Temporal QoS-Aware Web Service Recommendation via Non-negative Tensor Factorization

Wancai Zhang, Hailong Sun, Xudong Liu, Xiaohui Guo
School of Computer Science and Engineering, Beihang University
Beijing, China
{zhangwc, sunhl, liuxd, guoxh}@act.buaa.edu.cn

ABSTRACT

With the rapid growth of Web Service in the past decade, the issue of QoS-aware Web service recommendation is becoming more and more critical. Since the Web service QoS information collection work requires much time and effort, and is sometimes even impractical, the service QoS value is usually missing. There are some work to predict the missing QoS value using traditional collaborative filtering methods based on *user-service* static model. However, the QoS value is highly related to the invocation context (e.g., QoS value are various at different time). By considering the third dynamic context information, a Temporal QoS-Aware Web Service Recommendation Framework is presented to predict missing QoS value under various temporal context. Further, we formalize this problem as a generalized tensor factorization model and propose a Non-negative Tensor Factorization (NTF) algorithm which is able to deal with the triadic relations of *user-service-time* model. Extensive experiments are conducted based on our real-world Web service QoS dataset collected on Planet-Lab, which is comprised of service invocation response-time and throughput value from 343 users on 5,817 Web services at 32 time periods. The comprehensive experimental analysis shows that our approach achieves better prediction accuracy than other approaches.

Categories and Subject Descriptors

H.3.5 [**On-line Information Services**]: Web-based services; I.2.6 [**Artificial Intelligence**]: Parameter Learning

Keywords

Web Service Recommendation, QoS, Collaborative Filtering, Tensor Factorization

1. INTRODUCTION

As the developing of Service-oriented computing (SOC), Web Service has become the standard technology for sharing data and software, and Web service users can compose services to accomplish a more complex task. Web services provide the means for such seamless integration of business processes across organizational boundaries [3]. Quality-of-Service (QoS) encompasses important nonfunctional attributes such as performance metrics (e.g., response time),

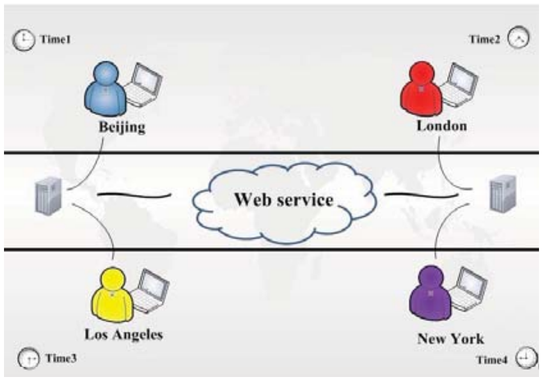


Figure 1: A motivating scenario

security attributes, transactional integrity, reliability, scalability, and availability [18]. Web services environment demands greater availability of applications and introduce complexity of accessing and managing services. In addition, QoS is widely employed in describing non-functional properties of Web Services for optimizing the Web service composition.

Since the number of functionally equivalent services offered on the web with different QoS properties is increasing, it is quite important to recommend services considering their non-functional QoS properties. Although the Web service providers may declare the QoS properties of services (e.g., availability, response-time, throughput, etc.), which are highly related to the Web service invocation context. Web service QoS value observed from the users is usually quite different from those value declared by the service providers, due to: (1) Web service performance is related to invocation time, because the network environment and service status (e.g., workload, number of clients ,etc.) are changing over time; (2) the user locations are distributed in different geographical locations which greatly influence the user-observed QoS value of Web service. For these reasons, to carry out a QoS-aware service recommendation system, predicting missing QoS value of service is often required. Nowadays there are some Collaborative Filtering (CF) Recommendation Frameworks to predict the service QoS value. But one significant limitation of most of the existing CF methods [24, 31, 30, 17, 6], is that they are static models in which relations are assumed to be fixed in different temporal sequence where the Web service QoS properties are varying.

Example. Figure 1 shows a scenario for finding the best QoS value of a Web service. The service users from different geographical locations submit their requests to the Web Service Recommender System at various time, specifying some criteria for service QoS parameters (e.g., responsiveness, availability, throughput.). The recommendation system then returns a list of Web services with best QoS value by taking comprehensive account of the users and services invocation context information. The recommended Web services can be exposed to service users as a Web service, API or widget.

In this scenario, we assume that some service users from Beijing, London, Los Angeles and New York invoke the Web service at different time (e.g., 8, 9, 10, 11 o'clock local time), respectively. Now the recommendation system needs to predict the Web service QoS value which will be decided by the time context information of service users. Through the QoS prediction, the Service-Oriented Architecture system designer can make more informed decisions on the Web service composition. As a result, a Temporal QoS-aware Web Service Recommendation Framework is presented to predict missing QoS value at different service invocation time. Though this example, the *user-service* intra-relations are often represented by two-dimensional matrix, which is used by the previous work [24, 31, 30, 17, 6]. We extend the two two-dimensional *user-service* matrix into a more complicated *user-service-time* triadic relations represented by three-dimensional tensor, and present a novel tensor factorization (TF) [13] that is based on a generalization of matrix factorization (MF) [15]. The key idea of our temporal three-dimensional TF approach is to replace the *user-service* matrix in MF with the *user-service-time* interaction relations by considering the difference of QoS value at difference time. However, the QoS value is non-negative, the conventional TF method dose not guarantee the result factors for non-negativity. To deal with the non-negativity issue, Non-negative Tensor Factorization (NTF), which is decomposed into a set of factor matrices, is an emerging research topic in recent years. Some NTF algorithms have been proposed in literatures [27], and we use multiplicative updating rules to approximate the QoS value tensor.

Contributions. In this paper, we clarify some intra-relations specific to personalized Web service QoS property based on temporal data analysis, and observe that the comprehensive knowledge of Web service QoS properties are difficult to acquire. The QoS information collection work requires much time and effort, and is sometimes even impractical (e.g., service invocations are various by different users at different time). A new temporal QoS-aware recommendation approach based on tensor factorization is proposed to address the issue of Web service QoS value prediction with considering Web service invocation time. This framework collects QoS information from geographically distributed service users, and filters out qualified value as training data for predicting missing QoS value. After a period of collection, a QoS value dataset is produced, which contains different context information types: user, service and time. The relations among these objects are complicated. For example, users with similar location may get different service invocation response-times at different invocation time. By performing analysis on the QoS information data, we propose a model-based CF method to discover the latent factors that govern the associations among these multi-type objects. Compared with prior related works, our main contributions can be summarized as follows.

1. There exist the complicated relations among user,service and temporal information in the real-world Web service QoS properties data. We discover the fact and utilize tensor to represent the triadic relations.
2. An Non-negative Tensor Factorization (NTF) approach is proposed to predict the Web service QoS value with considering service invocation time, and present a Temporal QoS-Aware Web Service Recommendation Framework.
3. We evaluate our approach experimentally using real-world Web service dataset, which contains more than 19 millions real-world Web service invocation results from 343 distributed service users on 5,817 real-world Web services at 32 time periods. Our real-world Web service dataset has been published online[1].

The rest of the paper is organized as follows. Section 2 discusses related work, while Section 3 introduces the tensor preliminaries and notations. Section 4 presents the problem statement and our NNCP approach. Section 5 describes our implementation and experiments. Finally, Section 6 concludes the paper.

2. RELATED WORK

Web Service QoS has been widely discussed for a long time. Zeng et al. [28] employ a five-dimensional Web Service QoS property model (i.e., execution price, execution duration, reputation, availability, and reliability) for dynamic Web service composition, and then they transfer QoS-aware service selection into an optimization problem [29]. Alrifai et al. [2] propose an efficient service composition approach by considering both generic QoS properties and domain-specific QoS properties. These previous QoS-aware Web service research usually assume that the Web service QoS information is already known or can be easily obtained from the service providers and third-party institution. However, there is so much missing QoS value to the service users in real-world. Therefore, dealing with the prediction of missing QoS value is useful to advance the service composition.

