A Spatial-Temporal QoS Prediction Approach for Time-aware Web Service Recommendation

XINYU WANG and JIANKE ZHU, Zhejiang University
ZIBIN ZHENG, Sun Yat-sen University
WENJIE SONG and YUANHONG SHEN, Zhejiang University
MICHAEL R. LYU, The Chinese University of Hong Kong

Due to the popularity of service-oriented architectures for various distributed systems, an increasing number of Web services have been deployed all over the world. Recently, Web service recommendation became a hot research topic, one that aims to accurately predict the quality of functional satisfactory services for each end user. Generally, the performance of Web service changes over time due to variations of service status and network conditions. Instead of employing the conventional temporal models, we propose a novel spatial-temporal QoS prediction approach for time-aware Web service recommendation, where a sparse representation is employed to model QoS variations. Specifically, we make a zero-mean Laplace prior distribution assumption on the residuals of the QoS prediction, which corresponds to a Lasso regression problem. To effectively select the nearest neighbor for the sparse representation of temporal QoS values, the geolocation of web service is employed to reduce searching range while improving prediction accuracy. The extensive experimental results demonstrate that the proposed approach outperforms state-of-art methods with more than 10% improvement on the accuracy of temporal QoS prediction for time-aware Web service recommendation.

Categories and Subject Descriptors: H.3.5 [**Online Information Services**]: Web-Based Services

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Web service, service recommendation, QoS prediction, spatial-temporal QoS prediction

ACM Reference Format:
Xinyu Wang, Jianke Zhu, Zibin Zheng, Wenjie Song, Yuanhong Shen, and Michael R. Lyu. 2016. A spatial-temporal QoS prediction approach for time-aware Web service recommendation. ACM Trans. Web 10, 1, Article 7 (January 2016), 25 pages.
DOI: http://dx.doi.org/10.1145/2801164

7

This work was partially supported by the Major State Basic Research Development Program of China (973 Program No. 2015CB352201), National Natural Science Foundation of China under Grants (61103105 and 61472338), and Guangdong Natural Science Foundation (Project No. 2014A030313151). Michael R. Lyu is supported by the Research Grants Council General Research Fund (CUHK 415113). Xinyu Wang is supported by National Key Technology R&D Program of the Ministry of Science and Technology of China (No. 2015BAH17F01).
Authors' addresses: X. Wang, J. Zhu, W. Song, and Y. Shen, the College of Computer Science, Zhejiang University, Hangzhou, China, 310027; email: {wangxinyu, jkzhu, seeyou, yhshen}@zju.edu.cn; Z. Zheng, School of Advanced Computing, Sun Yat-sen University, and Collaborative Innovation Center of High Performance Computing, National University of Defense Technology; email: zbzheng@cse.cuhk.edu.hk; M. R. Lyu, CSE Department, The Chinese University of Hong Kong; email: lyu@cse.cuhk.edu.hk; J. Zhu is the Corresponding Author.

DOI: http://dx.doi.org/10.1145/2801164

1. INTRODUCTION

Web service is a software component encapsulating a well-defined business functionality [Klein et al. 2012]. With the rapid development of cloud computing, more and more services are deployed across continents for users worldwide. In general, users extract a list of candidate Web services from different service brokers to compose complicated Service Oriented Applications (SOA), where one or more optimal services are selected to build the applications. Obviously, it is quite ineffective to evaluate all the real-world candidate services with equivalent function in order to find the optimal one [Zheng et al. 2009]. Therefore, it is important to study personalized Web service recommendation [Zhang et al. 2007].

The key to service selection and recommendation [Zhang et al. 2007; Zeng et al. 2004; Yu et al. 2007] is Quality of Services (QoS), which is designed to distinguish among different functionally equivalent services. Specifically, QoS is defined as a set of user-experienced properties for a specific Web service including response time, throughput, reputation, and the like. QoS prediction is essential to effective personalized service recommendation [Zheng et al. 2009].

Practically, the end users are from various geographical locations whose QoS values are greatly dependent on network conditions and network geographical locations. Different users may have quite different QoS even using the same service, and the QoS values evaluated by one user cannot be used directly by others. This makes QoS prediction a challenging task [Chen et al. 2013]. To tackle this issue, collaborative filtering is widely employed to predict the QoS values of all candidate services for the personalized Web service recommendation [Zheng et al. 2009; Chen et al. 2013; Rong et al. 2009; Shao et al. 2007].

In a highly dynamic Internet environment, QoS values for Web services usually change with time [Zheng et al. 2012], and service status, such as the number of clients and network conditions, always varies with time [Zhang et al. 2011]. Therefore, Web services with optimal QoS values also change over time, sometimes drastically. For a better illustration, we employ the following function to evaluate the changing rate of QoS values between two adjacent time slots [Zheng et al. 2012]:

$$r_i = |q_i - q_{i-1}|/q_{i-1}, \tag{1}$$

where q_i and q_{i-1} represent the QoS values of the time slots i and $i-1$ respectively. r_i denotes the changing rate between these two time slots. If r_i is greater than a threshold α, we define q_i as a sudden change in QoS value. We select a set of temporal response time data from a QoS repository [Zhang et al. 2011] that includes the QoS values from a user of two different services. As shown in Figure 1, there are lots of sudden changes in the temporal QoS sequence of Web service when α is below 1.0. Although temporal models such as ARIMA [Godse et al. 2010] can be employed to capture the dynamic behaviors of QoS values like response time, it is still challenging to predict sudden changes in QoS values due to its stationary stochastic characteristics. This may lead to recommending an inappropriate service to the user, which will degrade the performance of SOA systems.

Generally, data-driven methods can be employed for temporal QoS prediction, and these are usually formulated as missing value problems. Specifically, a large number of user-service-time-aware QoS values are collected offline and are further used to build a tensor representation. Assuming that the QoS values lie in few latent subspaces, the missing item is predicted by taking advantage of tensor decomposition [Zhang et al. 2011]. However, this low rank assumption may not be valid for those temporal QoS values with a large amount of sudden changes. In the time-aware service recommendation, a number of QoS values for candidate services should be predicted for the current time. To account for the dynamic environment of Web service recommendation, the QoS

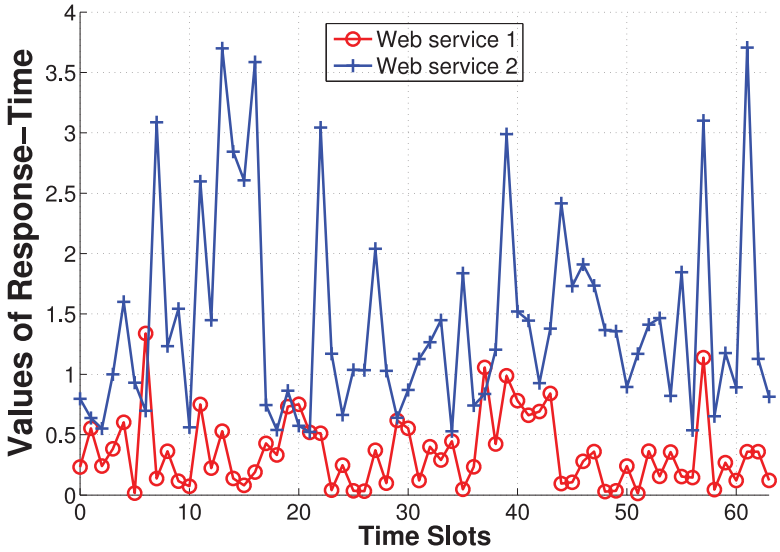

Fig. 1. Time-aware response-times of two Web service.

prediction should be invoked continuously to take advantage of currently and recently collected QoS values from other users. Factorization-based methods require rebuilding the model in order to make use of the newly accumulated QoS information. This will incur heavy computational cost.

To address these limitations, we propose a spatial-temporal QoS prediction approach for time-aware Web service recommendation in which sparse representation is employed to model QoS variations. Specifically, we make a zero-mean Laplace prior distribution assumption on the residuals of QoS prediction, which corresponds to a Least Absolute Shrinkage and Selection Operator (Lasso) problem. To effectively select the nearest neighbor for building the sparse representation, the geo-locations of Web services and users are employed to reduce the searching range while improving prediction accuracy. We conduct extensive experiments on a large-scale real-world dataset to study the efficacy of our proposed method compared with other state-of-art approaches. The promising experimental results demonstrate that our approach outperforms the state-of-art methods with more than 10% improvement on the accuracy of temporal QoS prediction for Web service recommendation. In addition, our method performs especially well on the task of predicting QoS values with sudden changes. We also show the effective results of providing QoS prediction and reprediction continuously with the newly accumulated QoS data.

The rest of this article is organized as follows. Section 2 reviews some existing approaches for Web service recommendation. Section 3 presents our proposed spatial-temporal approach to time-aware Web service recommendation. Section 4 describes a QoS reprediction method for the accumulated data. Section 5 provides our experimental results and the details of our experimental implementation. Section 6 sets out our conclusion and addresses future work.

2. RELATED WORK

Extensive research efforts have been devoted to Web service recommendation [Liu et al. 2010]. The key to QoS-aware Web service recommendation is to accurately predict QoS values [Zhang et al. 2007; Zeng et al. 2004; Yu et al. 2007; Menasce 2002;

Jaeger et al. 2004], which are employed to help the user distinguish among a huge number of functionally equivalent Web services and improve the overall quality of Web applications.

In general, QoS performance for Web services is typically measured from the user side, including response time, reliability, and throughput. Therefore, different users may have quite different QoS values for the same service. In this article, we focus on the time-aware Web service recommendation. The related work can be roughly categorized into two groups: temporal model-based methods and data-driven approaches.

For temporal model-based methods, the collected QoS values are typically treated as a time series, in which lots of techniques on modeling the dynamic behaviors of QoS characteristics can be employed to predict the temporal QoS values. Godse et al. [2010] monitor the QoS data continuously while forecasting the QoS values based on an ARIMA model, which involves a single time expert without human intervention during the execution. Li et al. [2009] take advantage of the structural equation time series model to fit the QoS values of Web services and to predict the change of QoS values dynamically. However, this approach mainly concerns global QoS containing multiple QoS attributes, which does not focus on real-world QoS values (i.e., response time and throughput). Cavallo et al. [2010] conducted an empirical study on QoS prediction including average values, linear models, and time series that demonstrated that the time series-based method can achieve promising prediction accuracy. Zeng et al. [2008] argue that the performance metrics and Key Performance Indicators (KPI) can be predicted using the ARIMA time series model in QoS management. They present the design and implementation of a novel event-driven QoS prediction system that can process operational service events in a real-time fashion to predict or refine the prediction of metrics and KPIs. Amin et al. [2012] integrate both ARIMA and GARCH models to capture QoS attributes' volatility. Hooman and Kennedy [2009] propose a Historical Symbolic Delay Approximation (HDAX) model to predict network delays. Experimental results demonstrate that their method shows better prediction accuracy in forecasting the delay-time series as well as in reducing the time cost of the forecasting method. The main show-stopper is that these methods require a large amount of historical QoS values and use some special assumptions.

