International Journal of u- and e- Service, Science and Technology
Vol.7, No.3 (2014), pp.303-318
http://dx.doi.org/10.14257/ijunesst.2014.7.3.26

Predicting Web Service QoS via Combining Matrix Factorization with Network Location

Li Zhou[1,2], Zhibo Song,[1,2] Suichu Zhai[3], Tan Xiao[3] and Yuyu Yin[1,2,4*]

[1]*School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China*
[2]*Key Laboratory of Complex System Modeling and Simulation, Ministry of Education*
[3]*Hangzhou Power Supply Cooperation, Hangzhou, China*
[4]*Electric Engineering School, Zhejiang University, Hangzhou, China*
{zhouli,s.zb.com,yinyuyu}@hdu.edu.cn;{yyy718,xiaotan}@gmail.com

Abstract

With the increasing abundance of Web Services across Internet, Quality of Service (QoS)-based service recommendation has become a hot issue. It is necessary to predict the missing values of QoS for service recommendation. Because Web services run on the Internet, their network locations may be anther critical factor for QoS prediction. Although there have existed many works on QoS prediction, few consider the influence of the network locations of users or Web Services. In this paper, we propose a novel collaborative QoS prediction framework with network location-based regularization (NLBR). We first elaborate the popular Matrix Factorization (MF) model for missing values prediction. Then, by taking advantage of the local connectivity between Web services users, we incorporate network location information to identify the neighborhood. We conduct the experiments on a public large-scale real-world QoS dataset, Experiments show that our proposed approaches have the better prediction performance compared with the existed approaches.

Keywords: *Web service, QoS prediction, Matrix Factorization, Network Location*

1. Introduction

Web Services are software components designed to support interoperable machine to machine interaction over a network, and through standard Web protocol to provide services [1]. Quality of Service (QoS) is usually used to describe the non-functional characteristics of Web Services. With the increasing abundance of Web Services on the World Wide Web, studies on QoS become more and more attractive. In the recent years, a number of QoS-based approaches have been applied to Web Service composition [2], Web Service selection [3],Web service recommendation [4] and so on.

Since a growing number of Web Services that provide similar functionalities but different quality properties of services, it is very important to recommend the best service from the candidate services considering their QoS. To carry out a QoS-based service recommendation, we firstly need to predict missing QoS values of services. Collaborative Filtering (CF) algorithms has been widely used to predict missing values in commercial recommender

[*] Corresponding Author: yinyuyu@hdu.edu.cn

systems [5-7]. These algorithms have also been applied to Web services research domain recent years.

CF approaches can be used to predict QoS values of a target service for an active user. The main idea is to identify a group of users who have similar QoS experiences to the active user or a group of services which have similar QoS records to the target service. CF approaches usually conduct Pearson Correlation Coefficient (PCC) [5] calculations to identify a similar collection for the active users and the target service. However, CF approaches have the unsatisfactory prediction performance under the condition of large dataset or data sparseness. Because: (1) we may not find out a similar collection for the active users and a target service if the dataset is too sparse. (2) It is memory and time consuming largely to find out a set of similar users or services when the dataset is too large.

QoS of Web Services mainly includes response time, throughput, availability, *etc.*, Values of these factors are usually highly dependent on network environment, which users and services are located in. Since local user share the same IT infrastructures, such as router, network workloads, and so on, thus they obtain similar values of QoS when invoking the same Web service. However, there are rare approaches considering the network location of users, which can be useful to improve the prediction performance. Based on the above analysis, we propose a novel collaborative QoS prediction with network location-based regularization (NLBR). We first elaborate the Matrix Factorization (MF) model [8] for missing values prediction. Then by understanding the local connectivity between Web services users, NLBR incorporates network location information to identify the neighborhood rather than PCC. Based on the idea that users in the same neighborhood of network tend to receive similar QoS values, a network location-based regularization term is used to revamp the classic MF model. And then we show that the computational complexity of our proposed approaches is linear with the input size, and thus NLBR can scale to very large datasets.

The major contributions of this paper include the following:

(1) We elaborate the idea of utilizing the network locations of Web services users for the QoS prediction problem.

(2) We systematically discuss how to design a network location-based regularization term to capture the latent relationship inside a neighborhood.

(3) A highly scalable implementation allows linear time and space complexity, thus our model is very efficient and can scale to very large datasets.

The rest of this paper is organized as follows: Section 2 states the motivating scenario. Section 3 introduces network location information representation and process. Section 4 details the concept of Matrix Factorization. Section 5 introduces how to revamp the Matrix Factorization model with a novel network location-based regularization terms. Section 6 discusses specific issues in our framework. Section 7 presents the experimental results. Section 8 introduces some related work, and finally, Section 9 concludes this paper.

2. A Motivating Scenario

In this section, we present a scenario to explain the motivation of our work. Users and Web services are distributed all over the world. As shown in Figure 1, we describe five users and five web services. Net 1 and Net 2 are located in China while Net 4 and Net 5 are located in the United States. Each network contains one or more users and services. Suppose that Service 1 and 2 are to provide the railway timetable for Web services, while Service 3 and 4 are to provide the web service of English and Chinese translation.