The problem of predicting the missing QoS value of Web service has received a lot of attention during the last years. In order to obtain a accurate prediction of missing QoS value, a practical prediction algorithm is required. Among the previous algorithms, Collaborative Filtering (CF) is a family of popular methods, and has been widely used in commercial recommender [16, 20, 21]. Typical collaborative filtering algorithms can be categorized into two classes: neighborhood-based methods and model-based methods [15]. Neighborhood-based method computes the similarity between users or items to make recommendation, which can be divided into two types: user-based nearest neighbor and item-based nearest neighbor [31]. For example, a group of users with similar interests, the items selected by one user can be recommended to others in the group. In [24], the authors propose a user-based CF algorithm to predict the QoS value from similar Web service users. Zheng et al. [31] propose a hybrid method of user-based and item-based CF algorithm to predict the QoS value on real-world Web services dataset. They developed an advanced Pearson Correlation Coefficient (PCC) measurement for user similarity computation, which addressed the problem that PCC often overestimates similarities of users who are actually not similar but happen to have similar QoS properties. Chen et al. [6] first discover the

[1]http://www.service4all.org.cn/servicexchange/

great influence of user location in Web services QoS prediction and propose a novel region-based hybrid CF algorithm. They integrate users into a hierarchy of regions according to both user locations and their QoS value, so that when identifying similar users for a specific user, the method only finds the region which the user belongs to. Jiang et al. [9] propose a personalized hybrid CF method which takes into account the influence of personalization of Web services when computing degree of similarity between users. It means that the more popular services should contribute the less degree to user similarity measurement.

Since neighborhood-based methods are sensitive to sparse data, the low-rank Matrix Factorization (MF) model [15] is widely used. MF characterizes both users and services by vectors of factors in a joint latent factor space of low dimension. The philosophy behind the model-based method is that a QoS value relates not only to how similar Web service users preferences and services features are, but also to the relationship between users and services interaction. In the model-based methods, training data is used to train a predictor model with fitting the *user-service* matrix with low-rank approximations. Although, model-based methods are considered more effective than those based on neighborhood, these two classes are often complementary. Recently, some researchers [14, 17, 30] blend them for obtaining better performance. In [30], a model-based method has been adopted to predict the Web service QoS value. The authors use the neighborhood-based method to identify similar users, and on the basis, MF method is employed to construct a global model. Lo et al. [17] use the geographical information of Web services and users to find similar neighbors, and apply MF method with location-based regularization to improve prediction accuracy.

The issue of the above concerns is that they deals with the *user-service* two-dimensional matrix data, without considering of the temporal information of Web service invocation. The absence of temporal information in QoS value prediction leads recommendation to suffering from the static model issue. We argue that temporal information is an essential element in QoS-aware Web service recommendation, so the prediction model should consider using the temporal information to reveal the complex triadic relations of *user-service-time* model, rather than the straightforward dyadic relations of *user-service* model. To overcome the drawbacks of CF methods described above and learn the triadic relations from the QoS value data, a tensor factorization approach is proposed for Web service QoS value prediction. We take the advantage of model-based method and extend it to model temporal three-dimensional data. The CANDECOMP/PARAFAC (CP) model [13] is utilized to represent the triadic relations among user, service and time. In this paper, we deal with constructing a temporal three-dimensional tensor whose element is the QoS value according to the triplet $\langle user, service, time\rangle$ and proposing a non-negative CP decomposition method [11, 27] to approximate this temporal QoS value tensor.

In real-world scenarios, the reason of limited work in the literature employing CF methods to predict Web service QoS value is the lack of real-world Web service QoS dataset for experimental studies. Without the convincing and sufficient real-world Web service QoS property data, the characteristics of Web service QoS value cannot be fully mined and the experimental results of the proposed prediction algorithms cannot be justified. In [1], a Web service QoS dataset is released, which is observed by one service user on 2,507 Web services. In [32], totally 1,974,675 real-world Web service invocations are executed by 339 service users on 5,825 real-world Web service. The fact that the QoS data is static cannot reflect the latent factors that govern the association relations among user,service and time factors. In order to explore the influence of dynamic factor, our released dataset includes more than 19 millions real-world Web service invocations observed by 343 service users in heterogenous network environment on 5,817 Web services during 32 continuous time periods.

Table 1: Notations of Tensor

Notations	Descriptions
$\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}$	tensor(calligraphic letters)
$\mathbf{A}, \mathbf{B}$	matrix(upper-case letters)
$\mathbf{A}_{(n)}$	n-mode matrix of the tensor
$\mathbf{A}^{(n)}$	n-subspace matrix of the tensor
$\mathbf{a}$,$\mathbf{b}$	vector(bold lower-case letters)
a, b	scalar(lower-case letters)
$\odot$	Khatri-Rao product
$\otimes$	Kronecker product
$*$	Hadamard product
$\circ$	outer product
$\cdot$	inner product

3. PRELIMINARIES

A tensor is a higher order generalization of a vector (i.e., first order tensor) and a matrix (i.e., second order tensor). Tensor factorization is an important technique in many applications, such as data mining, dimensionality reduction, chemometrics, signal processing, neuroscience, and web analysis [7, 10, 26, 8]. Before clarifying our model, some basic notations and operations for CP model are first introduced. There are mainly two popular kinds of TF approaches: Tucker decomposition and CP decomposition. In this paper we focus on CP decomposition model where our model is based.

3.1 Notations and Operations

In this paper, tensors are denoted by bold calligraphic upper-case letters $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \cdots$, matrices by bold upper-case letters $\mathbf{A}$,$\mathbf{B} \cdots$, vectors by bold lower-case letters $\mathbf{a}$,$\mathbf{b} \cdots$ and scalars by lower-case letters $a, b \cdots$. A tensor is a multi-dimensional or N-way array. An N-way tensor is denoted as: $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, which has N indices $(i_1, i_2, \cdots, i_N)$ and its elements are denoted by $a_{i_1 i_2 \cdots i_N}$. In tensor terminology, the n-mode matricization operation maps a tensor into a matrix. The n-mode matrix of an N-way tensor $\boldsymbol{\mathcal{A}}$ are the I_n-dimensional matrix obtained from $\boldsymbol{\mathcal{A}}$ by varying the index i_n and keeping the other indices fixed, and the elements of $\boldsymbol{\mathcal{A}}$ is mapped into the unfolding matrix $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times (I_1 I_2 \cdots I_{n-1} I_{n+1} \cdots I_N)}$. For example, $\mathbf{A}_{(2)}$ represents the mapping $\boldsymbol{\mathcal{A}}^{I \times J \times K} \rightarrow \boldsymbol{\mathcal{A}}_{(2)}^{J \times IK}$. The scalar product of two tensors $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$is defined as: $\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle = \sum_{i_1, i_2 \cdots, i_N} a_{i_1 i_2 \cdots i_N} b_{i_1 i_2 \cdots i_N}$. The n-mode product of a tensor $\boldsymbol{\mathcal{A}}$ by a matrix $\mathbf{U} \in \mathbb{R}^{J_n \times I_n}$ is an $I_1 \times I_2 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N$-tensor given by $(\boldsymbol{\mathcal{A}} \times_n \mathbf{U})_{i_1 \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i_n} a_{i_1 \cdots i_N} u_{j_n i_n}$. The Frobenius norm of a tensor $\boldsymbol{\mathcal{A}}$ is given by $\|\boldsymbol{\mathcal{A}}\| = \sqrt{\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{A}} \rangle}$. More

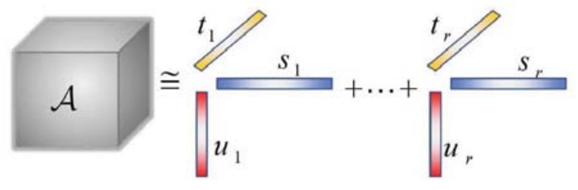


Figure 2: CP decomposition of a 3-way tensor into R components

details on matrix unfolding of tensor can be found in [13, 31]. A brief overview of important tensor notations is presented in Table 1.

