Location-based Hierarchical Matrix Factorization for Web Service Recommendation

Pinjia He*†, Jieming Zhu*†, Zibin Zheng*†‡, Jianlong Xu* and Michael R. Lyu*†
*Shenzhen Research Institute, The Chinese University of Hong Kong, China
†Computer Science and Engineering Department, The Chinese University of Hong Kong, China
‡State Key Lab for Novel Software Technology, Nanjing University, China
{pjhe, jmzhu, zbzheng, lyu}@cse.cuhk.edu.hk, jlxu@cuhkri.org.cn

***Abstract*—Web service recommendation is of great importance when users face a large number of functionally-equivalent candidate services. To recommend Web services that best fit a user's need, QoS values which characterize the non-functional properties of those candidate services are in demand. But in reality, the QoS information of Web service is not easy to obtain, because only limited historical invocation records exist. To tackle this challenge, in recent literature, a number of QoS prediction methods are proposed, but they still demonstrate disadvantages on prediction accuracy. In this paper, we design a location-based hierarchical matrix factorization (HMF) method to perform personalized QoS prediction, whereby effective service recommendation can be made. We cluster users and services into several user-service groups based on their location information, each of which contains a small set of users and services. To better characterize the QoS data, our HMF model is trained in a hierarchical way by using the global QoS matrix as well as several location-based local QoS matrices generated from user-service clusters. Then the missing QoS values can be predicted by compactly combining the results from local matrix factorization and global matrix factorization. Comprehensive experiments are conducted on a real-world Web service QoS dataset with 1,974,675 real Web service invocation records. The experimental results show that our HMF method achieves higher prediction accuracy than the state-of-the-art methods.**

***Keywords*—Web service; QoS prediction; clustering; location**

I. Introduction

Nowadays, more and more Web services designed by different organizations emerge on the Internet, providing a variety of functionalities. These Web services are widely employed as components in complex distributed systems, which greatly reduce software development time. But during the developing process, the developer will often find out a lot of functionally-equivalent Web services due to the fact that the number of Web services is experiencing a rapid growth. Consequently, Web service recommendation [1], [2] is recognized as an effective solution to assist developers in determining the most suitable candidate services.

To facilitate effective Web service recommendation, the quality of candidate services needs to be assessed from non-functional properties. Quality-of-Service (QoS) is a group of attributes (e.g., response time, throughput, reputation, etc.) that are usually employed to characterize the non-functional properties of Web services [3], [4], [5]. In principle, during Web service recommendation, services with similar functionalities are compared with each other automatically based on their QoS properties. Then the most suitable Web services in terms of user-defined QoS requirements can be recommended to the user. As a result, the user can just select from a small list of Web services returned by the recommendation system instead of struggling in all candidate services.

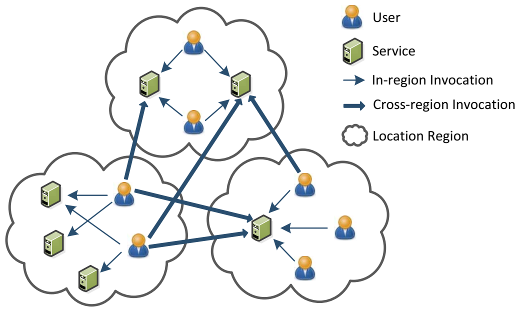


Fig. 1. Web Services Invocation Scenario

However, in practice, it is not easy to obtain the QoS values of all the candidate services, due to the following three reasons: 1) The QoS values need to be assessed from point of view of users, because different users may perceive different QoS values. 2) Only a limited number of service invocation records exist, since each user usually just invokes a handful of Web services. 3) It is time-consuming and resource-consuming to assess all the QoS values by invoking candidate services one by one, due to the large number of users and services. The QoS values of Web services observed by different users can be represented as a user-service matrix, whose rows represent users, columns represent services and entries are observed QoS values. But there are many missing values in the user-service matrix. To address this problem, QoS prediction is proposed to get approximated QoS values for those missing values in the user-service matrix.

Matrix factorization (MF) is a model-based collaborative filtering method, which has been used to make QoS prediction and widely studied in recent years. As other model-based collaborative filtering methods do, matrix factorization trains a model according to historical invocation records and uses patents found to predict QoS values for the missing values in the user-item matrix. In matrix factorization, we suppose that user-perceived QoS values are determined by a few latent features. These latent features (e.g. network

978-1-4799-5054-6/14 $31.00 © 2014 IEEE
DOI 10.1109/ICWS.2014.51

bandwidth, I/O operations, firewalls) not only affect users, but also have impact on the service side. Thus, the user-service matrix is approximated by two low-rank matrices, which represent the influence of latent features on users as well as services, respectively. Matrix factorization achieves good performance in traditional recommendation systems (e.g. movie recommendation systems) where entries in matrix are user-given ratings on different items (e.g. movies). Because each existing entity in that matrix is rated by a specific user, which is subjective, it can reflect the user's preference on an item. However, in Web service context, user-perceived QoS values of services are largely affected by physical factors. Location is such a key factor. For example, users and services in close locations tend to have small response time values. In Fig. 1, we have users and services which are in three location regions. Among all invocations, corresponding QoS values (e.g. response time values) of invocations in the same location region are more likely to have higher similarity (they tend to be small) compared with those cross-region ones.

