China (61772450), China Postdoctoral Science Foundation Grant (2018M631764), Hebei Postdoctoral Research Program (B2018003009), Colleges and Universities in Hebei Province Science and Technology Research Project (QN2016073), and the Doctoral Fund of Yanshan University (BL18003).

References

[1] H. Haas, A. Brown. Web services glossary, W3C Working Group Note (11 February 2004) [EB/OL], https://www.w3.org/TR/ws-gloss/ (2004).

[2] A. Bouguettaya, M. P. Singh, M. N. Huhns, et al. A service computing manifesto: the next 10 years [J], Communications of the ACM, vol.60, no.4, pp. 64-72, 2017.

[3] M. P. Papazoglou, P. Traverso, S. Dustdar, et al. Service-oriented computing: state of the art and research challenges [J], Computer, vol.40, no.11, pp.38-45, 2007.

[4] J. Yong, G. Fortino, W. Shen, et al. Special issue on service-oriented collaborative computing and applications [J], IEEE Transaction On Services Computing, vol.11, no.2, pp.277-278, 2018.

[5] Y. Duan, G. Fu, N. Zhou et al. Everything as a service (XaaS) on the cloud: origins, current and future trends [C], Proc. of IEEE Conference on Cloud Computing, pp.621-628, 2015.

[6] C. Kotas, T. Naughton, N. Imam. A comparison of Amazon Web Services and Microsoft Azure cloud platforms for high performance computing[C], Proc. of International Conference on Consumer Electronics, pp.1-4, 2018.

[7] T. Yu, Y. Zhang, K. J. Lin. Efficient algorithms for Web services selection with end-to-end QoS constraints [J], ACM Transactions on the Web, vol.1, no.1, pp.1-26, 2007.

[8] S. Sun, J. Zhao. A decomposition-based approach for service composition with global QoS guarantees [J], Information Sciences, vol.199, pp.138-153, 2012.

[9] A. Bekkouche, S. M. Benslimane, M. Huchard. QoS-aware optimal and automated semantic web service

composition with user's constraints [J], Service-Oriented Computing and Applications, vol.11, no.2, pp.183-201, 2017.

[10] N. Anithadevi, M. Sundarambal. A design of intelligent QoS aware web service recommendation system [J], Cluster Computing, no.1, pp.1-10, 2018.

[11] C. Yu, L. Huang. A Web service QoS prediction approach based on time- and location-aware collaborative filtering [J], Service-Oriented Computing and Applications, vol.10, no. 2, pp.135-149, 2016.

[12] Y. Xu, J. Yin, S. Deng, et al. Context-aware QoS prediction for web service recommendation and selection [J], Expert Systems With Applications, vol.53, pp.75-86, 2016.

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

[14] X. Yang, Y. Guo, Y. Liu. A survey of collaborative filtering based social recommender systems [J]. Computer Communication, vol. 41, pp. 1-10, 2014.

[15] B. M. Sarwar, G. Karypis, J. A. Konstan, et al. Item-based collaborative filtering recommendation algorithms [C]. Proc. of International World Wide Web Conferences, pp. 285-295, 2001.

[16] J. Liu, M. Tang, Z. Zheng, et al. Location-aware and personalized collaborative filtering for Web service recommendation [J]. IEEE Transactions on Services Computing, vol. 9, no. 5, pp. 686-699, 2016.

[17] S. Lin, Y. Yang, C. Lo, et al. A social trust based recommendation mechanism for Web service dynamic collaboration[C]. Proc. of International Conference on Service Oriented Computing, pp. 318-322, 2013.

[18] A. Kalai, C. A. Zayani, I. Amous, et al. Expertise and trust-aware social Web service recommendation[C]. Proc. of International Conference on Service Oriented Computing, pp. 517-533, 2016.

[19] G. Vadivelou, E. Llavarasan. Fusion of Pearson similarity and Slope One methods for QoS prediction for web services[C]. Proc. of International Conference on Contemporary Computing & Informatics, pp. 1118-1124, 2015.

[20] Z. Zheng, H. Ma, M. Lyu, et al. QoS-aware Web service recommendation by collaborative filtering [J]. IEEE Transactions on Services Computing, vol. 4, no. 2, pp. 140-152, 2011.

[21] F. M. Harper, J. A. Konstan, The MovieLens datasets: history and context [J]. ACM Transactions on Interactive Intelligent Systems, vol. 5, no. 4, pp. 1-19, 2016.

[22] Z. Chen, L. Shen, F. Li, ULMF:Web service QoS collaborative prediction with explicit ratings and implicit user location[J]. Journal of Internet Technology, vol. 17, no.6, pp. 1195-1205, 2016.

[23] J. Bobadila, F. Serradilla, A new collaborative filtering metric that improves the behavior of recommender systems[J], Knowledge Based Systems, vol. 23, no.6, pp.520-528, 2010.

[24] X. He, Y. Luo, Mutual information based similarity measure for collaborative filtering[C], Proc. of IEEE Conference on Progress in Informatics and Computing, pp.1117-1121, 2010.