Since the railway timetable service is associated with region, it’s highly for a user to choose the Web services near his location. In this scenario, Chinese users tend to choose Service 1 while users in the United States tend to choose Service 2. This suggests that if the user needs to use the web services related with location, it will naturally choose the web service near his location. As Service 3 and 4 provide English and Chinese translations which have no connection with location. Considering the network environment, such as response time, throughput, *etc.*, due to network delays, the greater the distance between user and service, the performance will be worse. So the user tends to choose the web service located in his location. In other words, a user gets better service when he selects the web server near his location services than runs on a remote web server. As User 1 and User 2 in the same network, so they will have similar performance when they use the Service 1-4. That is to say, the users with near location have the similar QoS values when invoking the same web service.

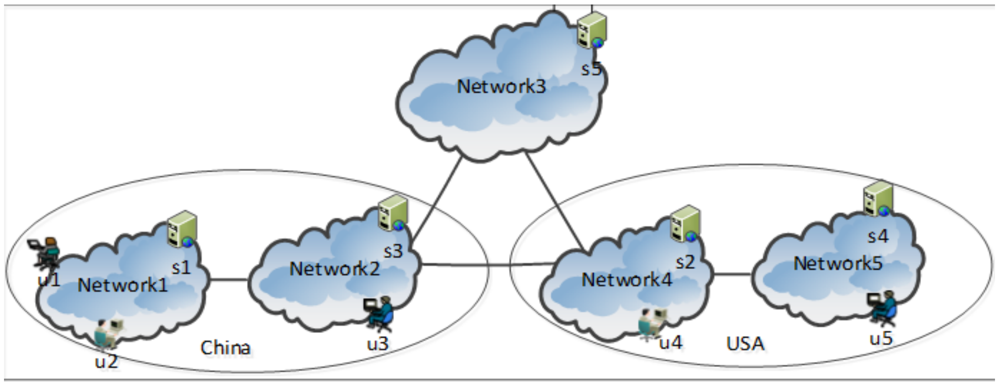


Figure 1. A Motivating Scenario

We find it help to recommend web services for the active user that were liked by other users who located in near to him in terms of performance by the observation above. Therefore it is very meaningful to consider the network location information, when predicting the missing values of QoS. Thus, it can not only reduce the search scope, but also can eliminate users who are really not QoS experiences but happen to invoke a few common services.

Our goal is to make more accurate prediction of QoS values using network locations of users. In order to achieve this goal, we first need to solve the following questions: 1) how to describe the network location information of users? 2) How to obtain location information? 3) How to use network location information to revamp MF model so that to improve the performance of predicting QoS values? 4) How to design the experiments for performance evaluation?

3. Location Information Representation, Acquisition and Processing

Wei Lo, *et al.*, [9] proposed a method using altitude and latitude of users to calculate the Euclidean distance between users to identify a neighborhood who located in near to the active user. This seems to be reasonable but may cause inaccuracies in reality due to the unique features of the network. If the two users with the near location use different broadband network operators, and even if they are using the same broadband network operator but not under the same router, network environment can have the very big difference, and so do the QoS values. Therefore, even if two users are located in the similar location, they may seem far away from each other due to their computers are within different network. So we first need to look for a new method to describe network locations of users.

We use the Autonomous System (AS) to represent the network locations of users[10]. Autonomous System (AS) is a collection of connected Internet Protocol (IP) routing prefixes under the control of one or more network operators that presents a common, clearly defined routing policy to the Internet. And each AS must have an officially registered Autonomous System Number (ASN). A unique ASN is allocated to each AS for use in Border Gateway Protocol routing. ASN uniquely identifies each neighbor region on the Internet.

In an autonomous system, all routers must be connected and running the same routing protocol, so they have the same network transmission performance (response time, throughput, *etc.*,). Therefore, in the same AS, QoS values are similar when users invoking the same web services. This is why we choose ASN to represent network locations of users.

We can easily get the ASN of users. The dataset contains the IP addresses of all users. There are many available services or online tools for this purpose. We use the robtex (http://www.robtex.com) to find the ASN. In our experiment, we used crawlers written by the Python program to obtain the ASN rapidly and accurately according to the users' IP addresses.

4. Matrix Factorization Model

Matrix Factorization model is a popular and effective tool to predict the missing values. This model maps both users and items to a joint latent factor space of a low dimensionality f , such that user-item interactions can be captured as inner products in that space. The premise behind a low-dimensional factorization approach is that there are only a few factors affecting the user-item interactions, and a user's interactive experience is influenced by how each factor applies to the user.

In this paper, We use the data sets including M service users and N Web services, the relationship between users and services can be denoted an $M \times N$ matrix R , and called user-item matrix. Every entry in the matrix r_{ui} represents a QoS value (*e.g.*, response-time, throughput, *etc.*,) which shows Web service i observed by user u . Considering the influence of the user and the service f features, this matrix can be divided into two low-dimensional matrices.

$$R = P^T Q \tag{1}$$

Where $P \in R^{f \times m}$ and $Q \in R^{f \times n}$ are the two low-dimensional matrices. Then, the prediction result user u for Web Service i is $\hat{R}(u,i) = \hat{r}_{ui}$, which can be calculated by the following formula:

$$\hat{r}_{ui} = p_u^T q_i = \sum_{f=1}^{F} p_{uf} q_{if} \tag{2}$$

In Eq. (2) p_{uf} and q_{uf} are the model parameters. Where p_{uf} measures the relation of user u interest and latent factor f and q_{if} describes the relation of latent factor f and Web service i . Then matrix P and Q are learned from minimizing RMSE using samples of training set.