For the data-driven approaches, QoS value prediction is usually formulated as a missing item problem, which is typically solved by Collaborative Filtering (CF) and factor analysis methods. CF [Zheng et al. 2010; Cai et al. 2010] was introduced to personalized QoS prediction and Web service recommendation [Chen et al. 2013; Shao et al. 2007; Zheng et al. 2009, 2011; Sun et al. 2012; Luo et al. 2012], which can be categorized into two groups: memory-based methods and model-based approaches. The memory-based CF methods employ user-rating data to compute the similarity between users or items and predict QoS values accordingly [McLaughlin and Herlocker 2004; Miller et al. 2003]. They include user-based approaches [Chen et al. 2009], item-based approaches [Deshpande and Karypis 2004], and hybrid approaches [Gong 2010; Zheng et al. 2011]. Tang et al. [2012] propose the location-aware DF method that incorporates the locations of both users and services, which reduces the search space for similar users and services and improves Web service recommendation performance. Sun et al. [2012] propose a normal recovery CF approach with improved Web service similarity. Yao et al. [2013] combine CF and content-based recommendation by taking into consideration both rating data and the content of Web services.

All these approaches, however, neglect QoS variations across different times. The model-based CF approaches, such as Bayesian model [Chen and George 2002] and K-means clustering [Ungar and Foster 1998], learn the statistical model from a training dataset, including clustering models [Ungar and Foster 1998], latent factor models [Salakhutdinov and Mnih 2008], and an aspect model [Singla and Richardson

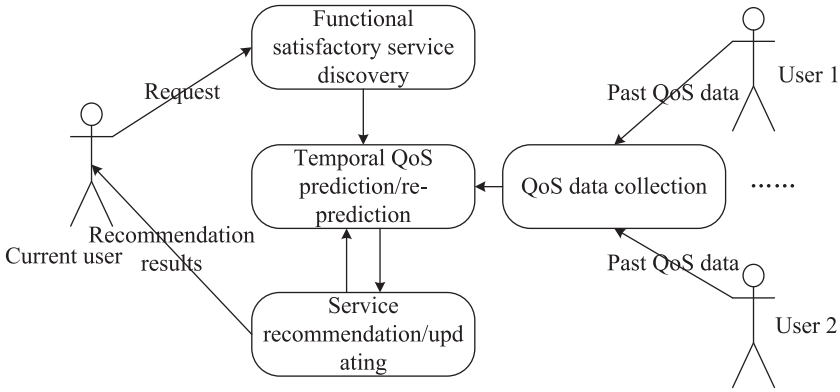

Fig. 2. Time-aware Web service recommendation framework based on spatial-temporal QoS prediction.

2008]. Chen et al. [2013] consider the users' physical locations and combine the model-based and memory-based CF algorithms, which improves personalized QoS prediction accuracy with reduced time complexity. To improve prediction accuracy, Luo et al. [2012] propose a collaborative QoS prediction framework with Location-Based Regularization (LBR), which incorporates geographical information to identify the neighborhood using the local connections between Web services users. Yu et al. [2013] propose an effective iterative algorithm to obtain the optimal completion of an arbitrary QoS matrix through factor analysis. Zhang et al. [2011] propose a WSPred method to provide a temporal personalized QoS value prediction service for different service users by tensor factorization. However, it is not easy to directly employ WSPred for a time-aware personalized service recommendation.

3. TIME-AWARE SERVICE RECOMMENDATION

In this section, we first present our proposed framework for time-aware Web service recommendation based on continuous temporal QoS prediction, which is formulated as a generic regression problem. Then, we propose a spatial-temporal QoS prediction approach using Lasso, which can take advantage of the geo-locations of both the end users and their associated Web services.

3.1. Overview

The objective of the time-aware service recommendation is to provide functionally optimal services to each user for the current time slot (i.e., an hour or a month). To improve recommendation accuracy, service users usually require an evaluation of the performance of all the functionally satisfied services for the current time slot, which incurs heavy computational cost and network traffic. To solve this problem, we present an effective spatial-temporal QoS prediction for time-aware service recommendation. As shown in Figure 2, the overall procedures of the time-aware Web service recommendation are summarized in the following:

(1) An end user requests a service in the current time slot;
(2) A set of satisfactory functional services are retrieved by the service search module;
(3) The temporal QoS prediction/reprediction module predicts the QoS values of the set of retrieved service for the current time slot;
(4) The service recommendation module selects the top m services with the optimal predicted QoS values for the current time slot and recommends these services to the end user.

As described in Zheng et al. [2011], the QoS data are collected through a user-collaborative QoS collection mechanism that archives the current or most recent QoS values from different services continuously contributed by different service users over time. Based on the newly collected QoS values, the temporal QoS prediction can be performed continuously. Additionally, repredicted QoS can be conducted using the newly collected temporal QoS values to further improve the prediction accuracy of the recommended services for the current time slot since more and more temporal QoS values are accumulated in the current time slot. Therefore, the services with optimal predicted QoS values for the current time slot can be adapted, and the recommended services can also be updated continuously to the service user.

3.2. Temporal QoS Prediction by Generic Regression

The key to time-aware Web service recommendation is to predict QoS values from a set of collected temporal QoS data. As with other prediction problem, QoS prediction can be solved using data driven methods, such as linear regression. The fundamental idea of the data-driven approach is to represent the QoS value by a linear combination of a set of similar examples. The popular matrix factorization-based CF methods also assume that the QoS data lies on the subspace spanned by the historical training data, which also resorts to representing the data point using the linear combination of similar examples. To this end, we formulate QoS prediction as a generic regression problem.

Given a set of collected temporal QoS data $\mathcal{X}$ with respect to users and services, we aim to predict the QoS value y_n at time slot n for the user u and service v. Let $\mathbf{y} = (y_1, \ldots, y_{n-1})^T$ denote the collected temporal QoS data for the user u and service v, where y_k $(1 \leq k \leq n-1)$ is the collected QoS value at time slot t_k. If the QoS value at t_k is not collected, then y_k is set to null (invalid). The fundamental idea of linear regression for QoS prediction is that the current QoS value can be represented by the linear combination of K most similar QoS sequences $X = (\mathbf{x}_1, \ldots, \mathbf{x}_K)^\top$ in the dataset. Therefore, we try to find a latent mapping function $f(x)$ between the previously collected historical data and the observation of QoS value for the user u and service v. Typically, we can assume that the QoS value y differs from the prediction by some additive Gaussian noise ϵ with zero mean and variance σ^2. Therefore, the target QoS value y can be represented as follows:

$$y = f(x) + \epsilon.$$

Let $\mathbf{w} \in R^{K\times 1}$ denote a vector of linear combination coefficients; then, the mapping function $f(x)$ can be formulated as:

$$f(x) = \sum_{i=1}^{K} w_i x_i = \mathbf{w}^\top \mathbf{x}. \tag{2}$$

Least squares regression minimizes the sum of squared distances between the observed QoS values for the user u and service v and the one predicted by the linear mapping function by minimizing the squared residual error:

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{w}^\top \mathbf{X}\|_2^2 + \lambda_1 \|\mathbf{w}\|_2^2, \tag{3}$$

where $\|\mathbf{w}\|_2$ is the regularization term and λ_1 is the regularization coefficient to avoid the overfitting issue. Equation (3) has the following closed-form solution:

$$\mathbf{w} = (X^\top X + \lambda_1 I)^{-1} X^\top \mathbf{y}.$$

Here, $I \in R^{K\times K}$ denotes the identity matrix. Therefore, the current QoS value for the user u and service v can be simply computed by the mapping function $f(x)$ in

Equation (2). The computational complexity for least squares regression is dominated by solving a linear system with the size $K \times K$ system matrix.

As mentioned in Section 1, QoS values vary drastically over time. However, the normal distribution assumption on prediction errors in least squares regression cannot effectively handle a large amount of abrupt changes.

3.3. QoS Prediction Using Lasso

In this article, we consider the zero mean Laplace distribution with probability density functions that have abrupt changes in gradient, which corresponds to a Lasso problem [Tibshirani 1994]. Lasso shrinks some coefficients and sets others to zero, and hence tries to retain the good features of both subset selection and regularized least square regression. This further leads to an effective sparse representation for temporal QoS data.

Instead of L_2 norm regularization in least square regression, Lasso imposes an L_1 penalty on the linear combination coefficients, which leads to the following optimization problem:

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{w}^\top \mathbf{X}\|_2^2 + \lambda \|\mathbf{w}\|_1, \tag{4}$$

where $\lambda \geq 0$ is the regularization coefficient that controls the amount of shrinkage on the predictions. Therefore, some coefficients shrink and may be exactly equal to zero. Obviously, this Lasso minimization problem uses a quadratic programming with linear inequality constraints, which can be efficiently solved by either a Least Angle Regression (LARS) algorithm [Efron et al. 2004; Mairal et al. 2010] or an L_1-regularized least squares method [Kim et al. 2007].

Once the optimal linear combination coefficients $\mathbf{w}$ is obtained through Lasso, the QoS value for the current time slot can be directly predicted by Equation (2). Note that the computational cost for Lasso is closely related to the size of selected example QoS sequences K. Because there are usually many user-service pairs, it is very time-consuming to use all the related information to predict the QoS value from a temporal sequence. Practically, we only select a very small portion of examples from the collected QoS dataset in order to facilitate the online applications. In the following, we discuss how to effectively choose the most similar sequences in the time-aware Web service recommendation task.

3.4. Spatial-Temporal QoS Prediction

To capture the dynamic characteristics of the input sequence $\mathbf{y}$, we employ the normalized cross-correlation between each sample $\mathbf{x}$ in the collected QoS dataset with $\mathbf{y}$ as the similarity measure:

$$S(\mathbf{x}, \mathbf{y}) = \frac{(\mathbf{x} - \bar{\mathbf{x}})^\top (\mathbf{y} - \bar{\mathbf{y}})}{\|\mathbf{x} - \bar{\mathbf{x}}\| \cdot \|\mathbf{y} - \bar{\mathbf{y}}\|}. \tag{5}$$

Moreover, we select the top K sequences with the highest cross-correlation scores. Because the size of the collected QoS dataset is usually very large, it is very inefficient to perform a linear scan across the whole dataset.