To make fully use of the location information and improve the performance of our model, we propose a hierarchical matrix factorization method to predict QoS values for the unobserved user-service pairs. We firstly make use of clustering methods to separate users and services into several user-service groups according to their location. After that, QoS values of these users and services are represented as local user-service matrices. Different from global matrix which contains all users and services, local matrices only contain users and services in the same location region (same as cluster in this paper). A simple way to utilize location information is to only conduct matrix factorization on local matrices and simply put together their prediction results. However, it will degrade the prediction performance because we don't use any information of global matrix at all. To utilize both global and local information, our model performs matrix factorization on local matrices and global matrix sequentially in each approximation step. This model is run in a hierarchical way that in each approximation step, we linearly combine the predicted results of both global matrix and local matrices. Finally, based on the integrated predicted results, services with the best QoS values are recommended to corresponding users.

The main contributions of this paper are as follows:

- A hierarchical matrix factorization model is designed to effectively improve the prediction accuracy. Our model leverages location of users and services and takes global and local information into consideration at the same time.
- Extensive experiments are conducted on a real world Web service QoS dataset with 339 users and 5825 services. Experimental results prove that our method is better than other state-of-the-art methods.

The rest of this paper is organized as follows: Section II describes the framework of our service recommendation method. Our proposed hierarchical matrix factorization model is explained in detail in Section III. Section IV presents experiments and discusses the experiment results. Section V introduces some related work. Finally, we conclude our work and discuss some future improvement directions in Section VI.

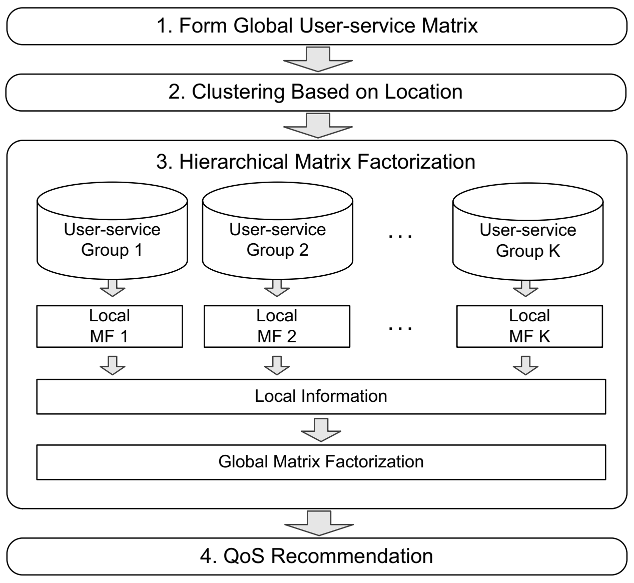
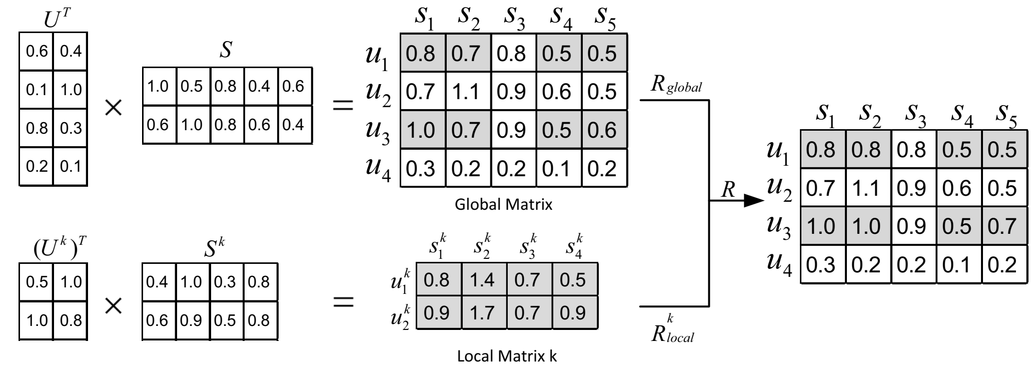


Fig. 2. Framework of Hierarchical Web Service Recommendation System

II. Framework of Web Service Recommendation

As we mentioned in Section. I, nowadays developers tend to utilize existing Web services provided by third parties to build complex distributed systems. An important issue for them is to choose the most suitable services among all functionally-equivalent ones without invoking all service candidates by themselves. Our recommendation system acts as a platform for these users to share their historical invocation records and obtain believable service recommendation according to their non-functional needs. The overview of our QoS-based hierarchical Web service recommendation system is shown in Fig. 2, which includes these main steps:

1) We collect and formalize existing information of users and services, including their IP address, longitude, latitude, invocation records shared by them, to name a few. Then all existing Web service invocation records will be used to form a global user-service matrix. Due to the fact that most of the users only called a few Web services before, the global matrix is very sparse.
2) In this step, we utilize longitude and latitude information to map users as well as services into a 2-dimensional space. Then we cluster all user nodes and service nodes into several user-service groups according to their coordinates in that space. In our method, each group contains both users and services, which coincides with the idea to make use of location of users and services at the same time. After clustering, users, services and their corresponding invocation records in each group can be used to form a local user-service matrix.
3) The above mentioned steps can be viewed as preprocessing steps, whose outcomes are one global matrix and several local matrices. Then our hierarchical matrix factorization model is used to predict missing values. In each approximation step, we firstly perform matrix factorization on all local matrices. After lo-



$$U^T = \begin{bmatrix} 0.6 & 0.4 \\ 0.1 & 1.0 \\ 0.8 & 0.3 \\ 0.2 & 0.1 \end{bmatrix} \quad S = \begin{bmatrix} 1.0 & 0.5 & 0.8 & 0.4 & 0.6 \\ 0.6 & 1.0 & 0.8 & 0.6 & 0.4 \end{bmatrix}$$