We use the RMSE as evaluation indicators, and find the right matrix P and Q to minimize the prediction error of the training set; it can also minimize the prediction error of the test set. Therefore, the loss function is shown as following:

$$\min_{P,Q} \psi(P,Q) = \frac{1}{2}\sum_{u=1}^{m}\sum_{i=1}^{n} \| r_{ui} - P_u^T Q_i \|_F^2 \tag{3}$$

In the Eq. (3), the factor 1/2 is used to simplify subsequent partial derivative. And$\|\cdot\|_F$ denotes the Frobenius norm. However, in real word cases, the matrix R only contains a few Web services records. This sparse problem is solved as follows:

$$\min_{P,Q} \psi(P,Q) = \frac{1}{2}\sum_{u=1}^{m}\sum_{i=1}^{n} I_{ui}^{R} \| r_{ui} - P_u^T Q_i \|_F^2 \tag{4}$$

However, directly optimizing loss function (4) causes over fitting in learning. So, $\lambda(\|P\|_F^2 + \|Q\|_F^2)$ is introduced to overcome this shortness, where λ is a regularization parameter. The whole loss function is defined as:

$$\min_{P,Q} \psi(P,Q) = \frac{1}{2}\sum_{u=1}^{m}\sum_{i=1}^{n} I_{ui}^{R} (r_{ui} - P_u^T Q_i)^2 + \frac{\lambda}{2}(\|P\|_F^2 + \|Q\|_F^2) \tag{5}$$

To minimize the loss function of above, we can use the stochastic gradient descent method. The stochastic gradient descent method first finds the steepest descent direction by calculating parameters of the partial derivative, and then the parameters are optimized continuously by the iterative method.

There are two groups of parameters P_u and Q_i in the loss function defined by Eq. (5), Gradient steepest descent method first need compute the parameters' partial derivative respectively. The computational formula is as following:

$$\frac{\partial \psi}{\partial P_u} = \sum_{i=1}^{n} I_{ui}^{R}(-Q_i)(r_{ui} - P_u^T Q_i) + \lambda P_u$$
$$\frac{\partial \psi}{\partial Q_i} = \sum_{u=1}^{m} I_{ui}^{R}(-P_u)(r_{ui} - P_u^T Q_i) + \lambda Q_i \tag{6}$$

In the learning algorithm, we first need to initialize the matrix P and Q .There is many methods of initialization. Generally the two matrices are filled with random numbers. From our experiments, we found that the random numbers are proportional to $1/sqrt(F)$ can get better results.

5. Network Location-Based Regularization

In this section, we discuss how to incorporate the network location information as regularization term to revamp the traditional Matrix Factorization model in detail. Subsection A introduces the neighborhood computation based on network location information. And then subsection B proposes the NLBR approach.

5.1. Notations and Definitions

The following are important notations used in the rest of paper:

$U = \{u_1, u_2, \ldots, u_m\}$ is a set of service users, where m is the total number of service users in the dataset.

$S = \{s_1, s_2, \ldots, s_n\}$ is a set of Web services, where n is the total number of Web services in the dataset.

$R = \{r_{ui} \mid 1 \le u \le m, 1 \le i \le n\}$ is the user-service matrix, where r_{ui} is the QoS value acquired from user u invoking service i . We set $r_{ui} = null$ if user u has no experiences on service i .

$A = \{ASN(u) \mid 1 \le u \le m\}$ is a set of ASNs. For a user u , $ASN(u)$ denotes the

Autonomous System Number (ASN) that user u belongs to.

$C = \{country(u) \mid 1 \le u \le m\}$ is a set of countries. For a user u , $country(u)$ denotes the country where user u is located in.

5.2. Neighborhood Similarity Computation

Pearson Correlation Coefficient (PCC) and Vector Space Similarity (VSS) are two usually employed in computing the similarity between different service users. As mentioned in work [18], PCC approach can achieve higher performance than VSS, since the former considers the differences in the user value style and can achieve high accuracy. Therefore, we use PCC for the similarity computation between user u and user v . Its computational formula is as follows:

$$sim(u,v) = \frac{\sum_{i \in I} (R_{ui} - \overline{R}_u)(R_{vi} - \overline{R}_v)}{\sqrt{\sum_{i \in I} (R_{ui} - \overline{R}_u)^2} \sqrt{\sum_{i \in I} (R_{vi} - \overline{R}_v)^2}} \tag{7}$$

Where I is the subset of Web services that are commonly invoked by user u and user v . R_{ui} Denotes the QoS values of Web service i observed by user u . $\overline{R}_u$ And $\overline{R}_v$ represent the average QoS values of user u and user v respectively. It can be seen from the Formula (7) that $sim(u,v)$ is in the interval of [-1, 1]. The larger a value is, the more similar two users are.

To identify the similar neighbors for the active user u , we according to the steps as follow:

Step 1: Search all service users who located in the same AS for the active user u , and compute the similarity between each user u and other users who located in the $ASN(u)$ regarding their historical QoS experiences based on PCC.

Step 2: AS similar as the step 1, Search all service users who located in the same country for the active user u , and compute the similarity between each user u and other users who located in the $country(u)$.

Step 3: Search all services users who not located in the $ASN(u)$ and $country(u)$, and compute the similarity between each u and other users.

After computing the degree of similarity between the active user and all other users, we get a user similarity vector. Then we adapted the traditional $Top - K$ algorithm to identify the similar neighbors' size of the active user. We set parameter K is a network location threshold to control the neighborhood size, and the similar neighbors whose value of PCC is equal to or smaller than 0 will be removed. In the process of calculation, if the size of neighbors is equal to K , we stop the following steps. This not only considers the network location, but also reduces the search scope. We denote the set of similar neighbors of user u as $G(u)$.