3.2 CP Decomposition

An N-way tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is rank-1 if it can be written as the outer product of N vectors:

$$\boldsymbol{\mathcal{A}} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(N)}, \tag{1}$$

where $\mathbf{a}^{(n)} \in \mathbb{R}^{I_n}$ for $n = 1, 2, \cdots N$ is a vector. This means that each entry of the tensor can be written as:

$$\boldsymbol{\mathcal{A}}_{i_1 i_2 \cdots i_N} = \mathbf{a}^{(1)}_{i_1} \mathbf{a}^{(2)}_{i_2} \cdots \mathbf{a}^{(N)}_{i_N}, \tag{2}$$

where $\mathbf{a}^{(n)}_{i_n}$ is the i_nth element of the vector $\mathbf{a}^{(n)}$.

The rank of tensor $\boldsymbol{\mathcal{A}}$, denoted rank($\boldsymbol{\mathcal{A}}$) $= R_{\boldsymbol{\mathcal{A}}}$, is defined as the smallest number of rank-1 tensors. Then, the tensor $\mathcal{A}$ can be written as:

$$\boldsymbol{\mathcal{A}} \approx \sum_{r=1}^{R_{\boldsymbol{\mathcal{A}}}} \mathbf{a}^{(1)}_r \circ \mathbf{a}^{(2)}_r \circ \cdots \circ \mathbf{a}^{(N)}_r, \tag{3}$$

where the vector $\mathbf{a}^{(n)}_r \in \mathbb{R}^{I_n \times R_{\boldsymbol{\mathcal{A}}}}$ for $n = 1, 2, \cdots N$, and the set of vectors $\langle \mathbf{a}^{(n)}_1, \mathbf{a}^{(n)}_2, \cdots, \mathbf{a}^{(n)}_r \rangle$ can be rewritten as a matrix $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R_{\mathcal{A}}}$. Such a decomposition is called CP rank decomposition as illustrated in Figure (2). For instance, given a three-dimensional tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I \times J \times K}$, the n-mode of $\boldsymbol{\mathcal{A}}$ can be written as: $\mathbf{A}_{(1)} \approx \mathbf{U}(\mathbf{T} \odot \mathbf{S})^T, \mathbf{A}_{(2)} \approx \mathbf{S}(\mathbf{T} \odot \mathbf{U})^T, \mathbf{A}_{(3)} \approx \mathbf{T}(\mathbf{S} \odot \mathbf{U})^T$, where $\odot$ denotes the Khatri-Rao product. Following [12], the CP model can be concisely expressed as: $\boldsymbol{\mathcal{A}} \approx [\![\mathbf{U},\mathbf{S},\mathbf{T}]\!] = \sum_{r=1}^{R_{\boldsymbol{\mathcal{A}}}} \mathbf{u}_r \circ \mathbf{s}_r \circ \mathbf{t}_r$. Using the notations defined above, the CP decomposition problem can be formulated as alternating least-squares(ALS) optimization problem:

$$\min_{\mathbf{U},\mathbf{S},\mathbf{T}} \frac{1}{2} \| \boldsymbol{\mathcal{A}} - [\![\mathbf{U},\mathbf{S},\mathbf{T}]\!] \|_F^2, \tag{4}$$

where $\|\cdot\|_F$ is the Frobenius norm.To avoid the issue of model over-fitting, three regularization terms related to $\mathbf{U}$,$\mathbf{S}$ and $\mathbf{T}$ are involved as follow:

$$\min_{\mathbf{U},\mathbf{S},\mathbf{T}} \frac{1}{2} \| \boldsymbol{\mathcal{A}} - [\![\mathbf{U},\mathbf{S},\mathbf{T}]\!] \|^2 + \frac{\lambda}{2} (\|\mathbf{U}\|_F^2 + \|\mathbf{S}\|_F^2 + \|\mathbf{T}\|_F^2), \tag{5}$$

where λ are regularization parameters.

The ALS approach fixes $\mathbf{U}$ and $\mathbf{S}$ to find the best $\mathbf{T}$, then fixes $\mathbf{U}$ and $\mathbf{T}$ to find the best $\mathbf{S}$, then fixes $\mathbf{S}$ and $\mathbf{T}$ to find the best $\mathbf{U}$, and continues to repeat the entire procedure until some convergence criterion is satisfied. For example, given that $\mathbf{S}$ and $\mathbf{T}$ are fixed, the above optimal problem can be rewritten as:

$$\min_{\mathbf{U}} \frac{1}{2} \left\| \mathbf{A}_{(1)} - \hat{\mathbf{U}}(\mathbf{T} \odot \mathbf{S})^T \right\|_F^2 + \frac{\lambda}{2} I, \tag{6}$$

where I is the indicator function which is equal to 1 when user u invokes Web service s at time t and equal to 0 otherwise. The optimal solution of the above problem is given by:

$$\hat{\mathbf{U}} = \mathbf{A}_{(1)}(\mathbf{T} \odot \mathbf{S})(\mathbf{T}^T\mathbf{T} * \mathbf{S}^T\mathbf{S} + \lambda I)^\dagger, \tag{7}$$

where $*$ denotes the Hadamard product. In the same way, the optimal solutions of $\mathbf{S}$ and $\mathbf{T}$ are given by:

$$\hat{\mathbf{S}} = \mathbf{A}_{(2)}(\mathbf{T} \odot \mathbf{U})(\mathbf{T}^T\mathbf{T} * \mathbf{U}^T\mathbf{U} + \lambda I)^\dagger, \tag{8}$$

$$\hat{\mathbf{T}} = \mathbf{A}_{(3)}(\mathbf{T} \odot \mathbf{S})(\mathbf{S}^T\mathbf{S} * \mathbf{U}^T\mathbf{U} + \lambda I)^\dagger, \tag{9}$$

The approximation result of tensor $\hat{\boldsymbol{\mathcal{A}}}$ can be written as:

$$\hat{\boldsymbol{\mathcal{A}}} = \left[\!\left[\hat{\mathbf{U}}, \hat{\mathbf{S}}, \hat{\mathbf{T}} \right]\!\right] = \sum_{r=1}^{R_{\boldsymbol{\mathcal{A}}}} \hat{\mathbf{u}}_r \circ \hat{\mathbf{s}}_r \circ \hat{\mathbf{t}}_r, \tag{10}$$

Above the derivation corresponding to ALS algorithm for CP decomposition, the pseudo-code of this algorithm is given by Algorithm 1. For each iteration, the primary cost of the computation complexity is Eq.(7 - 10), each of which is $\mathcal{O}(IJKR_{\boldsymbol{\mathcal{A}}} + (2I + J + K)R^2_{\boldsymbol{\mathcal{A}}} + JKR_{\boldsymbol{\mathcal{A}}})$, and Eq.(10) is $\mathcal{O}(IJKR_{\boldsymbol{\mathcal{A}}})$. In typical cases, the term $(IJKR_{\boldsymbol{\mathcal{A}}})$ is much larger than the rest, so the computation complexity can be considered as $\mathcal{O}(IJKR_{\boldsymbol{\mathcal{A}}})$ in one iteration.

Algorithm 1 : The ALS algorithm for CP decomposition

Input: the tensor $\boldsymbol{\mathcal{A}}$, the rank R of tensor $\boldsymbol{\mathcal{A}}$, the regularization parameter λ.