$$\begin{array}{c|ccccc} & s_1 & s_2 & s_3 & s_4 & s_5 \\ \hline u_1 & 0.8 & 0.7 & 0.8 & 0.5 & 0.5 \\ u_2 & 0.7 & 1.1 & 0.9 & 0.6 & 0.5 \\ u_3 & 1.0 & 0.7 & 0.9 & 0.5 & 0.6 \\ u_4 & 0.3 & 0.2 & 0.2 & 0.1 & 0.2 \end{array}$$

$$(U^k)^T = \begin{bmatrix} 0.5 & 1.0 \\ 1.0 & 0.8 \end{bmatrix} \quad S^k = \begin{bmatrix} 0.4 & 1.0 & 0.3 & 0.8 \\ 0.6 & 0.9 & 0.5 & 0.8 \end{bmatrix}$$

$$\begin{array}{c|cccc} & s^k_1 & s^k_2 & s^k_3 & s^k_4 \\ \hline u^k_1 & 0.8 & 1.4 & 0.7 & 0.5 \\ u^k_2 & 0.9 & 1.7 & 0.7 & 0.9 \end{array}$$

$$\begin{array}{c|ccccc} & s_1 & s_2 & s_3 & s_4 & s_5 \\ \hline u_1 & 0.8 & 0.8 & 0.8 & 0.5 & 0.5 \\ u_2 & 0.7 & 1.1 & 0.9 & 0.6 & 0.5 \\ u_3 & 1.0 & 1.0 & 0.9 & 0.5 & 0.7 \\ u_4 & 0.3 & 0.2 & 0.2 & 0.1 & 0.2 \end{array}$$

Fig. 3. An Example of QoS Prediction by Hierarchical Matrix Factorization ($\alpha = 0.8$)

cal matrix factorization, we obtain approximate QoS values for local matrices, which are referred to local information in Fig. 2. Then we calculate QoS values in global matrix. Instead of using only the product of user feature vectors and service feature vectors like traditional matrix factorization methods, for QoS values between users and services in user-service groups, we linearly combine predicted result of global matrix factorization and local matrix factorization to obtain the final prediction. User feature vectors and service feature vectors are columns of low-rank matrices used to perform QoS prediction during matrix factorization.

4) Now, all unobserved entries in the global matrix are predicted by our hierarchical matrix factorization model. Thus, our system is ready to recommend the services with the most suitable non-functional properties to all users in our system based on predicted results. For example, if an existing user, who is the i-th one in our global matrix, wants to find out a Web service with least response time among functionally-equivalent candidates. We just need to take out the i-th row from the global matrix, which can be regarded as a 1-dimensional vector. The service with the smallest numeric value in this vector will be recommended to that user.

III. Hierarchical Matrix Factorization

A. Overview

Our hierarchical matrix factorization consists of two main steps, which are clustering step and prediction step (modeling step). In clustering step (Subsection. III-B), users and services are clustered into different user-service groups according to their location information. In prediction step, our model is trained hierarchically on historical invocation records. There are two procedures in each training iteration, which are local matrix factorization (Subsection. III-C) and global matrix factorization (Subsection. III-D). Local matrix factorization will be done first and the result of it is used by global matrix factorization in each iteration.

To explain the "hierarchical" concept clearly, a straightaway example is given in Fig. 3. In this example, we assume there is one local matrix for simplicity, which means one user-service group is found by k-means. The calculation of Local Matrix k ($R^k_{local} = (U^k)^T S^k$) is local matrix factorization, while the remaining part of this figure is global matrix factorization. In global matrix factorization, the product of two low-rank matrices U^T and S is calculated at first, which is Global Matrix in Fig. 3. Then we combine Global Matrix and Local Matrix k to obtain the hierarchical prediction in this iteration, which is the rightmost matrix in this figure. u^k_1, u^k_2 in Local Matrix k and u_1, u_3 in Global Matrix are actually the same users, respectively. s^k_1,s^k_2,s^k_3 and s^k_4 are corresponding services in Local Matrix k to s_1, s_2, s_4 and s_5 in Global Matrix. Blocks in matrix containing QoS values between users and services which are both in the user-service group are marked as grey background, while the remaining ones are white. In the following, we will focus on block (u_3, s_5) in the rightmost matrix. To predict the QoS value $R(3,5)$ inside, we calculate in this way:

$$R(3,5) = 0.8 \times R_{global}(3,5) + (1 - 0.8) \times R^k_{local}(2,4) \quad (1)$$

where $R_{global}(3,5)$ is the QoS value of (u_3, s_5) in Global Matrix and $R^k_{local}(2,4)$ is the QoS value of (u^k_2, s^k_4) in Local Matrix k. The impact of parameter α will be discussed in Section. IV-D.

B. Users and Services Clustering

Location is used in our hierarchical matrix factorization method to improve prediction accuracy because of these following reasons: (1) Location information, which is represented by longitude and latitude in this paper, is an attribute that owned by every user and service. (2) Longitude and latitude of all users as well as services can be crawled on the Internet. (3) In Web service context, location does carry valuable information because geographically-close nodes tend to share similar network infrastructure, which to an extent affects QoS values such as response time. Although users and services located in close places may employ different network configurations, which also affect QoS values to varying degrees, it has been observed that this distinction has much less influence than the location information [6]. Because matrix factorization performs better on matrix with smaller variance, users and services are clustered into some user-services groups according to their longitude and latitude, which form local matrices in our method.