5.3. Network Location-Based Regularization (NLBR)

As the above analysis, it is natural to suppose that users in the same network location tend to share similar Web service invocation experience, thus they acquire the similar QoS values when invoking the same services. This indicates that the difference of user factors in the neighborhood should be minor. We convert this idea into following formula (8):

$$\min \| P_u - \frac{1}{|G(u)|} \sum_{v \in G(u)} P_v \|_F^2 \tag{8}$$

The above constraint term is used to minimize the invocation experience between a user u and its neighborhood $G(u)$. Thus, this constraint term detects strong associations among a small set of closely related users, precisely where the MF model would perform better. We add this regularization term in the MF model as follow:

$$\min_{P,Q} \psi_1(P,Q) = \frac{1}{2}\sum_{u=1}^{m}\sum_{i=1}^{n} I_{ui}^{R}(r_{ui} - P_u^T Q_i)^2 + \frac{\lambda}{2}(\|P\|_F^2 + \|Q\|_F^2)$$
$$+ \frac{\gamma}{2}\|P_u - \frac{1}{|G(u)|}\sum_{v\in G(u)} P_v\|_F^2 \tag{9}$$

Where $\gamma > 0$ is controlling the weight of this term. We use the gradient descent method to calculate its local minimum as follows:

$$\frac{\partial \psi_1}{\partial P_u} = \sum_{i=1}^{n} I_{ui}^{R}(-Q_i)(r_{ui} - P_u^T Q_i) + \lambda P_u + \gamma(P_u - \frac{1}{|G(u)|}\sum_{v\in G(u)} P_v)$$
$$\frac{\partial \psi_1}{\partial Q_i} = \sum_{u=1}^{m} I_{ui}^{R}(-P_u)(r_{ui} - P_u^T Q_i) + \lambda Q_i \tag{10}$$

6. Discussion

The main computation of NLBR approach is evaluating the object function ψ and its gradients against the variables. For NLBR, the computational complexities of evaluating the gradients $\frac{\partial \psi}{\partial P}$ and $\frac{\partial \psi}{\partial Q}$ are both $O(\omega f + |U| Kf)$ and $O(\omega f)$ respectively, where ω is the number of nonzero entries in the user-service matrix R, and K is the number of similar neighbors for the active user, $|U|$ is the number of all users in the dataset, and f is the dimensionality. Therefore the total computational complexity in one iteration is $O(\omega f + |U| Kf)$, which indicates that the computational complexity of NLBR is linear with the input size. This complexity analysis shows that our proposed approach is very efficient and can scale up to a large-scale dataset.

7. Experiments

In this section, we conduct a set of experiments to evaluate and validate the prediction accuracy of our new method. Particularly, our experiments are aiming answering the following questions: (1) how does our NLBR method compare with other well-known collaborative filtering methods? (2) How does the parameter K and γ influence prediction accuracy? (3)What is the impact of the matrix density and dimensionality on the prediction accuracy?

7.1. Experimental Setup

In our experiments, we adapt a public real world Web service QoS dataset, which is collected by Zibin Zheng, *et al.,* It contains QoS records of 1,974,675 web service invocations executed by 339 distributed service users on 5825 web services, which are transformed into two user-service matrices. The values of the two user-service matrices are response time and throughput respectively. More details about this dataset can be found in [11].

The dataset also contains IP addresses of all users and WSDLs of all web services. According to IP addresses of users and WSDLs of web services, we are able to get locational information (ASN) of all users. Through the use of our python crawler, we collect the ASN of

326 users and 5490 services, and make a new 326×5490 user-service matrix for experiments.

Our experiments are developed by using Python2.7, MySQL 5.5 and performed on a Lenovo desktop computer with configuration as: Intel Core i5-3470 3.20GHz CPU, 4GB RAM, and Windows 7 operating system.

7.2. Metrics

In our experiments, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) metrics are used to measure the prediction accuracy of our method in comparison with other collaborative filtering methods. MAE is defined as:

$$MAE = \frac{1}{N} \sum_{i,j} | R_{ij} - \hat{R}_{ij} | \tag{11}$$

where R_{ij} denotes the observed QoS values of Web service j observed by service user i , $\hat{R}_{ij}$ represents the predicted QoS values of service j for user i ,and N is the number of predicted values. The MAE places the equal weight on each individual difference in the average.

RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i,j} (R_{ij} - \hat{R}_{ij})^2} \tag{12}$$

In RMSE, the difference between a prediction result and the corresponding observed values are each squared and the averaged over the sample. Finally, the square root of the average is taken. Since the errors are squared before they are averaged, the RMSE amplifies the importance of relatively errors. Therefore it could be a good way to reflect the large errors. Smaller MAE and RMSE values represent higher prediction accuracy.

7.3. Comparison

In this section, we compare our method with the following well-known methods:

(1) UMEAN: This method uses each user’s mean QoS value on the used Web Services to predict the missing values.

(2) IMEAN: This method employs the mean QoS value of the Web service observed by other users to predict the missing values.

(3) UPCC: User-based collaborative filtering method using Pearson Correlation Coefficient [12].This method is a very classical method that employs similar users for the QoS value prediction.

(4) IPCC: Item-based collaborative filtering method using Pearson Correlation Coefficient [13].This method is widely used in e-commerce Company.

(5) UIPCC: This method combines the UPCC and IPCC results together and set a parameter to balance the value of each other [6].