On the other hand, the geo-location of both users and services can be easily obtained by mapping the IP addresses to the geographical points using the precollected database [Heath 2011]. Because geographically closed user-service pairs have more chances to share the same IT infrastructures, such as routers and network workloads, they may have quite similar QoS values or trends [Shen et al. 2013], especially in terms of the sudden changes occurring over the time domain. Thus, the temporal QoS values of spatially close user-service pairs are very likely correlated. In this article, we

Table I. Spatial Similarity: The Discovered Maximal Correlation Coefficient in Temporal QoS Sequences

Spatial similarity (km)	Mean discovered maximal correlation coefficient in response time sequences
(0, 500]	0.837
(500, 1000]	0.818
(1000, 1500]	0.784
(1500, 2000]	0.743
(2000, 2500]	0.598

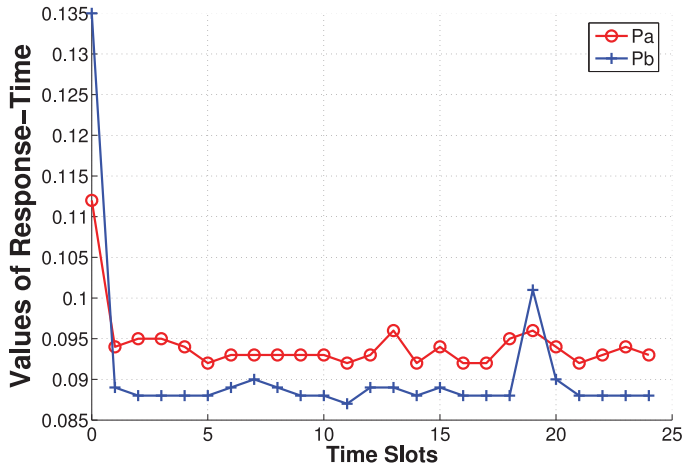

Fig. 3. Temporal response time sequences: $\mathbf{y}_a$ (corresponding to P_a) and $\mathbf{x}_b$ (corresponding to P_b), $S(\mathbf{y}_a, \mathbf{x}_b) = 0.955$, $simS(P_a, P_b) = 83.121$km.

employ the spatial information from user-service pairs to effectively reduce the searching range while improving prediction performance.

We first testify our assumption that the temporal QoS values of spatially close user-service pairs are very likely correlated. We collected a large number of test cases from a QoS repository [Zhang et al. 2011]. In each test case, we randomly choose a temporal QoS sequence $\mathbf{y}_a$ corresponding to a user-service pair P_a and search for its most correlated temporal QoS sequence $\mathbf{x}_b$ corresponding to user-service pair P_b within the different spatial similarities between P_a and P_b in all the other sequences. Each validation result is the average of over 1,000 different test cases. We employ the spatial similarity $simS$ to represent the spatial distance between two user-service pairs. Let $P_a = (u_a, v_a)$ and $P_b = (u_b, v_b)$, where u and v denote the corresponding user and service. Spatial similarity $simS$ between two user-service pairs means the average value of two geodesic distances between the two users and two services: $simS(P_a, P_b) = \frac{1}{2}(dist(u_a, u_b) + dist(v_a, v_b))$, where $dist()$ denotes a geodesic distance function. As shown in Table I, we can find that the most correlated QoS sequences of $\mathbf{y}_a$ are more likely to be discovered in the spatially similar user-service pairs for P_a. Therefore, we can conclude that the spatial information of users and services can be utilized for discovering the most correlated temporal QoS sequences. Figure 3 presents an example of a discovered $\mathbf{x}_b$ corresponding to the user-service pair P_b with a very high correlation coefficient compared to $\mathbf{y}_a$ corresponding to the user-service pair P_a.

Based on this analysis, we take advantage of the spatial information of user u and service v to predict the current temporal QoS sequence $\mathbf{y}$, as well as the spatial information of each user u_i and service v_i of the collected reference temporal QoS sequences

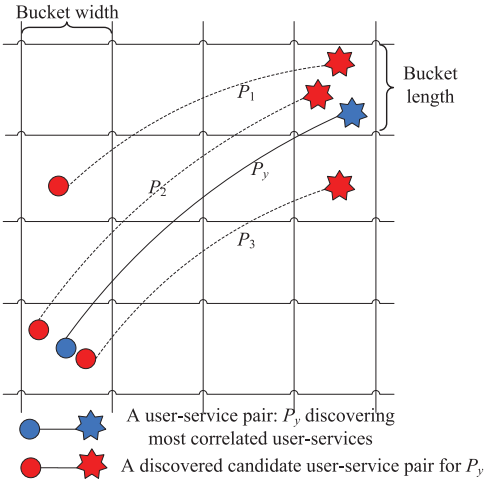

Fig. 4. Discover $GS(P)$ for P.

$\mathbf{z}_i$ ($1 \leq i \leq M$, where M is the number of other collected temporal QoS sequences). This reduces the search space for discovering the most correlated temporal QoS sequences for $\mathbf{y}$.

Let P, which contains user u and service v, have a temporal sequence $\mathbf{y}$. To find the most correlated temporal QoS sequences for $\mathbf{y}$, we first retrieve a set of spatially similar user-service pairs $GS(P)$. In order to efficiently obtain $GS(P)$, we build a grid representation based on the geographical map and divide the map into multiple buckets, as shown in Figure 4. The length and width of each bucket are represented by differences in longitude and latitude, respectively. As in Wang et al. [2014], we empirically set the bucket length to 0.1156 km and bucket width to 0.1491 km. Then, we map u, v, and each user u_i and service v_i of other temporal QoS sequences to a bucket, which are represented by $u.bucket$, $v.bucket$, $u_i.bucket$, and $v_i.bucket$, respectively. Note that we randomly select a set of user-service pairs from the repository when there is no user/service found in the same bucket. For each u_i and v_i, if $u.bucket$ equals $u_i.bucket$ or $v.bucket$ equals $v_i.bucket$, we calculate the spatial similarity between each selected P_i and P and select the top K' user-service pair P_i as the discovered elements in $GS(P)$. Furthermore, we calculate the correlation coefficient between $\mathbf{y}$ and each $\mathbf{z}_i$ corresponding to the user-service pair P_i in $GS(P)$ and select the top K most correlated user-service pairs P_js with the highest correlation coefficients as well as their corresponding temporal QoS sequence $\mathbf{x}_j$ ($1 \leq j \leq K$). Note that the correlation coefficients should be greater than zero.

It is also interesting to compare the prediction accuracy using the exact top K user-service pairs using the approximate method. To this end, we conduct experiments on the QoS repository [Zhang et al. 2011] with λ while keeping the remaining parameters fixed. In the experiment, our proposed method for the exact top 20 user-service pairs obtains the MAE result 0.8786, 0.8837, 0.8965, and 0.9135, when λ is set to 0.2, 0.4, 0.6, and 0.8, respectively. On the other hand, our proposed method using the approximate top pairs by bucket achieves the MAE result 0.8829, 0.8908, 0.9040, and 0.9203. It can be clearly seen that the performance drop of the approximation method is negligible, usually less than 1%. However, the approximate method is essentially faster than exact search. Therefore, spatial information is very effective in reducing the searching range for a temporal QoS sequence.

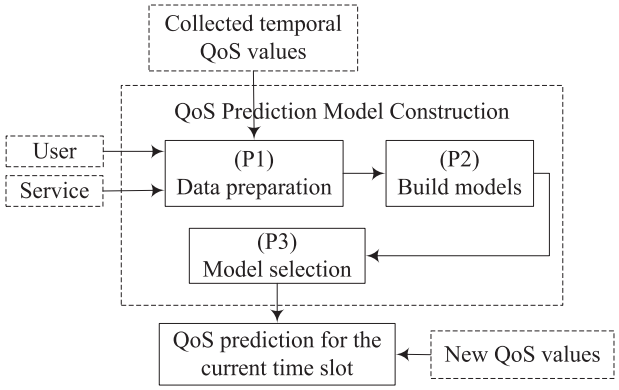

Fig. 5. Spatial-temporal QoS prediction model development.

3.5. Implementation Details

From the preceding, we present a practical development approach to the spatial-temporal QoS prediction model for time-aware Web service recommendation, as shown in Figure 5. The whole scheme consists of the following three steps:

(P1) Data preparation: Before retrieving the most correlated temporal QoS sequences for $\mathbf{y} = (y_1, \ldots, y_{n-1})^T$ from $\mathbf{z}_i = (z_{i1}, \ldots, z_{in})^T$ $(1 \leq i \leq M)$, we need to calculate the correlation coefficients between $\mathbf{y}$ and each $\mathbf{z}_i$. Note that the dataset may contain some invalid values with zeros in $\mathbf{y}$ and $\mathbf{z_i}$, which are dealt with as follows:

(1) If z_{in} is invalid, then $\mathbf{z}_i$ cannot be retrieved as the most correlated temporal QoS sequence for $\mathbf{y}$;
(2) If y_k $(1 \leq k \leq n-1)$ is invalid, we search for y_l in $\mathbf{y}$ such that y_l is valid, $k-l > 0$, $l \geq 1$, and $k-l$ is the minimum. Then, we use y_l to replace y_k. Note that we assume y_1 is valid in our proposed approach;
(3) If z_{ik} $(1 \leq k \leq n-1)$ is invalid, we search for z_{il} in z_{ik} such that z_{il} is valid, $k-l > 0, l \geq 1$, and $k-l$ is the minimum. Then, we use z_{il} to replace z_{ik}. If such z_{il} cannot be discovered, $\mathbf{z}_i$ cannot be discovered as the most correlated temporal QoS sequence for $\mathbf{y}$.

After dealing with invalid QoS values, the most correlated temporal QoS sequences for $\mathbf{y}$ can be obtained in $\mathbf{z}_i$ $(1 \leq i \leq M)$ according to the method described in Section 3.4.

(P2) Model Construction: If the K most correlated temporal QoS sequences for $\mathbf{y}$ can be retrieved, we choose them as the examples to train the regression models for $\mathbf{y}$. The next step is to estimate the unknown parameters in the models. Since we apply the least squares or Lasso method to estimate the unknown parameters $\mathbf{w}$ in the regression models, we name our approaches "LS" or "Lasso." We apply either least squares or Lasso to estimate $\mathbf{w}$ as discussed in Section 3.2. Note that ϵ can also be modeled by the ARMA(p,q) model [Amin et al. 2012] if the sequence of residuals is not a white noise series. In this article, we simply employ the Gaussian noise assumption.

(P3) Model Selection: Practically, although we can construct several regression models between $\mathbf{y}$ and $\mathbf{x}_j$ $(1 \leq j \leq K)$ using the different algorithms, we often need to select one best model. We apply the standard error of estimation and Schwartz's Bayes Criterion (SBC) [Box and Jenkins 1976] as our criterion for best model selection. We use $\text{SBC} = -2ln(l) + ln(k)$, where l is the maximized value of the likelihood function for the estimated model, and k is the total number of parameters in the regression model. The best model is the one with minimum standard error and minimum SBC value.