Since longitude and latitude information is selected to cluster user nodes and service nodes, now the problem is how to cluster 2-dimensional points into different groups. We chose k-means to cluster nodes in this problem because it is fast and tends to form globular clusters, which fits our locational similarity concept well. One big problem of k-means algorithm is that the clustering effect is closely related to the choice of initial mean points. However, in this problem, since user nodes as well as service nodes are all 2-dimensional, it is convenient for us to visualize all the points and predefine suitable initial mean points. After clustering, we may find out some clusters in which the number of users or services is very limited (close to or even less than the number of latent features we set in our model). This phenomenon often exists because of the geographical maldistribution of users or services in real-world datasets. Users and services belong to those clusters are defined as outliers. For these outliers, we do not consider them in local matrix factorization, but they will be used in the global matrix factorization step. Because too few users or services do not carry enough information to train the local matrix factorization model well, which leads to inaccurate prediction results.

C. Local Matrix Factorization

We employ traditional matrix factorization to perform prediction for all local user-service matrices. In local matrix factorization, each local user-service matrix is predicted by two low-rank matrices U^k and S^k, whose sizes are $d \times m_k$ and $d \times n_k$, where m_k is the number of users, n_k is the number of services and d is the number of latent features in our model. For matrix U^k or S^k, columns represent how much corresponding latent features will affect QoS values on user side or service side. Missing QoS values in local user-service matrix k are predicted by minimizing the following formula:

$$\mathcal{L}_k = \frac{1}{2}\sum_{i=1}^{m_k}\sum_{j=1}^{n_k} I_{ij}^k (R_{ij}^k - (U_i^k)^T S_j^k)^2 + \frac{\lambda_u^k}{2}\|U^k\|_F^2 + \frac{\lambda_s^k}{2}\|S^k\|_F^2 \tag{2}$$

where I_{ij}^k indicates whether QoS value on service j observed by user i in local matrix k is missing. If it is missing, I_{ij}^k will be 0, otherwise, its value is 1. R_{ij}^k means the available QoS value that user i experienced on service j in local matrix k. The rest two terms are regularization terms that help us get rid of overfitting issues.

To get a local minimum of the objective function in Equ. 2, we apply the gradient descent algorithm on both U_i^k and S_j^k:

$$(U_i^k)' \leftarrow U_i^k - \eta_u^k \frac{\partial \mathcal{L}_k}{\partial U_i^k} \tag{3}$$

$$(S_j^k)' \leftarrow S_j^k - \eta_s^k \frac{\partial \mathcal{L}_k}{\partial S_j^k} \tag{4}$$

where $\frac{\partial \mathcal{L}_k}{\partial U_i^k}$ and $\frac{\partial \mathcal{L}_k}{\partial S_j^k}$ are calculated by:

$$\frac{\partial \mathcal{L}_k}{\partial U_i^k} = \sum_{j=1}^{n_k} I_{ij}^k ((U_i^k)^T S_j^k - R_{ij}^k)(S_j^k) + \lambda_u^k U_i^k \tag{5}$$

$$\frac{\partial \mathcal{L}_k}{\partial S_j^k} = \sum_{i=1}^{m_k} I_{ij}^k ((U_i^k)^T S_j^k - R_{ij}^k)((U_i^k)^T) + \lambda_s^k S_j^k \tag{6}$$

We set $\lambda_u^k = \lambda_s^k$ in all experiments to reduce the complexity of our model.

D. Global Matrix Factorization

As mentioned in Section I, matrix factorization predicts all missing values by minimizing the error between prediction results and historical QoS records. In traditional matrix factorization, location information is not taken into consideration, which to some extent degrades the performance of the model. But we observed that the global user-service matrix can actually be regarded as several user-service groups, where users and services are located in similar places, and remainders which are geographically far apart from those groups. Thus, we cluster users and services into user-service groups according to longitude and latitude, each of which contains QoS values with small variance that benefits the performance of matrix factorization. An extreme idea to utilize knowledge given by local matrices is to only apply matrix factorization model on those user-service groups independently, which will make fully use of local information but lead to the loss of global structure. Thus, to consider both global and local information, it is natural to predict QoS values by the linear combination of matrix factorization on global matrix as well as local matrices. Hence, the following term is designed:

$$\alpha U_i^T S_j + (1-\alpha)\hat{R}_{ij}^k \tag{7}$$

where U_i^T and S_j represent global user feature vector and global service feature vector respectively. $\hat{R}_{ij}^k$ is calculated by $(U_i^k)^T S_j^k$, which are the corresponding local ones. Notice that although U_i and U_i^k are related to the same user, the value of i is actually different because the numbers of users in these two matrices are not equal. We both use i here just for simplicity and clarity. It is also the case for S_j and S_j^k. This term integrates the approximate values given by global vectors and local vectors, which coincides with the idea to utilize both global structure and local information at the same time. α is a tunable parameter that indicates how much global information we use in our hierarchical model. The value of α is related to the dataset we use. This term will be integrated into traditional matrix factorization to obtain our hierarchical model:

$$\mathcal{L} = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - (\alpha U_i^T S_j + (1-\alpha)\hat{R}_{ij}^k))^2 + \frac{\lambda_u}{2}\|U\|_F^2 + \frac{\lambda_s}{2}\|S\|_F^2 \tag{8}$$

Since some users and services do not belong to any user-service groups, for those corresponding missing QoS values, we only use global matrix factorization to perform prediction. To formalize this concept, α is selected by:

$$\alpha = \begin{cases} 1 & \text{if } u_i \text{ or } s_j \text{ or both are not in any local groups} \\ \alpha_k & \text{if } u_i \text{ and } s_j \text{ are both in local group } k \end{cases} \tag{9}$$