Output: the approximate tensor $\hat{\boldsymbol{\mathcal{A}}}$, three factor matrices $\hat{\mathbf{U}}, \hat{\mathbf{S}}, \hat{\mathbf{T}}$.

```
1: Procedure [$\hat{\mathcal{A}}, \hat{U}, \hat{S}, \hat{T}$] = CP-ALS($\mathcal{A}, R, \lambda$)
2: Initialize: $U \in \mathbb{R}^{I \times R}, S \in \mathbb{R}^{J \times R}$, and $T \in \mathbb{R}^{K \times R}$ by
   small random value.
3: Repeat
4: Fix $S, T$. $\hat{U} \leftarrow Eq.(7)$
5: Fix $U, T$. $\hat{S} \leftarrow Eq.(8)$
6: Fix $U, S$. $\hat{T} \leftarrow Eq.(9)$
7: Until convergence or maximum iterations exhausted.
8: $\hat{\mathcal{A}} \leftarrow Eq.(10)$
9: Return: $\hat{\mathcal{A}}, \hat{U}, \hat{S}, \hat{T}$.
10: EndProcedure
```

4. PROPOSED MODEL

For Web service users, the service QoS properties are not as accurate as the service provider declared. To obtain accurate Web service QoS value for every service user, we employ the Temporal QoS-aware Web Service Recommendation Framework to make prediction of QoS value. As shown in Figure 3, our QoS prediction framework collects Web services QoS information from different service users. A Web service user can obtain the service QoS value prediction through our prediction framework, if the service QoS information contributions of the user surpass the threshold. The more service QoS information contributions, the higher QoS value prediction accuracy can be achieved. After collecting a large number of QoS information, we filter some inferior QoS information for the training data and employ the prediction engine to generate the predictor model for

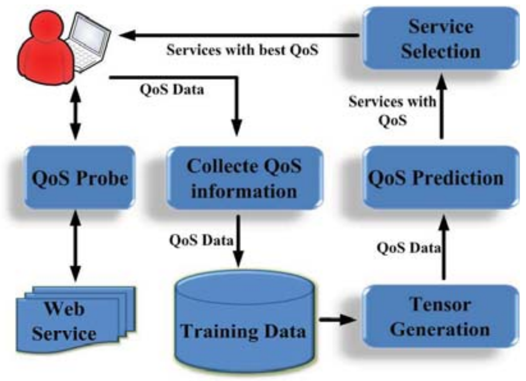


Figure 3: Temporal QoS-Aware Web Service Recommendation Framework

predicting the missing QoS value. Due to the space limitation, we mainly introduce the prediction algorithm principle in this paper.

4.1 Problem Statement

The research problem studied in this paper is stated as follows: *Given a Web service QoS dataset of temporal information with user-service interactions, recommend to each user under a given temporal context an optimal services list.* To illustrate these concepts, the following example is given.

A Toy Example: Consider the instance of recommending services to users in specific temporal context which is assigned to service invocation time in this paper. Then the $\langle user, service, time\rangle$ triplets have the following attributes:

- *User: the set of all service users to whom Web services are recommended; it is defined as UserID .*
- *Service: the set of all the Web services that can be recommended; it is defined as ServiceID .*
- *Time: the Web service invocation time when the user invoke the service; it is defined as TimeID .*

Then the service QoS value assigned to a service invocation from a user also depends on where and when the service was invoked. For instance, a specific service is recommended to users in different locations, significantly depending on when they are planning to invoke it.

Each QoS value is described by three dimensionality according to $userID$, $serviceID$ and $timeID$. Thus the QoS value is represented as points in the three-dimensional space, with the coordinates of each point corresponding to the index of the triplet $\langle userID, serviceID, timeID\rangle$. A straightforward method to capture the three-dimensional interactions among the triplet $\langle user, service, time\rangle$ is to model these relations as a tensor. The QoS value of Web service invocations from J services by I users at K time intervals are denoted as a tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I\times J\times K}$, i.e., a three-dimensional tensor, with $I \times J \times K$ entries which are denoted as $\boldsymbol{\mathcal{Y}}_{ijk}$: (1) $\boldsymbol{\mathcal{Y}}_{ijk} = Rating$ indicates the missing QoS value that the service j has been invoked by user i under the context type k, and $Rating$ is this service QoS value; (2) $\boldsymbol{\mathcal{Y}}_{ijk} = 0$ indicates that the service has not been invoked. The real-world Service QoS value dataset is very sparse, even though the density of the dataset collected by our system is only 30%.

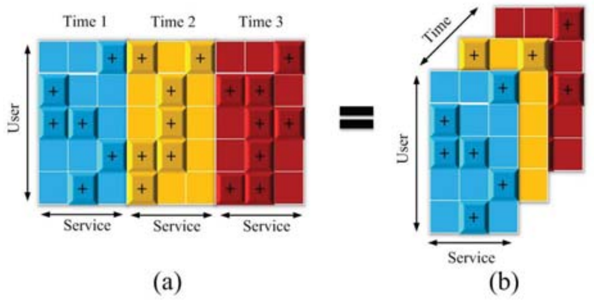


Figure 4: Slices of time-specific matrices with users and services are transformed into a temporal tensor

To obtain the missing QoS value in the *user-service-time* tensor, the Web service QoS observed by other service users can be employed for predicting the Web service for the current user. Once these initial Web service QoS value is obtained, our recommendation system will try to estimate the Web service QoS value which has not been obtained for the $\langle user, service, time\rangle$ triplets by using the QoS value function T:

$$UserID \times ServiceID \times TimeID \rightarrow Rating$$

where $UserID$, $ServiceID$ and $TimeID$ are the index of users, services and time periods, respectively and $Rating$ is the QoS value corresponding to the three-dimensional index.

As we can see from this example and other cases, an algorithm is needed to estimate the QoS value function T. In this paper, CP decomposition model is used to reconstruct the temporal three-dimensional *user-service-time* tensor. As mentioned in Section 3, the main idea behind CP decomposition model is to find a set of low-rank tensors to approximate the original tensor. Our approach is designed as a two-phase process. Firstly, the temporal QoS value tensor composed of the observed QoS value is constructed. Then we propose a non-negative tensor factorization approach to predict the missing QoS value in the tensor.

4.2 Construct QoS Value Tensor

When a service user invokes a Web service, the QoS properties performance will be collected by our recommendation system. After running a period of time, the recommender accumulates a collection of Web service QoS property data, which can be represented by a set of quadruplets $\langle UserID, ServicID, TimeID, Rating\rangle$(or $\langle u, s, t, r\rangle$ for short). Using the QoS value data, a temporal three-dimensional tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I\times J\times K}$ can be constructed, where I, J, K are the number of users, services and time periods, respectively. Each entry of tensor represents the QoS value of $\langle u, s\rangle$ pair at time period k.

The three-dimensional Temporal Tensor Construct algorithm is given in Algorithm 2: the input is a set of Web service QoS value, and the output is the constructed temporal tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I\times J\times K}$. Each frontal slice in tensor $\boldsymbol{\mathcal{Y}}$ corresponds to a $\langle u, s\rangle$ pair QoS value matrix for each time interval.

4.3 Non-negative CP Decomposition

In the real-world, the Web service QoS value is always non-negative, so the temporal QoS value tensor is presented

Algorithm 2 :Temporal Tensor Construct

Input: a set of quadruplets $\langle u,s,t,r\rangle$ for Web service QoS value dataset.