(6) LACF: This method improves the UIPCC by incorporating locations of both users and services [10].

(7) SVD: This method is proposed by Koren, *et al.,* in [8]. It captures the latent structure of the original data distribution.

(8) LBR2: This method focused on capturing geographical connectivity to identify similar users, and then combine these regularization terms in Matrix Factorization framework [9].

In the real word, the user-service matrices are usually very sparse because of a service user usually only invokes a small number of web services. Therefore, in order to make our experiments more realistic, we randomly remove values to sparse the matrix. We conducted a series of experiments in the cases of response time matrix density is 5%, 10%, 15%, and 20%. For example, Matrix density 10% means that we randomly select 10% of entries for training and the remaining 90% for testing. When compared with the above methods, we use the same training and test cases. The parameter setting of our proposed methods are $K = 80$, dimensionality $f = 100$, $\gamma = 0.01$, and $\lambda = 0.001$, and more detailed analysis on parameter tunings will be provided as follow.

Table 1. Accuracy Comparison (A smaller MAE or RMSE Value Means a Better Performance)

	5%		10%		15%		20%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UMEAN	0.8821	1.8635	0.8792	1.8585	0.8796	1.8612	0.8794	1.8657
IMEAN	0.7223	1.6313	0.7088	1.6082	0.7016	1.6004	0.7004	1.6026
UPCC	0.7579	1.5342	0.7144	1.4941	0.6319	1.4464	0.5927	1.4232
IPCC	0.7183	1.5145	0.7352	1.5076	0.6998	1.4782	0.6507	1.4473
UIPCC	0.7632	1.5360	0.6806	1.4442	0.6337	1.4047	0.6120	1.3864
LACF	0.6875	1.5257	0.6498	1.4145	0.6023	1.3757	0.5723	1.3674
SVD	0.5793	1.5099	0.5683	1.3947	0.5438	1.3726	0.5328	1.3576
LBR2	0.5539	1.4239	0.5376	1.3539	0.5184	1.3329	0.4938	1.3196
NLBR	**0.5520**	**1.3953**	**0.5254**	**1.3376**	**0.5073**	**1.3245**	**0.4857**	**1.3067**

From Table 1, we find that our approach NLBR obtains smaller MAE and RMSE values than others, which means higher prediction accurate in all cases. Meanwhile, as matrix density increases MAE and RMSE values become smaller, which indicates that more information to improve prediction accuracy. Our proposed approaches, which incorporating network location information of users in MF Model can greatly improve the prediction performance.

7.4. Impact of K

In our proposed approach, the Top-K value determines the size of the similar neighbors. To study the impact of the Top-K values on the prediction performance, we set dimensionality $f = 100$, $\gamma = 0.01, \lambda = 0.001$, matrix density=10% and 15%, and vary the values of K from 0 to 200 with a step value of 40.

Figure 2 shows the impact of K on the MAE and RMSE. We find that the MAE and RMSE values decrease quickly (prediction accuracy increase) at first. But when K is larger than a threshold, the MAE and RMSE values increase again. This is because too small K will reduce similar users' contribution to missing QoS values predictions, and too large K value will introduce noise (dissimilar users), which both cases will potentially hurt the prediction accuracy.

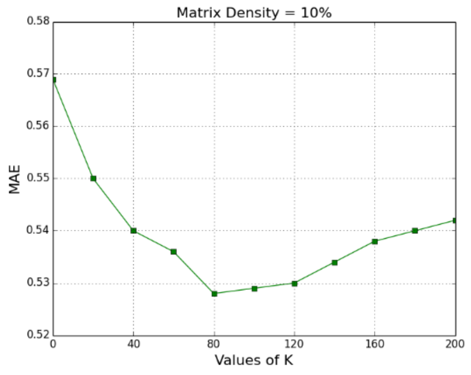
We can also observe that no matter what the matrix density is, K around 80 contributes to the smallest MAE values, which means K meets a threshold in this dataset. At the same time, K around 60 contribute to the smallest RMSE values. The optimal thresholds of MAE and RMSE are different because they are focusing on different aspects. This indicates that we should choose the optimal K by a lot of experiments.

7.5. Impact of γ

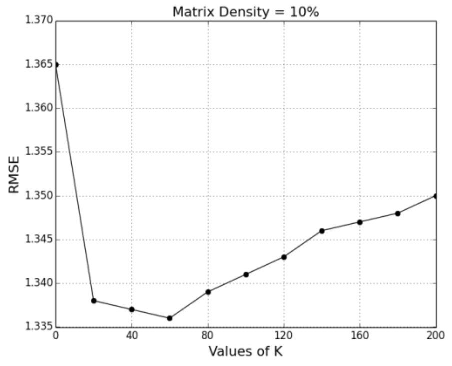
In our proposed approach, the parameter γ is the regularization parameter, which controls how much the regularization terms influence to the objective function.

If we set γ is too large, the network location information dominates the prediction process, which would potentially harm prediction performance. However, if we set γ is too small, we only focus on the general MF model and ignore the importance of network location-based regularization term. In this section, we analyze how the value of γ can affect the prediction accuracy, we set dimensionality $f = 100$, $\lambda = 0.001$ matrix density=10% and 15%, and vary the values of K from 60 to 80 with a step value of 20.