Similar to local matrix factorization, we apply gradient descent to approximate. User feature vectors and service feature vectors are updated as following:

$$U_i' \leftarrow U_i - \eta_u \frac{\partial \mathcal{L}}{\partial U_i} \tag{10}$$

$$S_j' \leftarrow S_j - \eta_s \frac{\partial \mathcal{L}}{\partial S_j} \tag{11}$$

where $\frac{\partial \mathcal{L}}{\partial U_i}$ and $\frac{\partial \mathcal{L}}{\partial S_j}$ are calculated by:

$$\frac{\partial \mathcal{L}}{\partial U_i} = \sum_{j=1}^{n} I_{ij}(\alpha U_i^T S_j + (1-\alpha)\hat{R}_{ij}^k - R_{ij})(\alpha S_j) + \lambda_u U_i \quad (12)$$

$$\frac{\partial \mathcal{L}}{\partial S_j} = \sum_{i=1}^{m} I_{ij}(\alpha U_i^T S_j + (1-\alpha)\hat{R}_{ij}^k - R_{ij})(\alpha U_i^T) + \lambda_s S_j \quad (13)$$

In all experiments, we set $\lambda_u = \lambda_s$ for simplicity.

IV. Experiments

A. Dataset Description

A real-world dataset consisting of 339 users and 5,825 services is used in all our experiments. This dataset contains much useful information about users and services, including IP address of each user, WSDL address of each service, users' longitude as well as latitude and so on. This dataset was introduced in detail in a related paper [7]. However, longitude and latitude information of services was not published. Thus we obtain their location information by an iplocation service [1].

B. Metrics

We use Mean Absolute Error (MAE) and Normalized Mean Absolute Error (NMAE) to measure the prediction accuracy of our proposed model. The definition of MAE is:

$$MAE = \frac{\sum_{ij} |R_{ij} - \hat{R}_{ij}|}{N} \quad (14)$$

where R_{ij} represents the observed QoS value between user i and service j, while $\hat{R}_{ij}$ denotes the QoS value predicted by our hierarchical matrix factorization model between the corresponding user-service pair. N is the number of missing QoS values in the user-service matrix. Different from MAE that calculate the absolute average error, NMAE is the standard MAE normalized by the mean of expected QoS values [1]:

$$NMAE = \frac{MAE}{\sum_{ij} R_{ij}/N} \quad (15)$$

C. Comparison

To prove the effectiveness of our hierarchical matrix factorization method, we ran extensive experiments on state-of-the-art QoS prediction methods and compare our method with them. Here is a brief introduction about those popular methods:

- **UMEAN:** UMEAN uses the mean of QoS values perceived by a user on all services he/she called.
- **IMEAN:** In this method, we predict a QoS value by calculating the mean of all existing historical records on that service observed by different users.
- **UPCC:** This approach [8] utilize historical invocation records of similar users to perform prediction.

[1]http://www.iplocation.net/

TABLE I. Parameters

Parameter	Value
λ_u, λ_s	35
λ_u^1, λ_s^1	10
λ_u^2, λ_s^2	20
Dimensionality	10

TABLE II. Value of α

Density	0.15	0.20	0.25	0.30
α	0.8	0.8	0.8	0.9

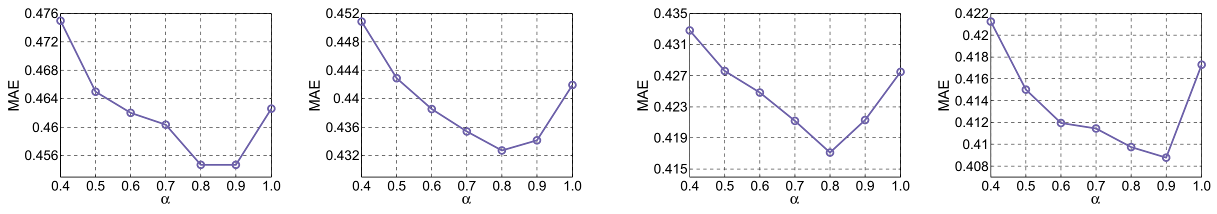
- **IPCC:** The overall idea of this approach is the same with UPCC, but instead of making use of similar users, it pays attention to digging out some services alike. Then the QoS values of similar services observed by the specific user are used in prediction step.
- **WSRec:** This method [1] is the hybrid one that linearly combines UPCC and IPCC, which takes advantage of both similar users and similar services.
- **PMF:** This method is proposed by Mnih and Salakhutdinov [9]. The product of two low-rank matrices is used as the predicted user-service matrix.
- **LBR2:** Lo et al. [6] considered location of users and add a related regularization term to PMF model.

Since the user-service matrix is sparse in real-world cases, we randomly remove some historical records to make our experiments more realistic. Then we will have some user-service matrices with different densities. In our experiment, each QoS prediction method is run on 4 different matrices, whose densities are 15%, 20%, 25% and 30% respectively. A matrix with 20% density means that there are 20% available user-service invocation records for us to regard as training set, while the remaining 80% are ones waiting to be predicted. K-means algorithm helps us separate all the users and services into 5 user-service groups. After the check of the number of users and services in each group, 3 groups are deleted because there are too few users in them. As we mention in Section. III, the number of useful groups is significantly related to the distribution of user nodes and service nodes as well as the number of nodes in the dataset. The parameters we used in our experiment are listed in Table. I and Table. II. For the purpose of simplicity, we set $\alpha_1 = \alpha_2$ in all our experiments, and α is used to represent for these two.