Once the best model is selected, we can predict the QoS at current time slot t_n: y_n based on $\mathbf{y}$, $\mathbf{x_j}$ $(1 \leq j \leq K$, containing $x_{jn})$, and the estimated $\mathbf{w}$.

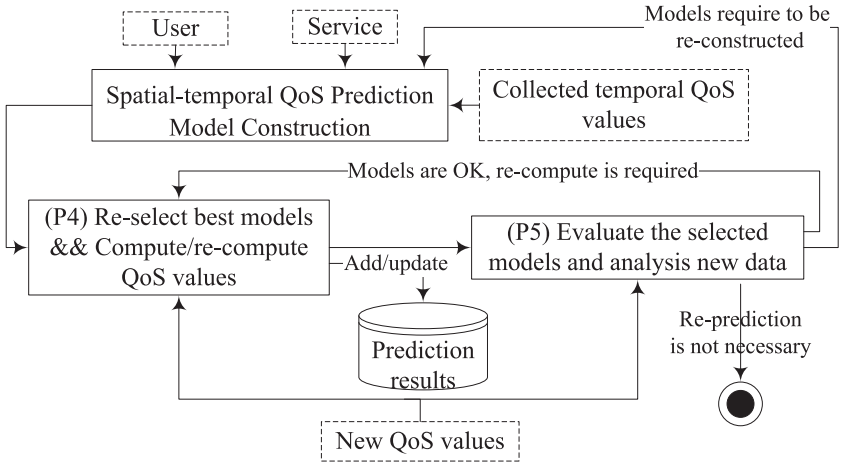

Fig. 6. The spatial-temporal QoS prediction framework: Geotime.

4. SPATIAL-TEMPORAL QOS PREDICTION AND RE-PREDICTION FRAMEWORK

Based on the proposed spatial-temporal QoS prediction model development approach, we present a spatial-temporal QoS prediction and reprediction framework, Geotime, for time-aware Web service recommendation, as illustrated in Figure 6.

Geotime can perform QoS prediction and reprediction continuously for the current time slot t_n. After the predicted QoS value for t_n: y_n in $\mathbf{y}$ is calculated (P4) based on the regression model, the new temporal QoS values in $\mathbf{x}_j$ $(1 \leq j \leq K)$ can be collected during the current time slot. Therefore, the new characteristics of $\mathbf{y}$ for prediction may be demonstrated. This requires us to evaluate whether it is necessary to reconstruct a new regression model for updating the predicted QoS values. Once a regression model is constructed between $\mathbf{x}_j$ $(1 \leq j \leq K)$ and $\mathbf{y}$, the framework stores $\mathbf{x}_j$ $(1 \leq j \leq K)$ as well as $\max S(\mathbf{y}, \mathbf{x}_j)|_{j=1}^{K}$ for $\mathbf{x}_j$ and $\mathbf{y}$. Then, the following rules are defined and considered for rebuilding the model:

Rule 1: If new QoS values are collected in $\mathbf{x}_j$ corresponding to P_j, or new values are collected in $\mathbf{y}$ corresponding to P, we calculate the correlation coefficient between $\mathbf{x}_j^*$ and $\mathbf{y}^*$ $S(\mathbf{y}^*, \mathbf{x}_j^*)$, where $\mathbf{x}_j^*$ is the updated $\mathbf{x}_j$ and $\mathbf{y}*$ is the updated $\mathbf{y}$. If $S(\mathbf{y}^*, \mathbf{x}_j^*) > \frac{\max S(\mathbf{y}, \mathbf{x}_j)|_{j=1}^{K}}{\tau}$ $(0 < \tau \leq 1)$ and $S(\mathbf{y}^*, \mathbf{x}_j^*) > r_{\min}$ $(0 < r_{\min} < 1)$, a new LRM will be constructed between $\mathbf{y}^*$ and $\mathbf{x}_j^*$ $(1 \leq j \leq K)$ to replace the current model. τ and $r_{\min}$ are parameters to adjust the frequency for rebuilding the regression model.

Rule 2: If a new temporal QoS sequence $\mathbf{x}_k$ corresponding to P_k is collected $(simS(P, P_k) < L$ km), such that $S(\mathbf{y}^*, \mathbf{x}_k^*) > \frac{\max S(\mathbf{y}, \mathbf{x}_k)|_{j=1}^{K}}{\tau}$ and $S(\mathbf{y}^*, \mathbf{x}_k^*) > r_{\min}$, a new LSM will be constructed between $\mathbf{y}^*$ and a set of new temporal QoS sequences containing $\mathbf{x}_k^*$ to replace the current model.

Rule 3: If the model rebuilding process is unnecessary and the most recent QoS values are in $\mathbf{x}_j$ $(1 \leq j \leq K)$, then we first reselect the best models for prediction based on these newly collected values. After that, we recompute y_n based on the current regression model and the newly collected values in $\mathbf{x}_j$ in P4.

The model rebuilding process is based on monitoring the newly collected QoS values as time passes. Therefore, Geotime can be used to predict and repredict QoS values for the current time slot continuously for a user-service pair. By taking advantage of

Table II. Statistics of Valid Web Service QoS Values

Statistics	Response time
Scale	(0-20] s
Mean all values	3.1773 s
Num. of users	142
Num. of Web services	64
Num. all values	30,170,567
Num. of temporal sequences	483,235
Mean SD of values in each sequence	2.0115 s
Mean changing rate of all values	4.7051

the repredicted QoS values for the current time slot, the recommended services with optimal predicted QoS values can be updated continuously.

5. EXPERIMENTS

In this section, we conduct comprehensive studies on the performance of our proposed spatial-temporal QoS prediction approach to time-aware Web service recommendation.

5.1. Experimental Setup

In our experiments, we employ a real-world Web service QoS performance repository [Zhang et al. 2011] to evaluate the proposed approach. The repository contains a large number of temporal response time sequences collected from 142 distributed computers located in 57 countries from PlanetLab[1] to 4,532 distributed services all around the world. Each sequence contains a set of temporal QoS values collected from a computer (user) for a service with at most 64 QoS values collected once after a time interval in a time slot. Each time slot lasts for 15 minutes, and the time interval between two adjacent time slots is 15 minutes. All the sequences are collected concurrently, lasting for 16 hours. Therefore, a $142 \times 4{,}532 \times 64$ user-service-time matrix is constructed containing a particular response time value from the QoS invocation records in each of its positions. Some QoS values in the matrix are invalid; these are marked as zero. If the response time is larger than 20s, it is recorded as 20s in the datasets.

Table II summarizes the statistics of all the valid Web service QoS values in the matrix. We treat the temporal sequence as an invalid record if there are no less than 12 collected QoS values at different time slots for a user-service pair. Also, Figure 7 shows the distribution of the response time and changing rate of all valid QoS values. We can observe that most response time values are between 0.2s and 0.8s, and most of the changing rates are below 0.1.

The ARIMA method is employed as the baseline method for evaluating the performance of the proposed spatial-temporal QoS prediction approach. Note that the ARIMA model should satisfy several assumptions including serial dependency, stationary, normality, and invertibility [Box and Jenkins 1976]. To this end, we use some statistical tests to check for these assumptions. Specifically, the QLB test [Box and Jenkins 1976] is employed to check $\mathbf{y}$'s serial dependency, the ADF test to check its stationarity, and the KPSS test [Kwiatkowski et al. 1992] to check its normality. If these assumptions are not satisfied, we simply average the last three observed values in the QoS sequence as the unknown values to be predicted.

We implemented the UPCC [Breese et al. 1998], IPCC [Sarwar et al. 2001], and WSRec [Zheng et al. 2009] methods for comparison, which are promising for conventional QoS prediction. UPCC is a user-based prediction algorithm using PCC [Breese et al. 1998]. When employing UPCC to predict a QoS value y for the current time slot, it uses

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

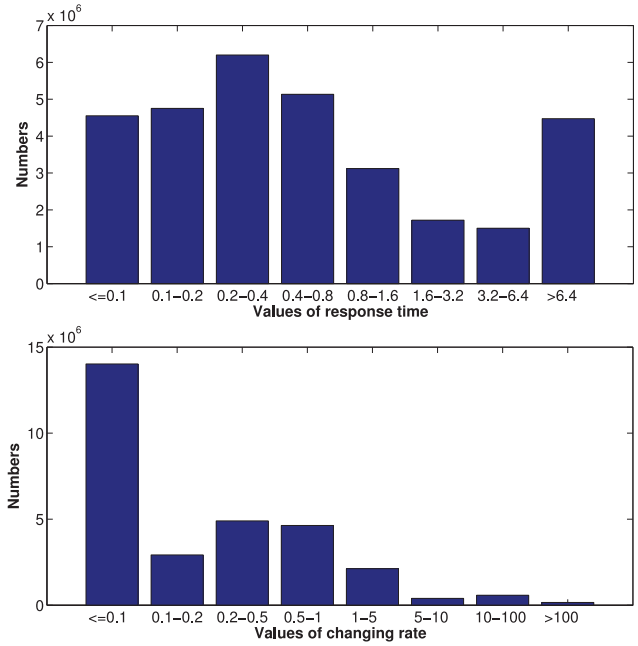

Fig. 7. QoS value distributions.

all other collected QoS values at the current time slot for prediction. On the other hand, IPCC is an item-based prediction algorithm using PCC. WSRec [Zheng et al. 2009] is a hybrid CF algorithm that combine UPCC with IPCC.

For the temporal QoS prediction, we also include the simple solution of average method (AVG), which simply averages the three valid QoS values at the most recent time slots in $\mathbf{y}$ to predict the QoS value for the temporal QoS sequence.

Furthermore, conventional CF techniques are mainly effective for the stationary estimation, which may have poor performance in a dynamic environment. We can improve them by fusing their prediction results with the temporal average. Specifically, UPCC* combines UPCC with AVG through linear interpolation. If we employ y_{UPPC} to represent the predicted value using UPCC, y_{AVG} to represent the predicted value using AVG, and y_{UPCC*} to represent the predicted value using UPCC*, then,

$$y_{UPPC*}(y_n) = w \cdot y_{UPPC} + (1 - w) \cdot y_{AVG}.$$

In all our experiments, w is set to 0.5. Similarly, we denote IPCC* and WSRec* as the improved version for IPCC and WSRec, respectively.