Output: a temporal tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I\times J\times K}$.

```
1: load all quadruplets $\langle u,s,t,r\rangle$ of the Web service Qos value,
2: use the set of $\langle u,s,1,r\rangle$ to construct a p user,service matrix $\mathbf{U}^{(1)}$ that takes all $I$ users as the rows and all $J$ services as the columns in the time of period 1,
3: the element of the matrix $\mathbf{U}^{(1)}$ is the $r$ of the quadruplet $\langle u,s,t,r\rangle$ according to the corresponding $\langle u,s,1\rangle$ triplet,
4: construct all the matrices $\mathbf{U}^{(1)},\mathbf{U}^{(2)},\cdots,\mathbf{U}^{(K)}$ for K time periods,
5: an augmented matrix $\mathbf{U}$ can be built by horizontally concatenating all matrices as shown in Figure 4 (a) denoted as $\mathbf{Y}_{(1)}$,
6: Construct tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I\times J\times K}$ as shown in Figure 4 (b), each slice of tensor is one matrix of $\mathbf{Y}_{(1)}$.
7: Return: $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I\times J\times K}$
```

as an non-negative three-way tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}_+^{I\times J\times K}$, and decomposed components are a set of matrices: $\mathbf{U} \in \mathbb{R}_+^{I\times R_{\boldsymbol{\mathcal{Y}}}}, \mathbf{S} \in \mathbb{R}_+^{I\times R_{\boldsymbol{\mathcal{Y}}}}$ and $\mathbf{T} \in \mathbb{R}_+^{I\times R_{\boldsymbol{\mathcal{Y}}}}$, here and elsewhere, $\mathbb{R}_+$ denotes the non-negative orthant with appropriate dimensions. As presented in the previous section, our goal is to find a set of factor matrices as the to approximate the tensor, whose rank is the number of the components. Adding the nonnegativity restriction to the CP decomposition model, we can get a non-negative CP decomposition model (NNCP). Our three-dimensional NNCP decomposition model is given by:

$$\boldsymbol{\mathcal{Y}} = \sum_{r=1}^{R_{\boldsymbol{\mathcal{Y}}}} \mathbf{u}_r \circ \mathbf{s}_r \circ \mathbf{t}_r + \boldsymbol{\mathcal{E}}, \tag{11}$$

where the vectors $\mathbf{u}_r, \mathbf{s}_r, \mathbf{t}_r$ are restricted to have only non-negative elements and the tensor $\boldsymbol{\mathcal{E}} \in \mathbb{R}_+^{I\times J\times K}$ is errors or noise depended on the application.

The QoS value tensor should be reconstructed for predicting all missing QoS value but the fitting Algorithm 1 for tensor is not based on the non-negative orthant. A new fitting algorithm which approximates the tensor with non-negative value should be designed. Firstly, we define a cost function to quantify the quality of approximation, which can be constructed using some measure of distance between two non-negative tensors $\boldsymbol{\mathcal{Y}}$ and $\hat{\boldsymbol{\mathcal{Y}}}$. One useful measure is simply the square of the Euclidean distance between $\boldsymbol{\mathcal{Y}}$ and $\hat{\boldsymbol{\mathcal{Y}}}$,

$$\|\boldsymbol{\mathcal{Y}} - \hat{\boldsymbol{\mathcal{Y}}}\|_F^2 = \sum_{ijk} (\boldsymbol{\mathcal{Y}}_{ijk} - \hat{\boldsymbol{\mathcal{Y}}_{ijk}})^2, \tag{12}$$

where $\boldsymbol{\mathcal{Y}}_{ijk}$ is the Web service QoS value of j-th service from i-th user at k-time, $\hat{\boldsymbol{\mathcal{Y}}_{ijk}}$ is the approximation value, the lower bound is zero, and clearly vanishes if and only if $\boldsymbol{\mathcal{Y}} = \hat{\boldsymbol{\mathcal{Y}}}$. Then, we consider the formulations of NNCP as a optimal problem:

$$\min_{\mathbf{u}_r,\mathbf{s}_r,\mathbf{t}_r} \frac{1}{2} \|\boldsymbol{\mathcal{Y}}_{ijk} - \sum_{r=1}^{R_{\boldsymbol{\mathcal{Y}}}} \mathbf{u}_r \circ \mathbf{s}_r \circ \mathbf{t}_r\|_F^2, \qquad s.t. \quad \mathbf{u}_r, \mathbf{s}_r, \mathbf{t}_r \geqslant 0. \tag{13}$$

We use multiplicative updating algorithms [23] for factor matrices $\mathbf{U}, \mathbf{S}$ and $\mathbf{T}$ to approximate the non-negative tensor. Then we are easy to obtain the partial derivative of the objective Eq.(13):

$$\frac{\partial f}{\partial \mathbf{u}_l^{(i)}} = \sum_{r=1}^{R_{\boldsymbol{\mathcal{Y}}}} \mathbf{u}_l^{(i)} (\mathbf{s}_r \cdot \mathbf{s}_l)(\mathbf{t}_r \cdot \mathbf{t}_l) - \sum_{j,k} \boldsymbol{\mathcal{Y}}_{ijk} \mathbf{s}_l^{(j)} \mathbf{t}_l^{(k)} \tag{14}$$

where $\mathbf{u}_l^{(i)}$ is the l-th column and i-th row element of factor matrix $\mathbf{U}$, $\mathbf{s}_r$ is the r-th vector of factor matrix $\mathbf{S}$, $l \in R_{\boldsymbol{\mathcal{Y}}}$, $\cdot$ denotes the inner product and for more details see [25]. Then we can obtain the following update rule by using a multiplicative update rule:

$$\mathbf{u}_l^{(i+1)} \leftarrow \frac{\mathbf{u}_l^{(i)} \sum_{j,k} \boldsymbol{\mathcal{Y}}_{ijk} \mathbf{s}_l^{(j)} \mathbf{t}_l^{(k)}}{\sum_{r=1}^{R_{\boldsymbol{\mathcal{Y}}}} \mathbf{u}_r^{(i)} (\mathbf{s}_r \cdot \mathbf{s}_l)(\mathbf{t}_r \cdot \mathbf{t}_l)}, \tag{15}$$

the updating rules for the rest of factor matrices can be easily derived in the same way, $\mathbf{s}_l^{(n)}$ and $\mathbf{t}_l^{(n)}$ are shown as follows:

$$\mathbf{s}_l^{(j+1)} \leftarrow \frac{\mathbf{s}_l^{(j)} \sum_{i,k} \boldsymbol{\mathcal{Y}}_{ijk} \mathbf{u}_l^{(i)} \mathbf{t}_l^{(k)}}{\sum_{r=1}^{R_{\boldsymbol{\mathcal{Y}}}} \mathbf{s}_r^{(j)} (\mathbf{u}_r \cdot \mathbf{u}_l)(\mathbf{t}_r \cdot \mathbf{t}_l)}; \tag{16}$$

$$\mathbf{t}_l^{(k+1)} \leftarrow \frac{\mathbf{t}_l^{(k)} \sum_{i,j} \boldsymbol{\mathcal{Y}}_{ikl} \mathbf{u}_l^{(i)} \mathbf{s}_l^{(j)}}{\sum_{r=1}^{R_{\boldsymbol{\mathcal{Y}}}} \mathbf{t}_r^{(k)} (\mathbf{u}_r \cdot \mathbf{u}_l)(\mathbf{s}_r \cdot \mathbf{s}_l)}, \tag{17}$$

where the vectors $\mathbf{u}_r, \mathbf{s}_r, \mathbf{t}_r$ are composed of non-negative value when they are initialized. So far we have described the details of NNCP algorithm for predicting the missing QoS value with non-negative value. In summary, Algorithm 3 gives the whole factorization scheme for NNCP. In each iteration of our algorithm, the new value of $\hat{\mathbf{U}}, \hat{\mathbf{S}}, \hat{\mathbf{T}}$ is calculated by multiplying the current value by a factor depended on the quality of the approximation in Eq.(11). The quality of the approximation improves monotonically with the application of these multiplicative update rules. The convergence proof of the multiplicative rule was introduced by Lee and Seung [23].