Table. III shows us the MAE and NMAE of different methods on 4 matrices with density from 15% to 30%. The MAE and NMAE of all methods decrease as the matrix density become larger, which means more information of users and services will benefit the prediction performance. Besides, it is obvious that the MAE and NMAE of our method are consistently lower than others under all matrix density settings. That means our method outperforms others under all circumstances. Thus, performing matrix factorization hierarchically and making use of geographical information really help us improve QoS prediction model in prediction accuracy.

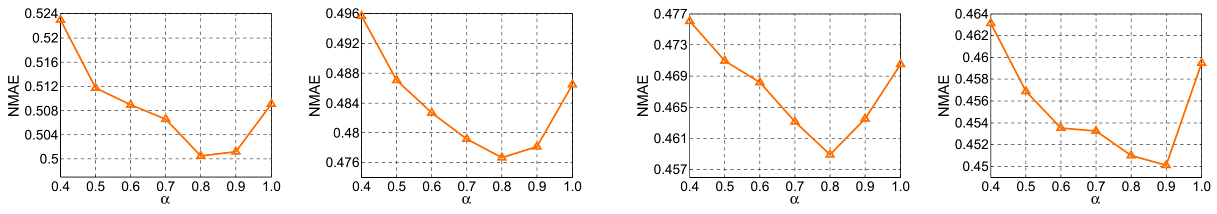
TABLE III. PERFORMANCE COMPARISON

Methods	Density = 15%		Density = 20%		Density = 25%		Density = 30%	
	MAE	NMAE	MAE	NMAE	MAE	NMAE	MAE	NMAE
UMEAN	0.8767	0.9650	0.8735	0.9608	0.8740	0.9604	0.8735	0.9599
IMEAN	0.6823	0.7512	0.6806	0.7489	0.6781	0.7453	0.6789	0.7461
UPCC	0.5196	0.5740	0.4911	0.5368	0.4715	0.5168	0.4574	0.5019
IPCC	0.5244	0.5753	0.4629	0.5079	0.4389	0.4814	0.4211	0.4681
WSRec	0.4999	0.5501	0.4530	0.4961	0.4310	0.4727	0.4147	0.4593
PMF	0.4626	0.5091	0.4420	0.4865	0.4275	0.4705	0.4173	0.4595
LBR2	0.4596	0.5060	0.4404	0.4846	0.4242	0.4667	0.4153	0.4574
HMF	**0.4547**	**0.5006**	**0.4327**	**0.4766**	**0.4171**	**0.4589**	**0.4088**	**0.4501**



(a) Density = 15% (b) Density = 20% (c) Density = 25% (d) Density = 30%

Fig. 4. Impact of α on MAE



(a) Density = 15% (b) Density = 20% (c) Density = 25% (d) Density = 30%

Fig. 5. Impact of α on NMAE

D. Impact of α

In our hierarchical matrix factorization model, parameter α controls how much local information we use in QoS prediction procedure. If α is set to be 1, no local information is taken into consideration. If α is 0, QoS values, whose corresponding users and services are in the same user-service group, are predicted by local matrix factorization independently without any information from global context. That is to say, we only use historical invocation records of geographically-close users and services to perform prediction. In a word, α is utilized to keep a good balance between global context and local information. To study the influence of α on our model and find an optimal one, we tune density from 15% to 30%, with a step size 5%.

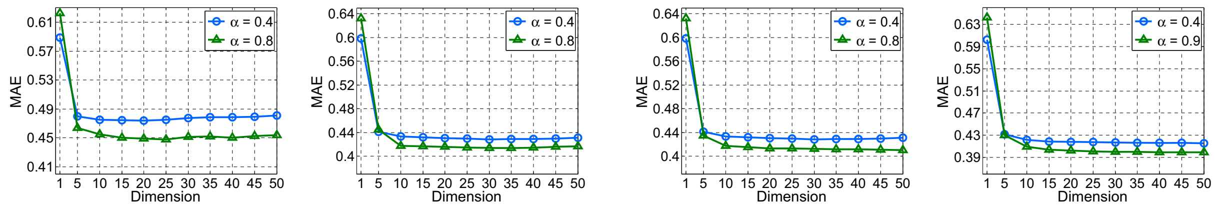
Fig. 4 and Fig. 5 show us, under 4 different matrix density settings, the change of MAE and NMAE as the value of α varies from 0.4 to 1.0. We can see that given matrix density 15%, 20% or 25%, both MAE and NMAE are the lowest when α is around 0.8. That means our hierarchical matrix factorization model performs the best when $\alpha = 0.8$ for density 15%, 20% or 25%, while for density 30%, $\alpha = 0.9$ is the most suitable choice.

In this paragraph, we will raise a discussion for the condition that matrix density is 15%. We observe from the figure that when α is 0.4, both MAE and NMAE are high. As the value of α changes from 0.4 to 0.8, MAE as well as NMAE drop down sharply at first but become smoothly as α gets close to the optimal value. That indicates too little global context usage harms the performance of our model. Adding the impact of global context will highly increase the prediction accuracy at the beginning, but the effect becomes small when the model is near a balance between local information and global context. We can also notice that when α is larger than 0.8, the MAE and NMAE get larger, which tells us that ignorance of local information will lead to the degradation of performance. When matrix density is 20%, 25% or 30%, the changing tendency and the reason of the change is similar to what we have just discussed.