To evaluate the performance of our proposed QoS prediction approach, we employ Mean Absolute Error (MAE) [Zheng et al. 2011] and Root Mean Square Error (RMSE) [Zheng et al. 2011] as the evaluation metrics. The metric MAE is defined as:

$$MAE = \frac{\sum_i |\hat{y_n} - y_n|}{N}, \tag{6}$$

and RMSE is defined as:

$$RMSE = \sqrt{\frac{\sum_i (\hat{y_n} - y_n)^2}{N}}, \tag{7}$$

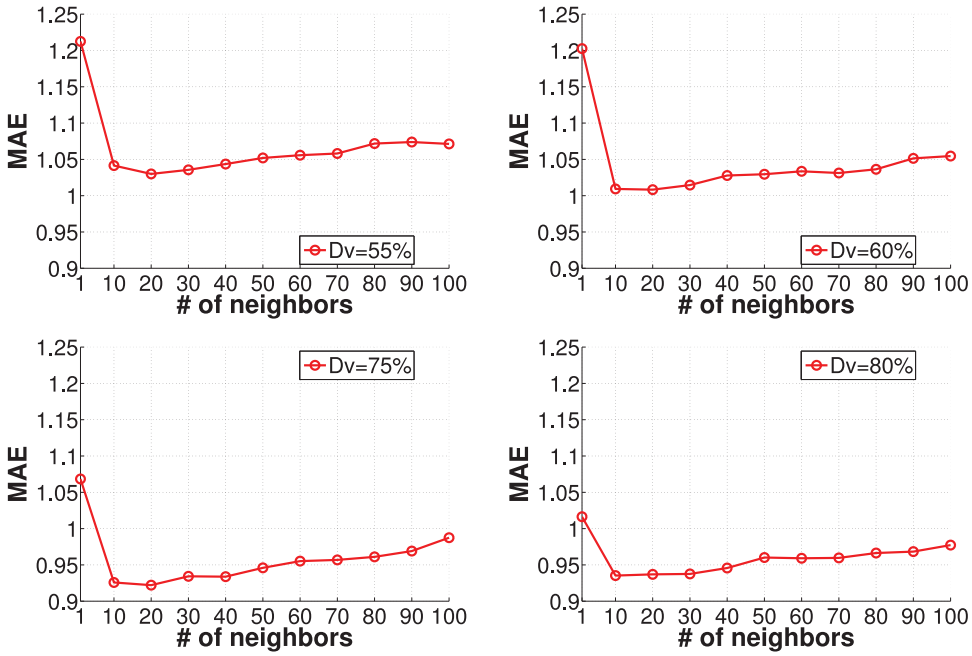

Fig. 8. MAE. K with different response time value densities.

where y_n is the predicted QoS value of a QoS sequence $\mathbf{y}$ for the current time slot t_n, $\hat{y}_n$ denotes the actual QoS value of $\mathbf{y}$ for t_n, and N is the number of predicted QoS values.

5.2. Parameters Settings

First, we study the important parameters for the proposed spatial-temporal QoS predictions using Lasso, including the number of most correlated neighbors K and the regularization parameter λ.

5.2.1. Number of Most Correlated Neighbors K. For the proposed spatial-temporal QoS prediction approach, we need to select the K most correlated temporal QoS sequences as the representative examples. To this end, we randomly select 10,000 test cases and perform QoS prediction using Lasso with various values of K. We calculate the prediction accuracy using MAE within different response value densities with each value of K, where the regularization coefficient λ is fixed to 0.2. Therefore, we can to some extent determine the best value of K for our approach with different densities.

As shown in Figure 8, the proposed Lasso method achieves a better prediction accuracy with the best value of K. For data with 100% temporal response time density, the best number of neighborhoods should be about 10 to 20 for optimizing the prediction accuracy of our proposed QoS prediction approach using Lasso by varying the density of response time values. Note that the proposed method is degenerated to 1-Nearest Neighbor search when K is set to 1. Therefore, the prediction performance is poor. The representation capability of Lasso increases with more exemplar data so that the prediction error decreases with the growing number of the selected nearest neighbors K. However, such a model may have overfitting issues when the total number of selected examples becomes larger because the learned regression model aims to minimize the fitting error on the training data rather than on the testing data. Overfitting on the training data may lead to poor generalization capability on the new test data.

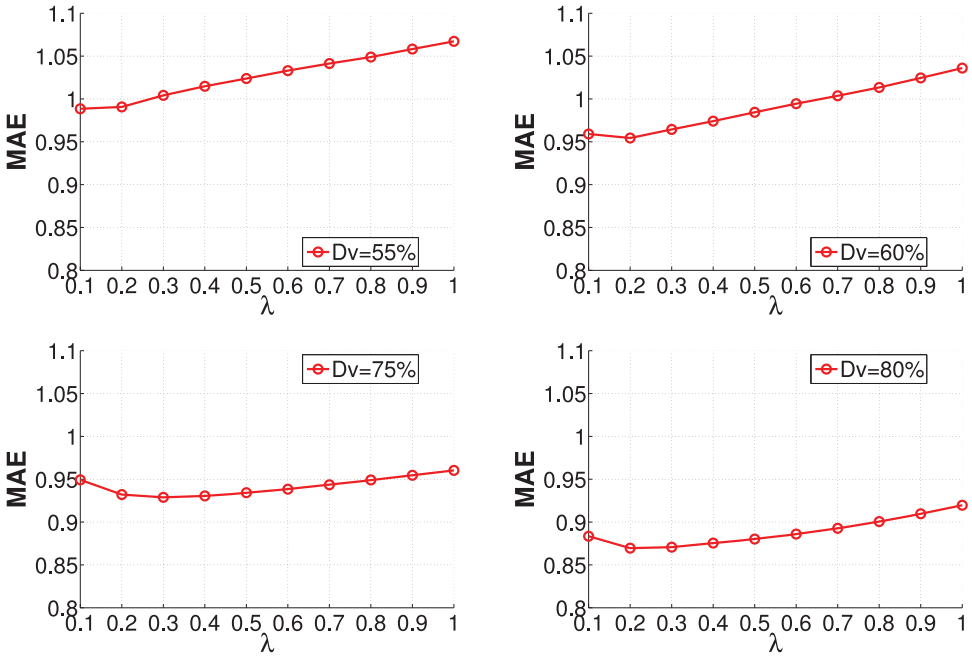

Fig. 9. MAE. λ with different response time value densities.

5.2.2. Regularization Coefficient λ for Lasso. In this experiment, we intend to find the proper value for the regularization coefficient λ for the proposed method using Lasso. We fix the number of the most correlated neighbors K to 20. Moreover, we randomly select 10,000 test cases and conduct temporal QoS predictions with various λ. The Lasso optimization is solved by the method described in Mairal et al. [2010].

As shown in Figure 9, the best value of λ is in the range of [0.1,0.3] for the Lasso regression with various density of response time values when the temporal response time density remains 100%.

5.3. Performance Comparisons on QoS Prediction

To conduct comprehensive experiments on the QoS prediction for time-aware Web service recommendation, we first randomly remove a number of temporal QoS sequences in the user-service-time dataset. If $T\%$ temporal QoS sequences have been removed, we denote the temporal QoS density as $(100 - T)\%$, represented by $D_t = (100 - T)\%$. Then, we randomly remove 20% QoS values in all temporal QoS sequences. We vary the temporal QoS density ranges from 10% to 60% and randomly create more than 50,000 different test cases for each density. In each test case, we randomly select a removed QoS value. In addition, there are at least 12 QoS values left before t_n in the matrix in order to facilitate ARIMA model construction. For each test case, we perform experimental evaluations comparing prediction algorithms and compute their predicted values. Then we evaluate their prediction accuracies using the two metrics. For our presented spatial-temporal approaches, we set the bucket length to 0.1156 and bucket width to 0.1491, as shown in Figure 4. K' is set to 400. For our presented Lasso method ($K = 20$), the parameter λ greatly affects prediction accuracy. In our experiments, we empirically set λ to 0.1.

Next, we randomly remove a number of QoS values in all temporal QoS sequences in the initial user-service-time dataset. If $V\%$ QoS values have been removed in each

Table III. Performance Comparisons of Prediction Approaches on Response Time with Reduced Temporal QoS Sequences; Δ Represents the Performance Increment Compared to Baseline ARIMA Method

Approaches		$D_v = 55\%$		$D_v = 60\%$		$D_v = 65\%$	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
AVG		1.159	3.206	1.159	3.206	1.159	3.206
UPCC		1.470	3.034	1.467	3.027	1.464	3.019
UPCC*		1.252	2.775	1.251	2.773	1.244	2.759
IPCC		1.396	2.951	1.388	2.937	1.384	2.926
IPCC*		1.221	2.742	1.216	2.736	1.209	2.724
WSRec		1.391	2.951	1.384	2.937	1.381	2.926
WSRec*		1.220	2.747	1.215	2.740	1.208	2.727
ARIMA		1.028	2.986	1.028	2.986	1.028	2.986
LS	$K = 1$	1.111	2.946	1.066	2.881	1.001	2.729
	Δ	−8.1%	1.3%	−3.7%	3.5%	2.6%	8.6%
	$K = 20$	1.229	3.294	1.187	3.226	1.111	3.067
	Δ	−19.6%	−10.3%	−15.5%	−8.0%	−8.1%	−2.7%
Lasso	$K = 1$	1.097	2.907	1.055	2.844	0.991	2.689
	Δ	−6.7%	2.7%	−2.6%	4.8%	3.6%	10.0%
	$K = 20$	1.012	2.784	0.969	2.705	0.902	2.533
	Δ	1.6%	6.8%	5.7%	9.4%	12.3%	15.2%
Approaches		$D_v = 70\%$		$D_v = 75\%$		$D_v = 80\%$	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
AVG		1.159	3.206	1.159	3.206	1.159	3.206
UPCC		1.466	3.027	1.464	3.026	1.466	3.032
UPCC*		1.244	2.759	1.242	2.763	1.242	2.753
IPCC		1.378	2.928	1.374	2.923	1.374	2.925
IPCC*		1.206	2.720	1.202	2.717	1.200	2.714
WSRec		1.376	2.928	1.372	2.923	1.372	2.925
WSRec*		1.206	2.724	1.202	2.721	1.200	2.715
ARIMA		1.028	2.986	1.028	2.986	1.028	2.986
LS	$K = 1$	1.020	2.793	1.006	2.774	1.010	2.789
	Δ	0.8%	6.5%	2.1%	7.1%	1.8%	6.6%
	$K = 20$	1.101	3.035	1.102	3.062	1.110	3.076
	Δ	−7.1%	−1.6%	−7.2%	−2.6%	−8.0%	−3.0%
Lasso	$K = 1$	1.008	2.743	0.997	2.735	1.002	2.751
	Δ	2.0%	8.1%	3.0%	8.4%	2.5%	7.9%
	$K = 20$	0.900	2.550	0.890	2.538	0.902	2.567
	Δ	12.5%	14.6%	13.4%	15.0%	12.3%	14.0%

sequence, we denote the QoS value density as $(100 - V)\%$, represented by $D_v = (100 - V)\%$. We also vary the QoS value density with ranges from 55% to 80%, randomly creating more than 100,000 different test cases for each density.