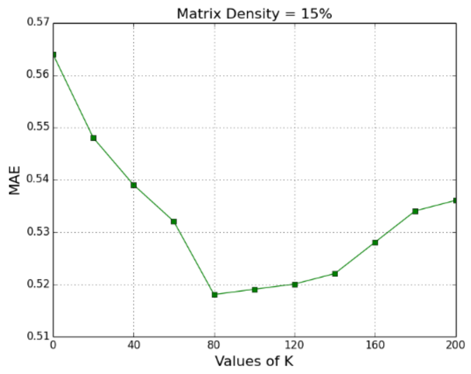
Observing from Figure 3, we draw the conclusion that the value of γ impacts the prediction accuracy significantly, and a suitable γ value will provide better prediction result. We also observe that γ around 0.01 contribute to the smallest MAE and RMSE values. The experiment also prove our point of view that combining the matrix factorization and network location can contribute to better prediction accuracy.



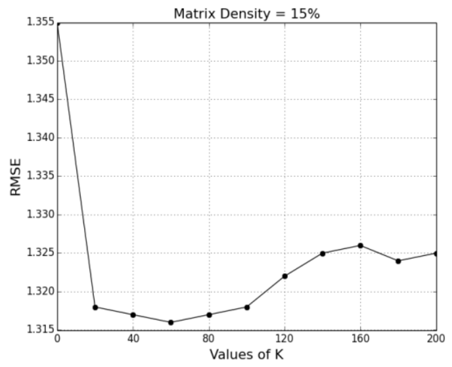
(a)



(b)



(c)



(d)