E. Impact of Dimensionality

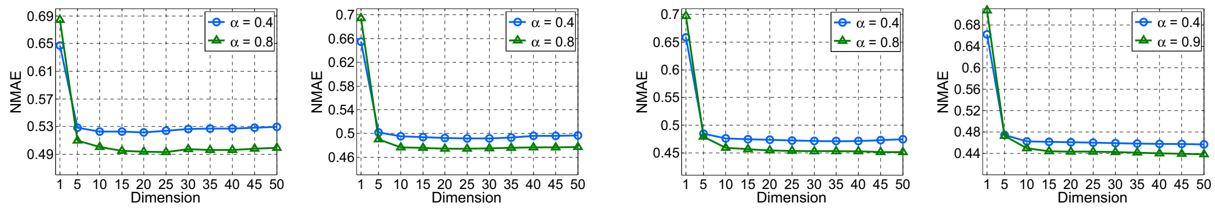
In our proposed method, dimensionality means the number of latent features that will affect the user-perceived QoS values on services. If this parameter is small, it indicates that only a few key latent features determine QoS value. If dimensionality is set to be a large number, it is assumed that there are many latent features contribute collectively to the final prediction result. To study the impact of dimensionality on our model we tune $\alpha = 0.4$ and $\alpha = 0.8$ for matrix density 15%, 20%, or 25%. When density is 30%, α was set to be 0.4 and 0.9.

Fig. 6 and Fig. 7 illustrate the impact of dimensionality



(a) Density = 15% (b) Density = 20% (c) Density = 25% (d) Density = 30%

Fig. 6. Impact of Dimensionality on MAE



(a) Density = 15% (b) Density = 20% (c) Density = 25% (d) Density = 30%

Fig. 7. Impact of Dimensionality on NMAE

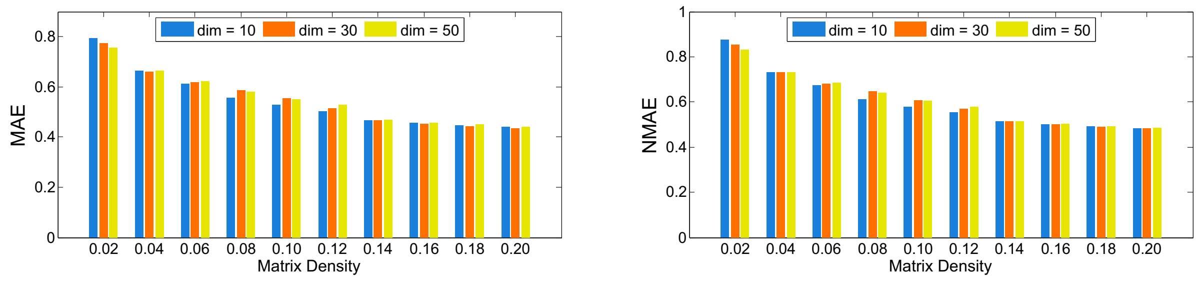


Fig. 8. Impact of Matrix Density

on MAE and NMAE of our model respectively. Both MAE and NMAE are high at first and decrease rapidly as dimension increase. It shows that only few latent features can not lead to a good prediction result, so we can effectively improve the performance by raising the dimensionality. However, the speed of decrease slows down as the dimensionality goes up. When dimensionality exceeds a threshold, MAE and NMAE even begin to increase little by little. These can be explained by following two reasons: (1) If dimensionality is larger than a threshold, our model comes across the overfitting problem, which will degrade the performance. (2) The number of users as well as services in our local user-service groups is smaller than that of the global matrix. When making use of local information, an oversize dimensionality will result in bad prediction performance. Thus the overall prediction accuracy will be affected negatively.

F. Impact of Matrix Density

In the problem context discussed in this paper, matrix density is the ratio of the number of observed user-service invocation records against the product of the number of users and services. It also indicates how much available information we have to help us make prediction. To study the effect of matrix density, we set $\alpha = 0.8$. Besides, we consider three different values of dimensionality, which are 10, 30 and 50.

Fig. 8 shows us the corresponding MAE and NMAE of our model from matrix density 2% to 20% with a step size 2%. The figure illustrates that as the matrix density goes up, the MAE and NMAE decrease rapidly at first. When matrix density becomes larger, the speed of decrease slows down. That means, when there are hardly any historical invocation records in the user-service matrix, the best way to improve recommendation performance is to motivate users to report more QoS values or try some services which have not been called before. But when the number of invocation records grows larger, it would be better to focus on improvement of prediction model instead.

V. Related Work

In recent years, service computing [10], [11], [12], [13], [14] has attracted more and more attention from industry communities as well as academic circles. Among all topics in service computing, QoS-aware service selection and service composition are studied in a large number of literatures [15], [16], [17], [18], whose goal is to decide which candidate services to be used as components in complex systems. However, most of the research work has a necessary precondition: QoS values of all candidate services for corresponding users are already known, which is always not satisfied in real-world cases. Thus, many researchers begin to focus on QoS prediction issues, which aims at analyzing existing user-services invocation records and then predict those unobserved ones.

Collaborative filtering was applied to this problem by Shao et al. [8] first. In their paper, a user-based collaborative filtering method was proposed, which makes prediction for a specific user based on similar users. Zheng et al. [1] designed a hybrid approach to leverage both user-based and item-based collaborative filtering. Chen et al. [3] built a region model before collaborative filtering step, their method can be tuned to trade off speed and recommendation accuracy. Tang et al. [4] raised an hierarchical method to predict QoS values on user side and service side separately. Different from previous methods that directly dig out neighbors on all historical records, they seek similar users in the same Autonomous System (AS) and country first. These collaborative filtering methods are classified as memory-based collaborative filtering techniques. Although these methods have a good performance when there are enough user-service invocation records, they do not perform well when the matrix becomes bigger and sparser.