Table III presents the MAE and RMSE results of different prediction approaches on response time when the temporal response time densities vary from 10% to 60%. Moreover, Table IV presents the prediction results when the response time value densities vary from 55% to 80%. From these results, we have the following observations:

(1) UPCC, IPCC, and WSRec are much worse in prediction accuracies than other approaches since they do not consider the historical temporal QoS values in the sequence;

(2) The prediction accuracy of ARIMA is slightly better than the AVG method when the density of response time value is high. The prediction accuracies of ARIMA and AVG are better than those of UPPC*, IPPC*, and WSRec*;

Table IV. Performance Comparisons of Prediction Approaches on Response Time with Various QoS Value Densities; Δ Represents the Performance Increment Compared to Baseline ARIMA Method

Approaches		$D_v = 55\%$		$D_v = 60\%$		$D_v = 65\%$	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
AVG		1.172	3.226	1.167	3.219	1.164	3.214
UPCC		1.470	3.034	1.467	3.027	1.464	3.019
UPCC*		1.252	2.775	1.251	2.773	1.244	2.759
IPCC		1.396	2.951	1.388	2.937	1.384	2.926
IPCC*		1.221	2.742	1.216	2.736	1.210	2.724
WSRec		1.391	2.951	1.384	2.937	1.381	2.926
WSRec*		1.220	2.747	1.215	2.740	1.208	2.728
ARIMA		1.201	3.332	1.160	3.285	1.089	3.175
LS	$K = 1$	1.193	3.119	1.162	3.076	1.110	2.999
	Δ	0.7%	6.4%	−0.2%	6.4%	−1.9%	5.5%
	$K = 20$	1.352	3.609	1.298	3.492	1.223	3.344
	Δ	−12.6%	−8.3%	−11.9%	−6.3%	−12.3%	−5.3%
Lasso	$K = 1$	1.184	3.112	1.153	3.053	1.101	2.960
	Δ	1.4%	6.6%	0.6%	7.1%	−1.1%	6.8%
	$K = 20$	1.000	2.822	0.984	2.781	0.954	2.723
	Δ	16.8%	15.3%	15.2%	15.3%	12.4%	14.2%
LS*	$K = 1$	1.427	3.448	1.331	3.213	1.321	3.215
	Δ	−18.8%	−3.5%	−14.7%	2.19%	−21.3%	−1.3%
Lasso*	$K = 1$	1.420	3.494	1.371	3.359	1.315	3.306
	Δ	−18.2%	−4.9%	−18.2%	−2.3%	−20.8%	−4.1%
	$K = 20$	1.240	3.338	1.208	3.250	1.176	3.148
	Δ	−3.3%	−0.2%	−4.1%	1.1%	−8.0%	0.9%
Approaches		$D_v = 70\%$		$D_v = 75\%$		$D_v = 80\%$	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
AVG		1.162	3.211	1.160	3.209	1.159	3.206
UPCC		1.466	3.027	1.464	3.026	1.467	3.032
UPCC*		1.244	2.759	1.242	2.763	1.242	2.753
IPCC		1.378	2.928	1.374	2.923	1.372	2.925
IPCC*		1.206	2.720	1.202	2.717	1.200	2.714
WSRec		1.376	2.928	1.372	2.923	1.372	2.925
WSRec*		1.206	2.724	1.202	2.721	1.200	2.716
ARIMA		1.087	3.042	1.051	3.030	1.028	2.986
LS	$K = 1$	1.062	2.868	1.035	2.820	0.991	2.756
	Δ	2.3%	5.7%	1.5%	6.9%	3.6%	7.7%
	$K = 20$	1.169	3.198	1.134	3.110	1.085	3.013
	Δ	−7.5%	−5.1%	−7.9%	−2.6%	−5.5%	−0.9%
Lasso	$K = 1$	1.055	2.847	1.027	2.800	0.982	2.729
	Δ	3.0%	6.4%	2.3%	7.7%	4.4%	8.6%
	$K = 20$	0.922	2.637	0.909	2.607	0.893	2.572
	Δ	15.2%	13.3%	13.5%	14.0%	13.1%	13.9%
LS*	$K = 1$	1.279	3.124	1.311	3.204	1.203	3.004
	Δ	−17.7%	−2.7%	−24.7%	−5.7%	−17.0%	−0.6%
Lasso*	$K = 1$	1.324	3.292	1.285	3.204	1.201	3.082
	Δ	−21.8%	−8.2%	−22.3%	−5.7%	−16.8%	−3.2%
	$K = 20$	1.214	3.225	1.160	3.111	1.112	3.026
	Δ	−11.7%	−6.0%	−10.4%	−2.7%	−8.2%	−1.3%

(3) Our proposed Lasso method outperforms other prediction approaches in prediction accuracy in most cases. When the temporal response time density varies from 10% to 60% and the response time value density is high (80%), Lasso ($K = 1$) can obtain as high as 9% improvements in prediction accuracy compared with the traditional ARIMA, whereas Lasso ($K = 20$) can achieve as high as 15% improvements. When the response time value density varies from 55% to 80%, Lasso ($K = 1$) can obtain as high as 8% improvements in prediction accuracy compared to the traditional ARIMA, whereas Lasso ($K = 20$) achieves as high as 16% improvements;

(4) When the density of temporal response time value increases from 10% to 60%, the prediction accuracy of our presented Lasso method will improve. Moreover, the advantages of the Lasso method ($K = 20$) compared to ARIMA will be more obvious. When the response time value density increases from 55% to 80%, the prediction accuracy of our Lasso ($K = 1$) will improve, and its advantages compared to ARIMA will reach about 8%; the prediction accuracy of our Lasso ($K = 20$) will also improve, and its advantages compared to ARIMA will be about 13% to 16%. When the density is high, more correlated temporal QoS sequences are fetched, which can be employed to construct a more precise LRM using our approaches; therefore, the dynamic characteristics of the temporal response time sequence to be predicted is more accurately captured.

(5) To verify the effectiveness of using the geo-locations of users and services in discovering the most correlated temporal QoS sequences compared to the QoS sequence to be predicted $\mathbf{y}$, we design two other temporal QoS prediction approaches named "LS*" and "Lasso*." LS* and Lasso* are very similar to LS and Lasso, except in the process of finding the spatially similar user-service pairs $GS(P)$, where the K' temporal QoS sequences are randomly selected from the whole dataset. We compared the prediction accuracy of Lasso* to Lasso under various response time value densities, and similar experiments were also conducted for LS* and LS. As shown in Table IV, the prediction accuracy of our proposed Lasso method is much better than that of Lasso*, and similar results can also be found in comparing LS* and LS. This demonstrates that it is very important to incorporate the spatial information of users and services for boosting prediction performance.

5.4. Performance Comparisons on QoS Prediction with Sudden Changes

This experiment intends to analyze the prediction accuracy of all competing QoS prediction models in predicting sudden change QoS (response time) values, as discussed in Section 1. We randomly remove a number of temporal QoS sequences and a number of QoS values in the remaining sequences of our initial user-service-time dataset. Moreover, we set the temporal response time density to 50% and the density of response time values to 70% in the dataset as our collected QoS values. Therefore, the overall QoS data density is low. Then, we vary the proportion of sudden change values denoted by D_s and prepare 12 different sets of test cases for each proportion. In each set, we randomly select a number of sudden change QoS values and a number of QoS values that are not sudden change values. Each set contains 1,000 different test cases. Then, we conduct some competing prediction methods for the corresponding sudden change value proportion based on each set of test cases. Table V presents the statistics of prediction accuracy on these test cases when the proportion of sudden change values varies from 0% to 100%.

It can be clearly seen that LS ($K = 1$), Lasso ($K = 20$), and ARIMA have similar effects in prediction accuracy when the sudden change value proportion is very low. If the sudden change value proportion equals the proportion in the initial collected response time sequences (10.63%), LS ($K = 1$) and Lasso ($K = 20$) achieve about 4% to 12% improvements in prediction accuracy compared to ARIMA. When the sudden change value proportion increases from 10% to 100%, the improvements of LS ($K = 1$)

Table V. Performance Comparisons of Prediction Approaches on Response Time with Different Proportions of Sudden Change QoS values; Δ Represents the Performance Increment Compared to Baseline ARIMA Method

Approaches		$D_s = 0\%$		$D_s = 10\%$		$D_s = 20\%$		$D_s = 30\%$	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
AVG		0.780	2.269	1.120	3.081	1.351	3.504	1.664	4.129
UPCC*		1.467	3.202	1.712	3.732	1.909	4.091	2.196	4.642
IPCC*		1.018	2.356	1.306	3.041	1.510	3.424	1.775	3.953
WSRec*		1.051	2.389	1.351	3.142	1.564	3.560	1.862	4.175
ARIMA		0.776	2.235	1.095	3.008	1.337	3.475	1.616	4.050
LS	$K = 1$	0.749	2.216	0.973	2.891	1.169	3.258	1.392	3.730
	Δ	3.5%	0.8%	11.1%	3.9%	12.6%	6.2%	13.9%	7.9%
	$K = 20$	1.320	3.739	1.575	4.118	1.665	4.187	1.878	4.508
	Δ	−70.1%	−67.3%	−43.8%	−36.9%	−24.5%	−20.5%	−16.2%	−11.3%
Lasso	$K = 20$	0.771	2.293	1.005	2.883	1.201	3.256	1.432	3.680
	Δ	0.7%	−2.6%	8.2%	4.2%	10.2%	6.3%	11.4%	9.1%
Approaches		$D_s = 40\%$		$D_s = 50\%$		$D_s = 60\%$		$D_s = 70\%$	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
AVG		1.937	4.568	2.172	4.876	2.461	5.171	2.733	5.424
UPCC*		2.455	5.018	2.614	5.233	2.852	5.481	3.039	5.709
IPCC*		2.019	4.344	2.218	4.608	2.497	4.912	2.756	5.216
WSRec*		2.126	4.595	2.330	4.875	2.603	5.176	2.835	5.442
ARIMA		1.919	4.560	2.165	4.883	2.456	5.236	2.734	5.522
LS	$K = 1$	1.538	3.894	1.698	4.104	1.940	4.400	2.198	4.647
	Δ	19.9%	14.6%	21.6%	15.9%	21.0%	16.0%	19.6%	15.9%
	$K = 20$	2.111	4.796	2.302	4.963	2.369	4.890	2.587	5.080
	Δ	−10.0%	−5.2%	−6.3%	−1.6%	3.5%	6.6%	5.4%	8.0%
Lasso	$K = 20$	1.596	3.875	1.697	3.956	1.923	4.235	2.129	4.385
	Δ	16.8%	15.0%	21.6%	19.0%	21.7%	19.1%	22.1%	20.6%
Approaches		$D_s = 80\%$		$D_s = 90\%$		$D_s = 100\%$		$D_s = 10.63\%$	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
AVG		3.009	5.695	3.234	5.898	3.485	6.144	1.160	3.195
UPCC*		3.277	5.986	3.507	6.202	3.715	6.471	1.750	3.822
IPCC*		3.019	5.506	3.213	5.691	3.417	5.880	1.338	3.131
WSRec*		3.090	5.724	3.315	5.942	3.543	6.202	1.388	3.249
ARIMA		3.022	5.834	3.241	6.021	3.451	6.206	1.148	3.164
LS	$K = 1$	2.380	4.833	2.580	5.031	2.742	5.193	1.008	3.014
	Δ	21.3%	17.2%	20.4%	16.5%	20.6%	16.3%	12.2%	4.7%
	$K = 20$	2.793	5.309	2.993	5.536	3.090	5.602	1.627	4.248
	Δ	7.6%	9.0%	7.7%	8.1%	10.5%	9.7%	−41.7%	−34.3%
Lasso	$K = 20$	2.319	4.621	2.538	4.859	2.646	4.947	1.050	3.011
	Δ	23.3%	20.8%	21.7%	19.3%	23.3%	20.3%	8.57%	4.8%

and Lasso ($K = 20$) compared to ARIMA also increase from about 10% to about 20%. These results demonstrate that our approaches maintain obvious advantages in capturing the highly volatile characteristics in temporal QoS sequence, which can increase the overall prediction accuracy of QoS values for the current time slot.