[25] Y. Zheng, L. Zhang, Z. M., Recommending friends and locations based on individual location history[J], ACM Transactions on The Web, vol. 5, no. 1, pp. 1-44, 2011.

[26] G. Sidorov, A. Gelbukh, H.Gomezadorno, et. al. Soft similarity and soft cosine measure: similarity of features in vector space model[J], Computación Y Sistemas, vol. 18, no. 3, pp. 491-504, 2014.

[27] H. J. Ahn, A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem [J]. Information Sciences, vol. 178, no.1, pp. 37-51, 2008.

[28] L. Shao, J. Zhang and Y. Wei, Personalized QoS prediction for Web service via collaborative filtering [C], Proc. of IEEE Conference on Web Services, pp.439-446, 2007.

[29] L. Zhang, B. Zhang, Y. Liu, et al. A Web service QoS prediction approach based on collaborative filtering[C]. Proc. of International Conference on IEEE Asia-Pacific Services Computing, pp. 725-731, 2010.

[30] H. E, J. Tong, M. Song, et al. QoS prediction algorithm used in location-aware hybrid Web services [J], Journal of China Universities of Posts and Telecommunications, vol. 22, no. 1, pp. 42-49, 2015.

[31] X. Ning, C. Desrosiers, G. Karypis, A comprehensive survey of neighborhood-based recommendation

methods [B], Recommender Systems Handbook, Second Edition, Springer US, pp.37-76, 2015.
[32] Z. Chen, L. Shen, F. Li, Web service QoS collaborative prediction approach considering range of QoS data [J], Computer Integrated Manufacturing Systems, vo.23, no.1, pp. 215-224, 2017.
[33] W. Lo, J. Yin, S. Deng, et al. An extended matrix factorization approach for QoS prediction in service selection[C]. Proc. of International Conference on Services Computing, pp. 162-169, 2013.
[34] Z. Zheng, H. Ma, M. R. Lyu, et al. Collaborative Web services QoS prediction via neighborhood integrated matrix factorization[J], IEEE Transaction on Services Computing, vol. 6, no.3, pp.1-10, 2013.
[35] J. Xu, Z. Zheng, M. R. Lyu. Web service personalized quality of service prediction via reputation-based matrix factorization [J], IEEE Transaction on Services Computing, vol. 6, no.3, pp.1-10, 2015.
[36] Z. Chen, L. Shen, F. Li, et al. Your neighbors alleviate cold-start: on geographical neighborhood influence to collaborative web service QoS prediction [J], Knowledge-Based Systems, vol.138, pp.188-201, 2017.
[37] Y. Feng, Q. Li. The distributed UDDI system model based on service oriented architecture[C]. Proc. of International Conference on Software Engineering, pp. 585-589, 2016.
[38] Z. Zheng, Y. Zhang, M. R. Lyu. Investigating QoS of real-world Web services [J], IEEE Transaction on Services Computing, vol.7, no.1, pp.32-39, 2014.
[39] J. L. Herlocker, J. A. Konstan, L. G. Terveen, et al. Evaluating collaborative filtering recommender systems [J], ACM Transaction on Information Systems, vol. 22, no. 1, pp. 5-53, 2004.
[40] G. Guo, J. Zhang, Y. Neil. TrustSVD: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings[C], Proc. of National Conference on Artificial Intelligence, pp.123-129, 2015.
[41] Y. Koren. Factor in the neighbors: scalable and accurate collaborative filtering [J]. ACM Transactions on Knowledge Discovery form Data, vol. 4, no. 1, pp. 1-24, 2010.

Zhen Chen received his Ph.D. and B.S. in computer application technology from Yanshan University in China, at 2017 and 2010. He is doing a postdoctoral research in the College of Computer Science and Engineering, Yanshan University, China. He is currently working on service computing, cloud computing and collaborative computing.

Limin Shen received his B.S. and Ph.D. degrees in Computer Science and Technology from Yanshan University, China. He is a professor and PhD supervisor in College of Computer Science and Engineering, Yanshan University, China. His main research interests include service computing, collaborative computing, and cooperative defense. Dr. Shen is a member of IEEE.

Feng Li received his Ph.D. and M.S. in computer application technology from Yanshan University in China, at 2012 and 2007, respectively, and B.S. in Computer Science and Technology from Qufu normal University in China at 2004. He is a lecturer in department of Computer and Communication Engineering at Northeastern University. His research interests include network security, trust computing and management, and service computing. Dr. Li is a member of ACM.

Zhen Chen

Limin Shen

Feng Li

Your Neighbors Are Misunderstood: On Modeling Accurate Similarity Driven by Data Range to Collaborative Web Service QoS Prediction

Highlights

- We observe Web service QoS data exhibit a large service effect with different data ranges.
- We discover a phenomenon of acclimatization in CF faced with QoS data that have large data ranges.
- We design a simple yet effective similarity model to alleviate the influence of QoS data range.
- Prediction accuracy is enhanced by leveraging the wisdom neighbors obtained using JacMinMax similarity.