Given the latent factor matrices $\hat{\mathbf{U}}, \hat{\mathbf{S}}, \hat{\mathbf{T}}$, the prediction QoS value of Web service j from service user i at time k is given by:

$$\boldsymbol{\mathcal{Y}}_{ijk} \approx \sum_{r=1}^{R_{\boldsymbol{\mathcal{Y}}}} \mathbf{u}_r^{(i)} \mathbf{s}_r^{(j)} \mathbf{t}_r^{(k)}. \tag{18}$$

Notice that increasing the number $R_{\boldsymbol{\mathcal{Y}}}$ of components allows us to represent more and more factor structures of the Web service QoS value. However, as the number of components increases, we go from under-fitting to over-fitting these structures, i.e., we face the usual tradeoff between approximating complex structures and over-fitting them.

5. EXPERIMENTS

In this section, we introduce the experiment dataset, our evaluation metrics, and the experiment results. We use the QoS prediction accuracy to measure prediction quality, and address the following questions: (1) How do the tensor density and factor matrices dimensionality influence prediction accuracy? The factor matrices dimensionality determines how many the latent factors which have direct influence on prediction accuracy. (2) How does our approach compare with other CF methods?

Algorithm 3 : Non-negative CP decomposition algorithm

Input: the tensor $\mathcal{Y} \in \mathbb{R}_{+}^{I \times J \times K}$, the rank R of tensor $\mathcal{Y}$.

Output: three non-negative factor matrices $\hat{\mathbf{U}}, \hat{\mathbf{S}}, \hat{\mathbf{T}}$.

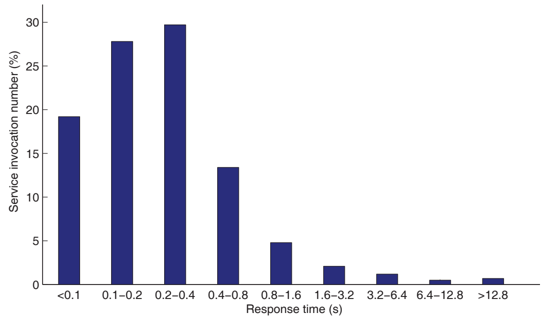
```
1: Procedure $[\hat{\mathbf{U}}, \hat{\mathbf{S}}, \hat{\mathbf{T}}] = \text{NNCP}(\mathcal{Y}, R)$
2: Initialize: $\mathbf{U} \in \mathbb{R}_{+}^{I \times R}, \mathbf{S} \in \mathbb{R}_{+}^{J \times R}$, and $\mathbf{T} \in \mathbb{R}_{+}^{K \times R}$ by small non-negative value.
3: Repeat
4: for $l = 1, \cdots, I$ do
5:     $\hat{\mathbf{U}} \leftarrow Eq.(15)$
6: end for
7: for $l = 1, \cdots, J$ do
8:     $\hat{\mathbf{S}} \leftarrow Eq.(16)$
9: end for
10: for $l = 1, \cdots, K$ do
11:     $\hat{\mathbf{T}} \leftarrow Eq.(17)$
12: end for
13: Until convergence or maximum iterations exhausted.
14: Return: $\hat{\mathbf{U}}, \hat{\mathbf{S}}, \hat{\mathbf{T}}$
15: EndProcedure
```

We implement the algorithms described in Section 4 with Matlab. For constructing the temporal QoS value tensor and solving the non-negative CP decomposition, we use the Matlab Tensor Toolbox [4]. The experiments are conducted on a Dell PowerEdge T620 machine with 2 Intel Xeon 2.00GHz processors and 16GB RAM, running Window Server 2008.

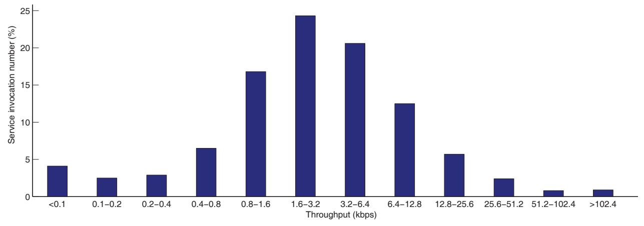
5.1 Dataset

To evaluate our proposed QoS prediction method in the real-world Web service, we implement a *ServiceXChange* and a *QoSDetecter*. The *ServiceXChange* is a platform, more than 20,000 openly-accessible Web services obtained by crawling Web service information from Internet. Recently we have integrated it with our other project *Service4All* [2] platform. We deploy our *QoSDetecter* on more than 600 distributed slices of **Planet-Lab**[3], which is a global research network that supports the development of new network services. We filter out slices which have successfully invoked at least 50 Web services so that there are enough observations to be split in various proportions of training and testing set for our evaluation. Finally, 343 slices were selected as the Web service users, and 5,817 publicly available real-world Web services are monitored by each slice continuously. The other of the more than 10,000 initially collected Web services are excluded in this experiment due to: (1) authentication required; (2) refused by the provider (e.g., the Web service is hosted by a private golf club); (3) permanent invocation failure (e.g., the Web service is shutdown). In this experiment, each 343 **Planet-Lab** slice invokes all the Web services continuously. This experiment dataset is consist of these Web services QoS performances of 4 days from July 26 to 29 of 2013 in 32 time intervals lasting for 3 hours.

We collect Web service invocation records from all the slices, and represent one observation in the dataset as a quadruplet $\langle u, s, t, r \rangle$. The dataset contains more than 19 million quadruplets, 343 users, 5,817 services and 32 time periods. Finally, we obtain two $343 \times 5817 \times 32$ *user-service-time* tensors. One tensor contains response time value, and the other one contains throughput value. Response time is defined as the persistent time between a service user sending a request and receiving the corresponding response, while throughput is defined as the average rate of successful message size per second. The statistics of Web service QoS performance dataset are summarized in Table 2. The distributions of response time and throughput are shown in Figure 5, and the more details of dataset and experiment result are released online for further research[4]. In Table 2, the means of response-time is 0.6840 seconds and throughput is 7.2445 kbps. In Figure 5 (a) shows that more than 95% of the response time elements are smaller than 1.6 seconds, and Figure 5 (b) shows that more than 99% of the throughput elements are smaller than 100 kbps. In this paper we only study the response-time and throughput, our NNCP method can be used to predicting any other QoS value directly without modifications. The value of the element in the three-dimensional tensor is the corresponding QoS value, when predicting value of a certain QoS value(e.g., popularity, availability, failure probability, etc.).



(a)



(b)

Figure 5: Web Services QoS Distribution

Table 2: Statistics of Web Service QoS Value

Statistics	**Response-Time**	**Throughput**
Scale	0-200s	0-1000kbps
Mean	0.6840	7.2445
Num. of service users	343	343
Num. of Web services	5817	5817
Num. of Time periods	32	32

5.2 Evaluation Measurements

Given a quadruplet $\langle u, s, t, r \rangle$ as $T = \langle u, s, t, r \rangle$, we evaluate the prediction quality of our method in comparison with other collaborative filtering methods using *Mean Absolute Error* (MAE) and *Root Mean Squared Error* (RMSE) [30].