5.5. Performance on Continuous QoS Re-prediction

To analyze the performance of the continuous QoS reprediction mechanism, we simulate an environment where increasing amounts of historical QoS values are collected and accumulated. First, we randomly remove a large number of temporal response time sequences or response time values in each sequence in the initial user-service-time

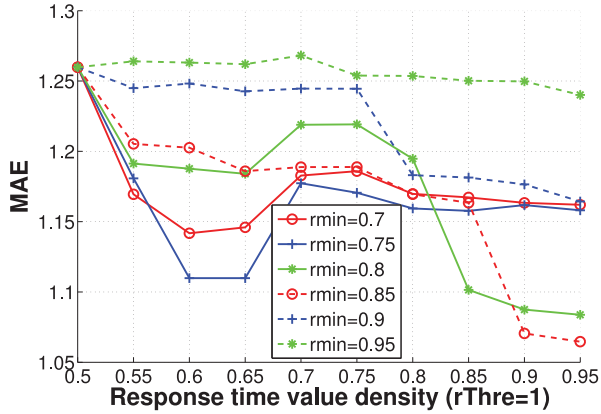

Fig. 10. MAE. response time value density with various $r_{\min}$s.

matrix. Then, we randomly select 168 remaining response time values in the matrix as the response time for the current time slot to be predicted and perform LS ($K = 1$). For simplicity, we only analyze the performance of QoS reprediction conducted by LS ($K = 1$). We randomly let the removed unknown temporal response time sequences or response time values be known one after another. We increase the temporal response time density or response time value density in the matrix. When the matrix density is increased to a certain extent, we perform our QoS reprediction for the predicted response time values, record the newly predicted values, and evaluate the prediction accuracy of all the most recent repredicted response time values for the current time slot to be predicted. Meanwhile, we record the overall model reconstruction times. In our experiment, we first increase the temporal response time density from 10% to 90% with 80% response time value density while conducting our continuous QoS reprediction. Then, we increase the response time value density from 50% to 95% with 100% temporal response time density while using this mechanism.

As shown in the results from Figures 10−17, temporal QoS prediction accuracy for the current time slot increases continuously in general as a result of prediction model reconstruction or QoS reprediction when the matrix density increases. Since the values of τ and $r_{\min}$ affect the model reconstruction frequency as in Section 4, we mainly analyze the values of them on the effects of this mechanism. As demonstrated in our results, when $r_{\min}$ is too large (0.95), the overall prediction accuracy of the QoS values for the current time slot cannot increase significantly with the increase of the response time value density or temporal response time density. When $r_{\min}$ is too small (0.7), on the other hand, the overall prediction accuracy increases in general. However, it may not increase stably. If $r_{\min}$ is too small, the required model reconstruction times increases dramatically with the increase of accumulated historical QoS values, which is very time-consuming, as shown in Figure 11 and Figure 15. Therefore, the best value of $r_{\min}$ is around 0.8 based on our empirical results. Then, we vary the value of τ and analyze the performance of our continuous QoS reprediction mechanism as shown Figures 12 and 16. When τ is large (1.0), the overall prediction accuracy for unknown QoS values does not increase stably with the increase of the response time value density or temporal response time density. Furthermore, if τ is large, the required model reconstruction times increases dramatically with the increase of accumulated historical QoS values. Based on these results, we can determine the best value of τ. Using the best values of both $r_{\min}$ and τ, our QoS reprediction mechanism can improve the prediction

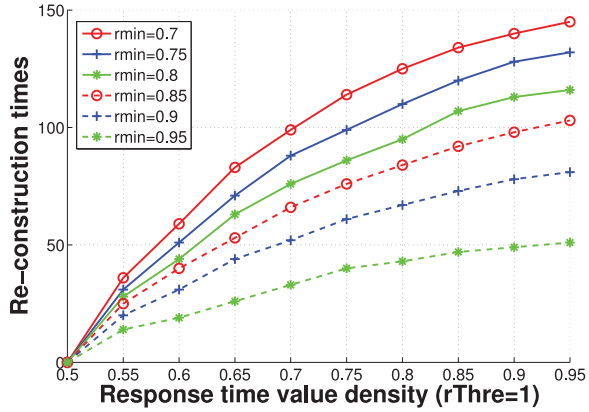

Fig. 11. Reconstruction times. response time value density with various $r_{\min}$s.

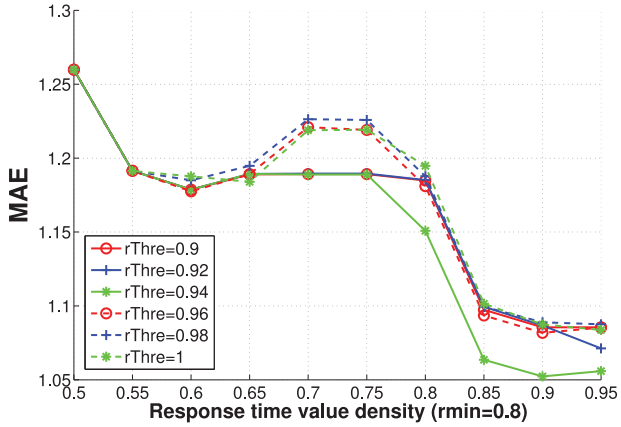

Fig. 12. MAE. response time value density with various τs.

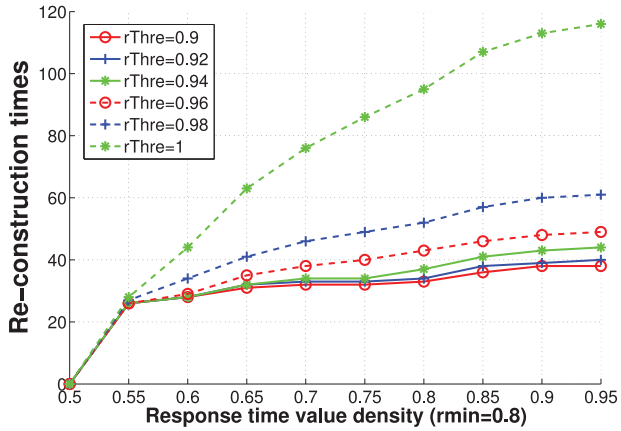

Fig. 13. Reconstruction times. response time value density with various τs.

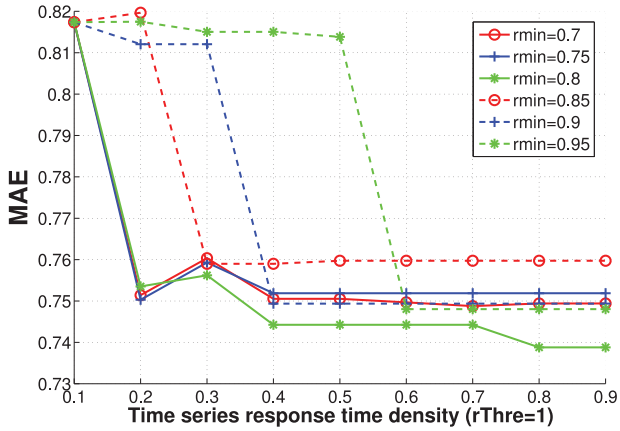

Fig. 14. MAE. temporal response time density with various $r_{\min}$s.

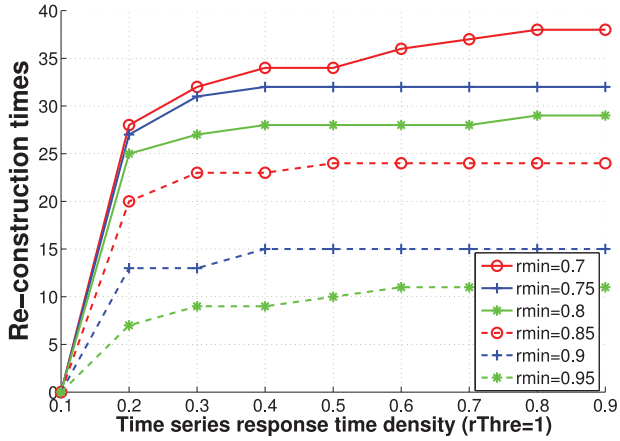

Fig. 15. Reconstruction times. temporal response time density with various $r_{\min}$s.

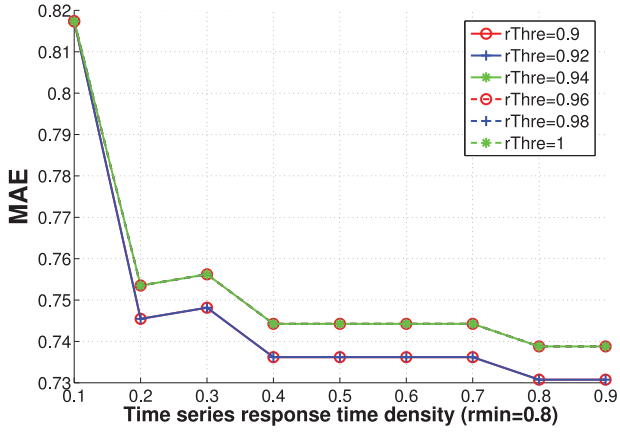

Fig. 16. MAE. temporal response time density with various τs.