Figure 2. Impact of K

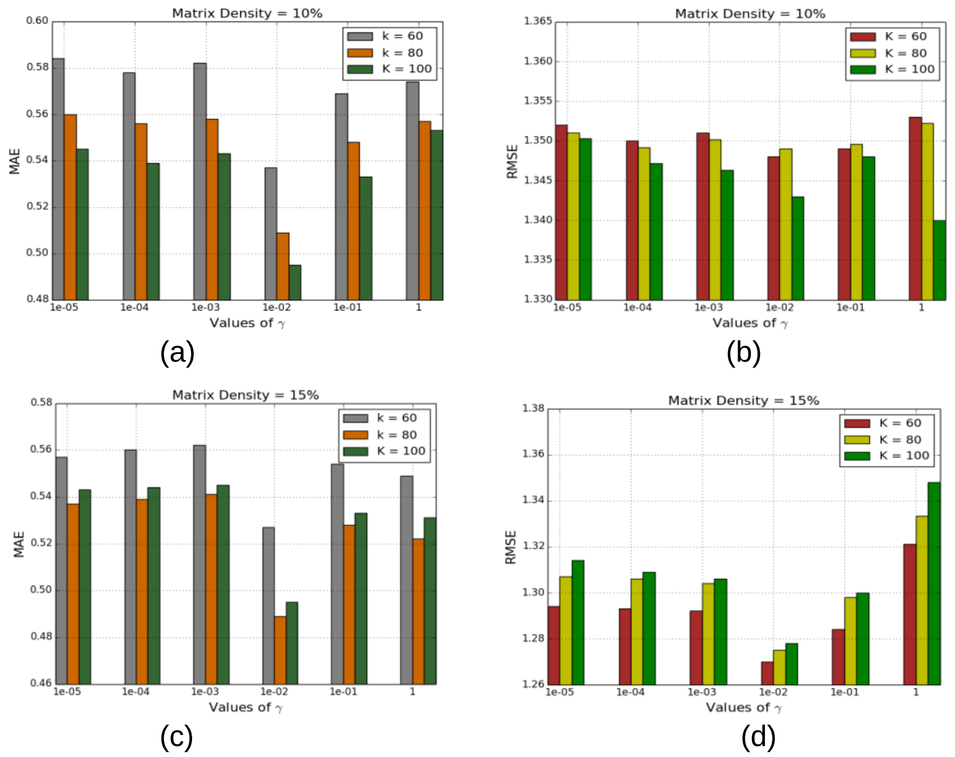


Figure 3. Impact of γ

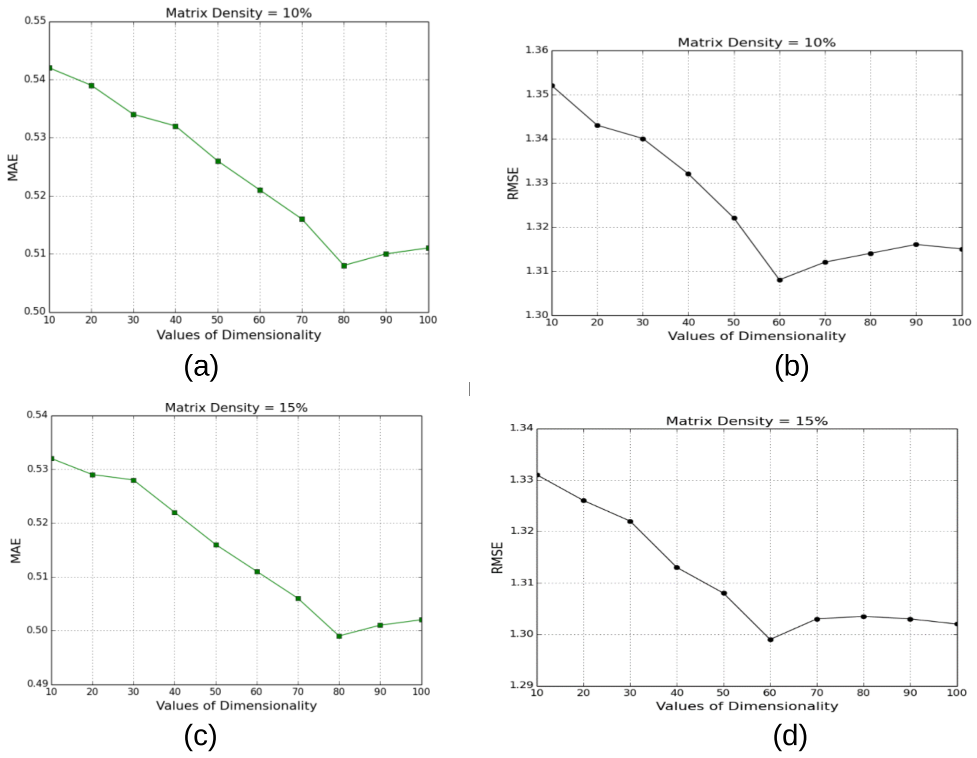


Figure 4. Impact of dimensionality

7.6. Impact of Dimensionality

In our proposed method, the dimensionality f determines how many latent factors of user and service apply to matrix factorization. To study the impact of the dimensionality on the MAE and RMSE, we set λ =0.01, γ =0.01, K =80, matrix density=10% and 15%. We vary the value of f from 10 to 100 with a step value of 10.

In Figure 4, we can find that MAE and RMSE reduce rapidly at first. However, when MAE going beyond 80 or RMSE going beyond 60, factors could barely improve performance, while slowing running time. This phenomenon is caused by over-fitting with the increase of dimensionality f , which reduces the prediction performance. In order to balance the prediction accuracy and running time, we should choose the best value. The optimal values of MAE and RMSE are different since they are different evaluative criteria focusing on different aspects.

7.7. Impact of Matrix Density

To study the impact of the matrix density on the MAE and RMSE, we set λ =0.01, γ =0.01, K =80, $f = 80$ and vary the value of matrix density from 2 to 20 with a step value of 2.

In Figure 5, as the density of the training matrix varies from 2 to 10, both MAE and RMSE reduce rapidly, which means the prediction accuracy is greatly improve. Then with the further increase of matrix density, both MAE and RMSE begin to reduce slowly. This indicates that with the training set contains more entries; our proposed approaches get higher prediction accuracy.

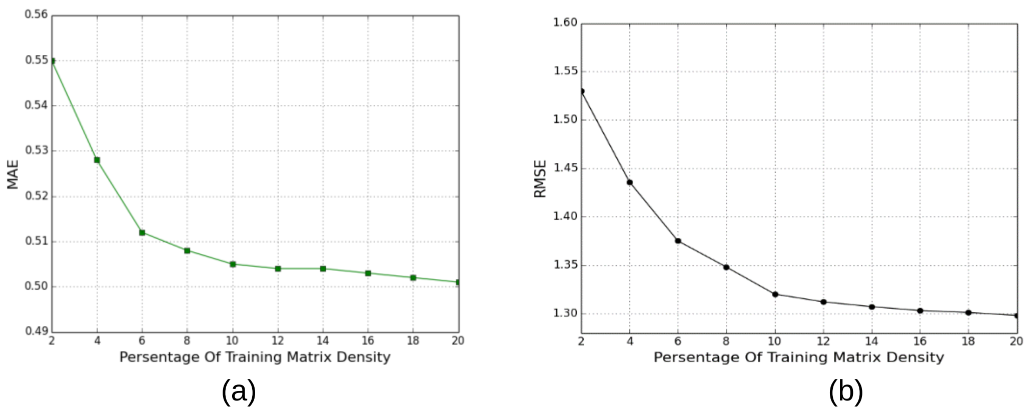


Figure 5. Impact of matrix density

8. Related Work

Predicting the missing values of QoS is the key to solve the service recommendation. At present, methods of web recommendation mainly include mean algorithm and collaborative filtering, and the latter is widely used in the modern recommender systems.

The collaborative filtering method is first proposed by Rich [14], Its basic idea is to use the history records to find the relationship between the users and services. Generally speaking, collaborative filtering method can be divided into two types: memory-based and model-based. Model-based approaches use history records to build a prediction model on latent user or item factors. The classical examples of model-based collaborative filtering method include cluster models [12], aspect models [13], and latent factor model [16]. Examples of memory-based collaborative filtering method include user-based methods [12], item-based methods [13], and their combination [6]. User-based methods predict the missing values of an active user based on values of a set of similar users to the active user. Similarly, item-based methods predict the missing values of an active user based on values of a set of similar items to the target item. User-based and item-based methods often use Pearson Correlation Coefficient (PCC) or Vector Space Similarity (VSS) to compute similarity. Finally, the third method combines the user-based and item-based results together and set a parameter to balance the value of each other.

However, there is limited work has been done to predict the missing values of QoS. One of the most important reasons is lack of real-world Web service dataset for experimental research. Recently, Zheng, *et al.,* [11] has released a large-scale real-world QoS dataset for research use. Shao, *et al.,* [7] propose a user-based collaborative filtering approach to predict the missing values of QoS from consumers' experiences. Zheng, *et al.*, [6] propose a hybrid method called WSRec that combines user-based and item-based method to predict the QoS vales and set a parameter to balance the value of each other. Chen, *et al.,* [17]discover the influence of users' location to the prediction accuracy and propose a region-based hybrid collaborative filtering method to predict the missing values of QoS. Wei Lo, *et al.,* [9] propose an approach that take advantage of the local connectivity between Web services users to identify the neighborhood, then combine these regularization terms in classic Matrix Factorization to build two models. Yao, *et al.,* [17] propose a hybrid approach that dynamically recommends Web Services that fit users' interests, which combines collaborative filtering and content-based recommendation. However, the above approaches fail to consider the influence of network location information of users to the accuracy of prediction, and capture the latent factors of users and services.