[2]http://www.service4all.org.cn/

[3]http://www.planet-lab.org/

[4]http://www.service4all.org.cn/servicexchange/

Table 3: Web Service QoS Performance Comparison (A Smaller Value Means a Better Performance)

WS QoS Property	Method	MAE					RMSE				
		5%	10%	15%	20%	25%	5%	10%	15%	20%	25%
Response time	UMean	0.8156	0.7247	0.7161	0.6758	0.6361	2.3807	1.9589	1.9937	1.6229	1.4217
	IMean	0.5708	0.4919	0.4988	0.4158	0.4083	2.3344	2.0264	2.4146	2.0878	1.7216
	IPCC	0.6861	0.7972	0.5146	0.6014	0.4073	3.8511	3.8336	3.3770	2.5129	1.9188
	UPCC	0.5965	0.6627	0.6625	0.6014	0.5435	2.3424	1.8843	1.9331	1.5129	1.2671
	WSRec	0.5135	0.5252	0.5268	0.3947	0.3717	2.1838	2.0207	2.1533	1.7144	1.2975
	RSVD	0.9162	0.8375	0.8168	0.8088	0.7800	6.6970	5.2284	3.8099	4.9581	3.6419
	NNCP	0.4838	0.3589	0.3254	0.3178	0.3148	1.1470	1.0685	1.0502	1.0434	1.0399
Throughput	UMean	8.3696	8.4262	8.0827	7.7713	7.7113	32.7424	35.3732	32.8413	44.4918	40.9749
	IMean	6.7947	7.0433	6.4606	5.7356	5.2033	33.5447	34.5250	25.6687	22.7903	19.3721
	IPCC	8.2521	8.6508	8.1413	8.8179	8.3416	41.4411	40.9693	37.4096	48.9877	42.6471
	UPCC	8.0533	7.7259	7.1103	7.3437	7.0486	31.8687	32.9089	29.6238	29.2614	25.1004
	WSRec	6.3139	6.2608	5.9656	5.9222	4.7879	23.0171	24.6223	22.4384	22.3709	17.9580
	RSVD	9.6429	8.9885	7.5998	5.6261	5.1030	23.5928	25.4172	20.3695	19.7478	19.9420
	NNCP	6.0007	5.4889	4.9859	4.5001	4.0385	10.8098	10.1738	9.57085	8.98722	8.43047

MAE is defined as:

$$MAE = \frac{1}{|T|}\sum_{i,j,k}\left|\boldsymbol{\mathcal{Y}}_{ijk} - \hat{\boldsymbol{\mathcal{Y}}}_{ijk}\right| \tag{19}$$

where $\boldsymbol{\mathcal{Y}}_{ijk}$ denotes actual QoS value of Web service j observed by user i at time period k, $\hat{\boldsymbol{\mathcal{Y}}}_{ijk}$ represents the predicted QoS value of service j for user i at time period k, and $|T|$ is the number of predicted value. The MAE is the average absolute deviation of predictions to the ground truth data, and places the equal weight on each individual difference. RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{i,j,k}\left(\boldsymbol{\mathcal{Y}}_{ijk} - \hat{\boldsymbol{\mathcal{Y}}}_{ijk}\right)^2} \tag{20}$$

where smaller MAE (or RMSE) indicates better prediction accuracy. Since the errors are squared before they are averaged, the RMSE gives extra weight to relatively large errors.

5.3 Baseline Algorithms

For comparison purpose, we investigate whether our approach can be captured by the following 6 baseline algorithms, according to the prediction performance measured on the dataset. The baselines involved in this comparative experiment are listed below:

- *UMean*: This method uses the mean QoS value of all Web services QoS value from a service user who has invoked these services to predict the missing QoS value which this service user has not invoked.
- *IMean*: This method uses the mean QoS value of each Web service QoS value from all service users who have invoked this service to predict the missing QoS value which other service users have not invoked this service.
- *UPCC* (User-based collaborative filtering method using Pearson Correlation Coefficient): This method is a classical CF algorithm that involves similar user behavior to make prediction [24].
- *IPCC* (Item-based collaborative filtering method using Pearson Correlation Coefficient): This method is widely used in e-commerce scenarios [22].
- *WSRec*: This method is a hybrid collaborative algorithm that combines both UPCC and IPCC approaches, and employs both the similar users and similar Web services for the QoS value prediction [31].
- *RSVD*: SVD (Singular Value Decomposition) is proposed by [5] in Collaborative Filtering area, and used to exploit the '*latent structure*' of the original data. In this paper, we use the regularized SVD method proposed in [19].

In this part, the above six baseline methods are compared with our NNCP approach given the same training and testing cases. Since the baseline algorithms cannot be directly applied to context-aware prediction problem, we employ a special formulation for making the comparison with our NNCP approach. We consider the three-dimensional *user-service-time* tensor as a set of *user-service* matrix slices in terms of time interval. Firstly, we compress the tensor into a *user-service* matrix. Each element of this matrix is the average of the specific $\langle user, service\rangle$ pair during all the time intervals. For each slice of the tensor, the baseline algorithms are applied for predicting the missing QoS value. Secondly, we compute the MAE and RMSE of these baselines, and make the comparison with our NNCP method.

In the real-world, the dataset is usually very sparse since a service user usually only invokes a very small number of Web services. We randomly remove QoS value to sparse the dataset, and obtain the sparser dataset with different density from 5% to 25%, ascending by 5% each time. For example, dataset density 5% means that we randomly leave 5% of the dataset for training and the other value becomes testing set. The parameter settings of our NNCP method is that latent features dimensionality set as 20. The comparison result of this experiment are presented in Table 3, and the detailed investigations of parameter settings will be provided in the following subsections.

From Table 3, we can observe that our NNCP approach significantly improves the prediction accuracy, and obtains smaller MAE and RMSE value consistently for both response-time and throughput with different matrix densities. The MAE and RMSE value of throughput are much larger than those of response-time, because the range of throughput is 0-1000 kbps, while the range of response-time is only 0-20 seconds. With the increase of dataset density from 5% to 25%, the MAE and RMSE value of our NNCP method becomes smaller, because denser dataset provides more information for the missing QoS value prediction. Our NNCP method achieves better performance than the baselines. But some factors of disharmony in the Table 3 are that the MAE and RMSE of baselines are not decreasing with the increase of dataset density in the strict sense. The fluctuation is cause by that prediction value of the baselines is only in one layer, and the value of testing set intersperse among 32 layers, which increase the uncertainty of prediction.

5.4 Impact of Dataset Sparseness

In this section, we investigate the impact of data sparseness on the prediction accuracy as shown in Figure 6. We vary the density of the training matrix from 5% to 25% with a step of 5%. Figure 6 (a) and (b) are the MAE and RMSE results of response-time. Figure 6 (c) and (d) are the MAE and RMSE results of throughput. Figure 6 shows that: (1) With the increase of the training density, the performance of our method enhances indicating that better prediction is achieved with more QoS data. (2) Our NNCP method outperforms baselines consistently. The reason of this phenomenon is that baselines only utilize the two-dimensional static relations of *user-service* model without considering the more useful triadic relations of both the user and the service with the temporal information in the *user-service-time* model.

5.5 Impact of Dimensionality

The parameter dimensionality determines how many latent factors involve to tensor factorization. In this section, we investigate the impact of the dimensionality. We set the tensor density as 25%, and vary the value of dimensionality from 1 to 20 with a step value of one.

Figure 7(a) and (b) show the MAE and RMSE results of response-time, and Figure 7(c) and (d) show the MAE and RMSE results of throughput. Figure 7 shows that with the increase of latent factor number from 1 to 20, the value of MAE and RMSE keeps a declining trend. These observed results coincide with the intuition that relative larger number of latent factor produce smaller error ratio. But, more factors will require longer computation time and storage space. Moreover, when the dimensionality exceeds a certain threshold, it may cause the over-fitting problem, which will degrade the prediction accuracy.

6. CONCLUSIONS

Matrix Factorization is one of the most popular approaches to CF but the two-dimension model is not powerful to tackle the triadic relations of temporal QoS value. We extend the MF model to three dimensions through the use of tensor and employ the non-negative tensor factorization approach to advance the QoS-aware Web service recommendation performance in considering of temporal information. A systematic mechanism for Web service QoS value collection is designed and real-world experiments are conducted. In the experimental results, a higher accuracy of QoS value prediction is obtained with using the three-dimensional *user-service-time* model , when comparing our method with other standard CF methods.