Compared with memory-based collaborative filtering methods, model-based collaborative filtering methods can provide us with more precise prediction result. The overall idea of model based methods is to train a model according to existing data and use that trained model to predict missing QoS values. Probabilistic matrix factorization (i.e. matrix factorization in this paper) was proposed by Salakhutdinov et al. [9]. Lo et al. [5] raised an extended matrix factorization approach, whose main contribution is two novel relational regularization terms that can improve prediction accuracy. They also proposed a location-aware matrix factorization model [6]. In that work, they designed two user-location-aware matrix factorization models, each of which extended by a location regularization term. However, location information of services is neglected, which is actually helpful in improving prediction accuracy. In this paper, the model proposed utilizes geographical information of both users and services simultaneously. Besides, our model is run in a hierarchical way, which means it can make use of not only global context, but also local information.

VI. Conclusion and Future Work

This paper proposes a new model-based collaborative filtering method to predict missing QoS values in the user-service matrix and recommend the most suitable Web services to certain users. Geographically-close users and services are clustered to form groups, which makes up several small user-service matrices. A hierarchical matrix factorization model is designed to integrate global matrix factorization and local matrix factorization. Empirical analysis shows that our model outperforms the state-of-the-art methods.

In the future, we will continue to improve our model. Firstly, we can try out more sophisticated ways to combine global matrix factorization and local matrix factorization. Regarding how to cluster users and services, we will use other information in addition to location. We can also allow a user node or a service node belong to more than one clusters.

Acknowledgment

The work described in this paper was supported by the National Basic Research Program of China (973 Project No. 2011CB302603), the National Natural Science Foundation of China (Project No. 61332010 and Project No. 61100078), and the Research Grants Council of the Hong Kong Special Administrative Region, China (No. N_CUHK405/11 of the NSFC/RGC Joint Research Scheme, No. 415212 of the General Research Fund).

References

[1] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "WSRec: A collaborative filtering based web service recommender system," in *Proc. 16th Int'l Conf. Web Services (ICWS'09)*. IEEE, 2009, pp. 437–444.

[2] L. Yao, Q. Z. Sheng, A. Segev, and J. Yu, "Recommending web services via combining collaborative filtering with content-based features," in *Proc. 20th Int'l Conf. Web Services (ICWS'13)*. IEEE, 2013, pp. 42–49.

[3] X. Chen, Z. Zheng, X. Liu, Z. Huang, and H. Sun, "Personalized qos-aware web service recommendation and visualization," *IEEE Transactions on Services Computing*, vol. 6, no. 1, pp. 35–47, 2013.

[4] M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for QoS-based service recommendation," in *Proc. 19th Int'l Conf. Web Services (ICWS'12)*. IEEE, 2012, pp. 202–209.

[5] W. Lo, J. Yin, S. Deng, Y. Li, and Z. Wu, "An extended matrix factorization approach for QoS prediction in service selection," in *Proc. 9th Int'l Conf. Services Computing (SCC'12)*. IEEE, 2012, pp. 162–169.

[6] ——, "Collaborative web service QoS prediction with location-based regularization," in *Proc. 19th Int'l Conf. Web Services (ICWS'12)*. IEEE, 2012, pp. 464–471.

[7] Z. Zheng, Y. Zhang, and M. R. Lyu, "Distributed QoS evaluation for real-world web services," in *Proc. 17th Int'l Conf. Web Services (ICWS'10)*. IEEE, 2010, pp. 83–90.

[8] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized QoS prediction forweb services via collaborative filtering," in *Proc. 14th Int'l Conf. Web Services (ICWS'07)*. IEEE, 2007, pp. 439–446.

[9] A. Mnih and R. Salakhutdinov, "Probabilistic matrix factorization," in *Advances in neural information processing systems*, 2007, pp. 1257–1264.

[10] L.-J. Zhang, J. Zhang, and H. Cai, "Services computing," in *Springer and Tsinghua University Press*, 2007.

[11] Q. Wu, A. Iyengar, R. Subramanian, I. Rouvellou, I. Silva-Lepe, and T. Mikalsen, "Combining quality of service and social information for ranking services," in *Proc. 7th Int'l Conf. Service Oriented Computing (ICSOC'09)*. Springer, 2009, pp. 561–575.

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

[13] M. Silic, G. Delac, and S. Srbljic, "Prediction of atomic web services reliability based on k-means clustering," in *Proc. 9th Joint Meeting on Foundations of Software Engineering (ESEC/FSE'13)*. ACM, 2013, pp. 70–80.

[14] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, "Towards online, accurate, and scalable qos prediction for runtime service adaptation," in *Proc. 34th Int'l Conf. Distributed Computing Systems (ICDCS'14)*. IEEE, 2014.

[15] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition," in *Proc. 18th Int'l Conf. World Wide Web (WWW'09)*. ACM, 2009, pp. 881–890.

[16] 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, pp. 73–85, 2010.

[17] A. F. Huang, C.-W. Lan, and S. J. Yang, "An optimal QoS-based web service selection scheme," *Information Sciences*, vol. 179, no. 19, pp. 3309–3322, 2009.

[18] A. Kattepur, N. Georgantas, and V. Issarny, "QoS composition and analysis in reconfigurable web services choreographies," in *Proc. 20th Int'l Conf. Web Services (ICWS'13)*. IEEE, 2013, pp. 235–242.