9. Conclusion and Future Work

In this paper, we proposed a novel approach based on network location to predict the unknown QoS values. Different from previous methods, we consider the location information of the users, and revamp the Matrix Factorization model with a novel network location-based regularization term. Through the experiment on large-scale real Web service data sets, it proves our method has greatly improved the efficiency and prediction accuracy of the prediction and it is better than the existing prediction methods.

In this paper, we only consider the network location information of users. In fact, the Web services located in the same network may provide similar factors. Therefore, some experiments need to be conducted on network locations of both users and Web services to improve the prediction performance in our future work.

Acknowledgement

This paper is granted by National Natural Science Foundation of China under Grant(No. 61100043), Zhejiang Provincial Natural Science Foundation(No.LY12F02003), China Postdoctoral Science Foundation(No. No.2013M540492) and The National Key Technology R&D Program under Grant(No.2012BAH24B04).

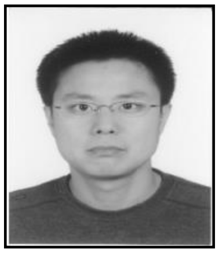
References

[1] L. J. Zhang, J. Zhang and H. Cai, "Services computing", Springer, **(2007)**.
[2] J. El. Hadad, M. Manouvrier and M. Rukoz, "TQoS: Transactional and QoS-aware selection algorithm for automatic Web service composition", IEEE Transactions on Services Computing, vol. 3, no. 1, **(2010)**, pp. 73-85.
[3] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition", Proceedings of the 18th international conference on World wide web, ACM, **(2009)**, pp. 881-90.
[4] H. Sun, Z. Zheng, J. Chen and M. R. Lyu, "NRCF: A Novel Collaborative Filtering Method for Service Recommendation", IEEE International Conference on Web Services, IEEE, **(2011)**, pp. 702-3.
[5] Z. Zheng and M. R. Lyu, "An adaptive QoS-aware fault tolerance strategy for web services", Empirical Software Engineering, vol. 15, no. 4, **(2010)**, pp. 323-45.
[6] Z. Zheng, H. Ma, MR. Lyu and I. King, "Wsrec: A collaborative filtering based web service recommender system", IEEE International Conference on Web Services, IEEE, **(2009)**, pp. 437-44.
[7] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie and H. Mei, "Personalized qos prediction forweb services via collaborative filtering", IEEE International Conference on Web Services, IEEE, **(2007)**, pp. 439-46.
[8] Y. Koren, "Collaborative filtering with temporal dynamics", Communications of the ACM, vol. 53, no. 4, **(2010)**, pp. 89-97.
[9] W. Lo, J. Yin, S. Deng, Y. Li and Z. Wu, "Collaborative Web Service QoS Prediction with Location-Based Regularization", IEEE 19th International Conference on Web Services, IEEE, **(2012)**, pp. 464-71.
[10] M. Tang, Y. Jiang, J. Liu and X. Liu, "Location-aware collaborative filtering for qos-based service recommendation", IEEE 19th International Conference on Web Services, IEEE, **(2012)**, pp. 202-9.
[11] Z. Zheng, Y. Zhang and M. R. Lyu, "Distributed qos evaluation for real-world web services", IEEE International Conference on Web Services, IEEE, **(2010)**, pp. 83-90.
[12] J. S. Breese, D. Heckerman and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering", Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence, Morgan Kaufmann Publishers Inc., **(1998)**, pp. 43-52.
[13] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl, "GroupLens: an open architecture for collaborative filtering of netnews", Proceedings of the ACM conference on Computer supported cooperative work, ACM, **(1994)**, pp. 175-86.
[14] E. Rich, "User modeling via stereotypes", Cognitive science, vol. 3, no. 4, **(1979)**, pp. 329-54.
[15] L. Si and R. Jin, "Flexible mixture model for collaborative filtering", Proceedings of the Twentieth International Conference on Machine Learning, ICML, **(2003)**, pp. 704-11.
[16] T. Hofmann, "Latent semantic models for collaborative filtering", ACM Transactions on Information Systems (TOIS), vol. 22, no. 1, **(2004)**, pp. 89-115.
[17] X. Chen, X. Liu, Z. Huang and H. Sun, "Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation", IEEE International Conference on Web Services, IEEE, **(2010)**, pp. 9-16.

Authors

Li Zhou, she is an associate professor of school of Computer Science and Technology, Hangzhou Dianzi University, China. She is with the Grid and Service Computing Lab in Hangzhou Dianzi University. Her current research areas include cloud computing, virtualization, *etc.*

Zhibo Song, he received the Bachelor Degree of Information and Computing in Henan University, Henan, China, in 2011. He is now studying the Master of Technology of Computer Application in Hangzhou Dianzi University, China. His research interest is Web Service Recommendation.



Yuyu Yin, he received the Doctors degree in computer science from Zhejiang University, Hangzhou, China, in 2010.He is currently an assistant professor at Hangzhou Dianzi University. His research interests include service computing, cloud computing and middleware techniques.

Zhai Sui-Chu, she received the master degree in computer science from Northeast Dianli University. Her research area is computer network security and its application in electric power system.