In this paper, our recommendation approach only considers and models the relations between QoS value and the triplet $\langle user, service, time \rangle$. But in other cases, service users in different geographic locations at the same time may observe different QoS performance of the same Web service. Besides the temporal contextual information, more contextual information that influences the client-side QoS performance (e.g., the workload of the service servers, network conditions of the users, etc.) should be considered to improve the prediction accuracy.

In our future work, more real-world Web services will be monitored, and more QoS properties and contextual information will be collected. Since there are so many CF methods, we also consider to explore an ensemble method for aggregating various CF algorithms to predict missing QoS value.

7. ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China (No. 61103031, No. 61370057), China 863 program (No. 2012AA011203), China 973 program (No. 2014CB340300, No. 2014CB340304), A Foundation for the Author of National Excellent Doctoral Dissertation of PR China (No. 201159), and Beijing Nova Program(2011022). We are pleased to acknowledge Mingyue Liang for collecting and preparing the dataset.

8. REFERENCES

[1] E. Al-Masri and Q. H. Mahmoud. Investigating web services on the world wide web. In *Proceedings of the 17th international conference on World Wide Web*, pages 795–804. ACM, 2008.

[2] M. Alrifai and T. Risse. Combining global optimization with local selection for efficient qos-aware service composition. In *Proceedings of the 18th international conference on World wide web*, pages 881–890. ACM, 2009.

[3] M. Alrifai, D. Skoutas, and T. Risse. Selecting skyline services for qos-based web service composition. In *Proceedings of the 19th international conference on World wide web*, pages 11–20. ACM, 2010.

[4] B. W. Bader, T. G. Kolda, et al. Matlab tensor toolbox version 2.5. Available online, January 2012.

[5] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *ICML*, volume 98, pages 46–54, 1998.

[6] X. Chen, X. Liu, Z. Huang, and H. Sun. Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation. In *Web Services (ICWS), 2010 IEEE International Conference on*, pages 9–16. IEEE, 2010.

[7] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation.* Wiley. com, 2009.

[8] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and C. Zhu. Personalized recommendation via cross-domain triadic factorization. In *Proceedings of the 22nd international conference on World Wide Web*, pages 595–606. International World Wide Web Conferences Steering Committee, 2013.

[9] Y. Jiang, J. Liu, M. Tang, and X. Liu. An effective web service recommendation method based on personalized collaborative filtering. In *Web Services*

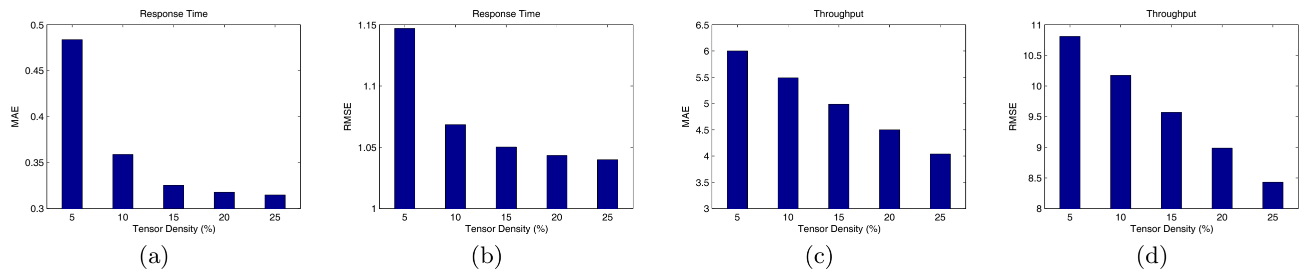


Figure 6: Impact of Tensor Density (Dimensionality = 20)

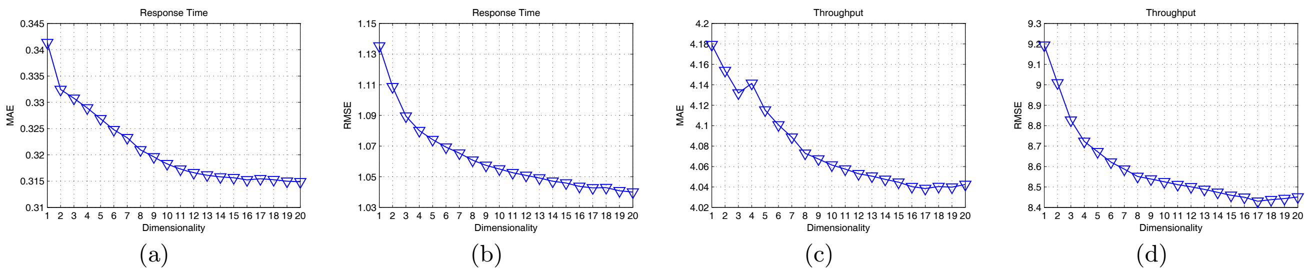


Figure 7: Impact of Factor Matrices Dimensionality (Tensor Density = 25%)

(ICWS), 2011 IEEE International Conference on, pages 211–218. IEEE, 2011.

[10] J. Kim. Nonnegative matrix and tensor factorizations, least squares problems, and applications. 2011.

[11] J. Kim and H. Park. Fast nonnegative tensor factorization with an active-set-like method. In *High-Performance Scientific Computing*, pages 311–326. Springer, 2012.

[12] T. G. Kolda. *Multilinear operators for higher-order decompositions*. United States. Department of Energy, 2006.

[13] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[14] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[15] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[16] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[17] W. Lo, J. Yin, S. Deng, Y. Li, and Z. Wu. Collaborative web service qos prediction with location-based regularization. In *Web Services (ICWS), 2012 IEEE 19th International Conference on*, pages 464–471. IEEE, 2012.

[18] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann. Service-oriented computing: State of the art and research challenges. *Computer*, 40(11):38–45, 2007.

[19] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[20] Y. Ren, G. Li, and W. Zhou. *Automatic Generation of Recommendations from Data: A Multifaceted Survey*. Deakin University, School of Information Technology, 2011.

[21] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[22] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[23] D. Seung and L. Lee. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13:556–562, 2001.

[24] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei. Personalized qos prediction forweb services via collaborative filtering. In *Web Services, 2007. ICWS 2007. IEEE International Conference on*, pages 439–446. IEEE, 2007.

[25] A. Shashua and T. Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, pages 792–799. ACM, 2005.

[26] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *Proceedings of the 14th international conference on World Wide Web*, pages 382–390. ACM, 2005.

[27] M. Welling and M. Weber. Positive tensor factorization. *Pattern Recognition Letters*, 22(12):1255–1261, 2001.

[28] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality driven web services composition. In *Proceedings of the 12th international conference on World Wide Web*, pages 411–421. ACM, 2003.

[29] L. Zeng, B. Benatallah, A. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. Qos-aware middleware for web services composition. *Software Engineering, IEEE Transactions on*, 30(5):311–327, 2004.

[30] Z. Zheng, H. Ma, M. Lyu, and I. King. Collaborative web service qos prediction via neighborhood integrated matrix factorization. 2012.

[31] Z. Zheng, H. Ma, M. R. Lyu, and I. King. Wsrec: A collaborative filtering based web service recommender system. In *Web Services, 2009. ICWS 2009. IEEE International Conference on*, pages 437–444. IEEE, 2009.

[32] Z. Zheng, Y. Zhang, and M. R. Lyu. Distributed qos evaluation for real-world web services. In *Web Services (ICWS), 2010 IEEE International Conference on*, pages 83–90. IEEE, 2010.