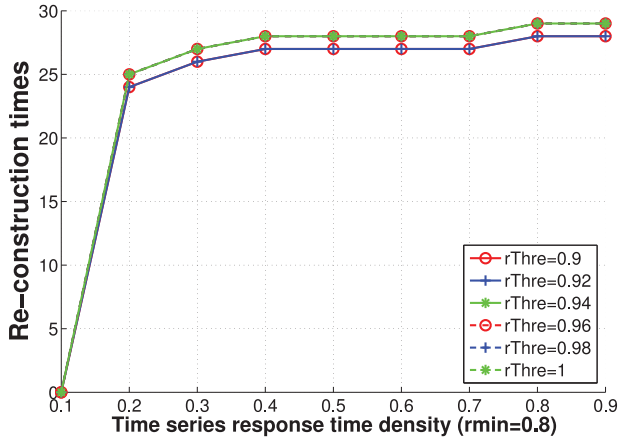

Fig. 17. Reconstruction times. temporal response time density with various τs.

Table VI. Computational Time on 483,235 Sequences

Method	ARIMA	UPCC	IPCC	WSRec	LASSO
Time	2416.2s	80.5s	672.7	773.8s	308.8s

Table VII. Computational Time Under Different K Values

K	10	20	30	40	50
Time	304.1s	412.5s	321.5s	327.5s	336.1s

accuracy consistently with the increasing number of accumulated historical QoS values while requiring less computational time to rebuild the model.

5.6. Computational Time

Finally, we conduct an experiment to analyze the efficiency of various method, including ARIMA, UPCC, IPCC, WSRec, and our proposed Lasso approach. The experimental results are summarized in Table VI. It can be clearly seen that the proposed approach is as efficient as the CF method while being much fast than ARIMA.

We also study the efficiency of our method under different K values. The experimental result are summarized in Table VII. It can be clearly seen that the computational time of our proposed method is insensitive to the values of K.

6. CONCLUSION AND FUTURE WORK

In this article, we proposed a novel spatial temporal QoS prediction approach to time-aware Web service recommendation. We formulated the temporal QoS prediction as a generic regression problem, where a zero-mean Laplace prior distribution assumption is made on the residuals of QoS prediction. Lasso regularization was introduced to facilitate the sparse representation of the temporal QoS sequence. Moreover, the geo-locations of end users and services were employed to effectively retrieve the most similar QoS series. The extensive experimental results demonstrated that the proposed approach outperforms the state-of-the-art temporal QoS prediction methods for time-aware Web service recommendation.

Although achieving promising performance, some limitations should be addressed. The current method requires us to retrieve other QoS values at the current time slot, which cannot be applied to forecast future temporal QoS values. In the future, we will

investigate an online algorithm to predict future QoS values based on accumulated data. Moreover, we will study the hierarchical indexing method to improve overall performance for the spatial-temporal QoS prediction

REFERENCES

A. Amin, A. Colman, and L. Grunske. 2012. An approach to forecasting QoS attributes of web services based on ARIMA and GARCH models. In *Proceedings of the IEEE International Conference on Web Services*. 74–81.

G.-E.-P. Box and G.-M. Jenkins. 1976. *Time Series Analysis: Forecasting and Control*. HoldenDay.

J. S. Breese, D. Heckerman, and C. Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence*. 43–52.

X. Cai, M. Bain, A. Krzywicki, W. Wobcke, Y. Kim, P. Compton, and A. Mahidadia. 2010. Learning collaborative filtering and its application to people to people recommendation in social networks. In *Proceedings of the International Conference on Data Mining*. 743–748.

B. Cavallo, M. D. Penta, and G. Canfora. 2010. An empirical comparison of methods to support QoS-aware service selection. In *Proceedings of the 2nd International Workshop on Principles of Engineering Service-Oriented Systems*. 64–70.

W. Chen, J. Chu, J. Luan, H. Bai, Y. Wang, and E. Chang. 2009. Collaborative filtering for orkut communities: Discovery of user latent behavior. In *Proceedings of the International World Wide Web Conference*. 681–690.

X. Chen, Z. Zheng, X. Liu, Z. Huang, and H. Sun. 2013. Personalized qos-aware web service recommendation and visualization. *IEEE Transactions on Service Computing* 6, 1 (2013), 35–47.

Y. H. Chen and E. I. George. 2002. A bayesian model for collaborative filtering. In *Proceedings of the International Workshop on Artificial Intelligence and Statistics*.

M. Deshpande and G. Karypis. 2004. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems* 22, 1 (Jan. 2004), 143–177.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. 2004. Least angle regression. *Annals of Statistics* 32, 2 (2004), 407–499.

M. Godse, U. Bellur, and R. Sonar. 2010. Automating QoS based service selection. In *Proceedings of the IEEE International Conference on Web Services*. 5–10.

S. Gong. 2010. A collaborative filtering recommendation algorithm based on user clustering and item clustering. *Journal of Software* 5, 7 (July 2010), 745–752.

C. Heath. 2011. GeoLiteCity.dat.gz. Retrieved from http://www.maxmind.com/download/geoip/database/.

H. Hooman and P. J. Kennedy. 2009. HDAX: Historical symbolic modelling of delay time series in a communications network. In *Proceedings of the 8th Australasian Data Mining Conference.* 129–137.

M. C. Jaeger, G. Rojec-Goldmann, and G. Muhl. 2004. QoS aggregation for web service composition using workflow patterns. In *Proceedings of the IEEE International Conference on Enterprise Computing*. 149–159.

S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. 2007. A method for large-scale l1-regularized least squares. *IEEE Journal on Selected Topics in Signal Processing* 1, 4 (2007), 606–617.

A. Klein, F. Ishikawa, and S. Honiden. 2012. Towards network-aware service composition in the cloud. In *Proceedings of the International World Wide Web Conference.* 959–968.

D. Kwiatkowski, P. Phillips, P. Schmidt, and Y. Shin. 1992. Testing the null hypothesis of stationarity against the alternative of a unit root. *Journal of Econometrics* 54 (1992), 159–178.

M. Li, J. Huai, and H. Guo. 2009. An adaptive web services selection method based on the QoS prediction mechanism. In *Proceedings of the IEEE/WIC/ACM Internatinoal Joint Conference on Web Intelligence and Intelligent Agent Technologies*. 395–402.

L. Liu, F. Lecue, N. Mehandjiev, and L. Xu. 2010. Using context similarity for service recommendation. In *Proceedings of the International Conference on Semantic Computing*. 277–284.

W. Luo, J. Yin, S. Deng, Y. Li, and Z. Wu. 2012. Collaborative web service QoS prediction with location-based regularization. In *Proceedings of the IEEE International Conference on Web Services*. 24–29.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. 2010. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research* 11 (2010), 19–60.

M. R. McLaughlin and J. L. Herlocker. 2004. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *Proceedings of the Annual International ACM SIGIR Conference* 329–336.

D. A. Menasce. 2002. QoS issues in web services. *IEEE Internet Computing* 6, 6 (Nov.– Dec. 2002), 72–75.

B. N. Miller, I.Albert, S. K. Lam, J. A. Konstan, and J. Riedl. 2003. MovieLens unplugged: Experiences with an occasionally connected recommender system. In *Proceedings of the ACM 2003 International Conference on Intelligent User Interfaces*. 263–266.

W. Rong, K. Liu, and L. Liang. 2009. Personalized web service ranking via user group combining association rule. In *Proceedings of the IEEE International Conference on Web Services*. 445–452.

R. Salakhutdinov and A. Mnih. 2008. Probabilistic matrix factorization. In *NIPS*. 1257–1264.

B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the International World Wide Web Conference.* 285–295.

L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei. 2007. Personalized QoS prediction for web services via collaborative filtering. In *Proceedings of the IEEE International Conference on Web Services*. 9–13.

Y. Shen, J. Zhu, X. Wang, L. Cai, X. Yang, and B. Zhou. 2013. Geographic location-based network-aware QoS prediction for service composition. In *Proceedings of the IEEE International Conference on Web Services*.

P. Singla and M. Richardson. 2008. Yes, there is a correlation: From social networks to personal behavior on the web. In *Proceedings of the International World Wide Web Conference.* 655–664.

H. Sun, Z. Zheng, J. Chen, and M. R. Lyu. 2012. Personalized web service recommendation via normal recovery collaborative filtering. *IEEE Transactions on Service Computing* PP (2012), 1–9.

M. Tang, Y. Jiang, J. Liu, and X. Liu. 2012. Location-aware collaborative filtering for QoS-Based service recommendation. In *Proceedings of the IEEE International Conference on Web Services*. 24–29.

Robert Tibshirani. 1994. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B* 58 (1994), 267–288.

L. H. Ungar and D. P. Foster. 1998. Clustering methods for collaborative filtering. In *Proceedings of the AAAI Workshop on Recommendation Systems*. 114–129.

X. Wang, J. Zhu, and Y. Shen. 2014. Network-aware QoS prediction for service composition using geolocation. *IEEE Transactions on Services Computing* (2014).

L. Yao, Q. Sheng, A. Segev, and J. Yu. 2013. Recommending web services via combining collaborative filtering with content-based features. In *Proceedings of the IEEE International Conference on Web Services*. 42–49.

Q. Yu, Z. Zheng, and H. Wang. 2013. Trace norm regularized matrix factorization for service recommendation. In *Proceedings of the IEEE International Conference on Web Services*. 34–41.

T. Yu, Y. Zhang, and K.-J. Lin. 2007. Efficient algorithms for web services selection with end-to-end QoS constraints. *ACM Transactions on the Web* 1, 1 (May 2007), 1–26.

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

L. Zeng, C. Lingenfelder, H. Lei, and H. Chang. 2008. Event-driven quality of service prediction. In *Proceedings of the 6th International Conference on Service-Oriented Computing*. 147–161.

L.-J. Zhang, J. Zhang, and H. Cai. 2007. *Performance prediction based EX-QoS driven approach for adaptive service composition*. *Springer and Tsinghua University Press* (2007).

Y. Zhang, Z. Zheng, and M. R. Lyu. 2011. WSPred: A time-aware personalized QoS prediction framework for web services. In *Proceedings of the IEEE 22nd International Symposium on Software Reliability Engineering*. 210–219.

V. Zheng, Y. Zheng, X. Xie, and Q. Yang. 2010. Collaborative location and activity recommendations with gps history data. In *Proceedings of the International World Wide Web Conference.* 1029–1038.

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

Z. Zheng, H. Ma, M. R. Lyu, and I. King. 2011. QoS-aware Web service recommendation by collaborative filtering. *IEEE Transactions on Service Computing* 4, 2 (2011), 140–152.

Z. Zheng, Y. Zhang, and M. R. Lyu. 2012. Investigating QoS of real-world web services. *IEEE Transactions on Service Computing* PP, 99 (Nov. 2012), 1.

Received May 2014; revised April 2015; accepted July 2015