Robust Web Service Recommendation via Quantile Matrix Factorization

Rui Zhu*, Di Niu*, and Zongpeng Li†
*Department of Electrical and Computer Engineering, University of Alberta
†Department of Computer Science, University of Calgary

***Abstract*—We study the problem of personalized Quality of Service (QoS) estimation for web services. State-of-the-art methods use matrix factorization or collaborative prediction to estimate web service response times and throughput for each user based on partial measurements collected from past invocations. We point out that in reality, both the response times and throughput of web services follow highly skewed distributions. In this case, the conditional mean QoS estimates generated by traditional matrix completion approaches can be heavily biased toward a few outliers, leading to poor web service recommendation performance. In this paper, we propose the Quantile Matrix Factorization (QMF) technique for web service recommendation by introducing quantile regression into the matrix factorization framework. We propose a novel and efficient algorithm based on Iterative Reweighted Least Squares (IRLS) to solve the QMF problem involving a non-smooth objective function. We further extend the proposed QMF approach to take into account user and service side attributes. Extensive evaluation based on a large-scale QoS dataset has shown that our schemes significantly outperform various state-of-the-art web service QoS estimation schemes in terms of personalized recommendation performance.**

I. Introduction

Cloud computing services, such as Amazon Web Services (AWS) and Microsoft Azure, provide Software as a Service (SaaS) to businesses mainly through the form of web services. Following the service-oriented architecture, a complex web application for a business is built upon a set of loosely coupled abstract functionalities [1]. For example, a travel planning website like Expedia may rely on numerous abstract services provided by airlines, hotels, car rental companies, insurance companies, credit card companies and banks, etc. For each abstract task, there may exist many web services that can provide the same functionality on the World Wide Web (WWW). Therefore, maintaining an up-to-date knowledge of these web services [1]–[3] and ranking them in terms of Quality of Service (QoS) metrics (e.g. in terms of response time, availability, and throughput) play a critical role for cloud service providers to fulfill the Service Level Agreement (SLA) for the applications relying on them.

The web service QoS estimation and ranking problem becomes especially challenging as published web services offered by diverse organizations are growing rapidly on the Internet. *First*, since different web service consumers, including end users and other web services, are distributed in diverse geographical regions worldwide, in addition to the QoS of the web service itself, they may experience different delays and throughput from the same service, depending on the specific network QoS. *Second*, conducting worldwide web service QoS evaluation between all client-service pairs is not free and often infeasible, since web services may charge a fee for invocation and are usually hosted by diverse organizations and companies. *Finally*, as web services are constantly being introduced and updated on the Internet, it is impractical to evaluate all web service candidates for every client in a short period of time, due to the associated overhead and cost.

An emerging approach to personalized web service selection is through collaborative estimation [4], [5], which estimates QoS parameters between each client and all the web services by learning from past invocations from similar clients to similar web services, in a *neighbour-based* method called collaborative filtering. Another closely related problem is to estimate the latency between any two hosts in a network based on partial pairwise latency measurements in the network. Traditional solutions to such a task have relied on network coordinate systems [6]. Recently, low-rank matrix completion [7], [8] has been adopted as a new approach to recover and estimate pairwise latencies on the Internet and have shown significant performance improvement over traditional network coordinate systems. Furthermore, matrix factorization is one of the most popular solution techniques for matrix completion problems due to its scalability and the ease of large-scale implementation. Although matrix factorization problems are non-convex, they can be solved efficiently in practice at a large scale by several standard iterative optimization methods such as alternating minimization and stochastic gradient descent [9].

Nevertheless, a common limitation of all existing web service QoS estimation or network latency estimation studies is that they aim to estimate a *conditional mean* QoS value for each user-service pair and have ignored the fact that QoS metrics, including both latencies and throughput, may be highly skewed in reality. For example, in a dataset made available by recent studies [10], we can observe that most web service response times are within 100 milliseconds while a few outliers could exceed one or even several seconds due to network congestion or temporary service unavailability [8], [11]. In this case, state-of-the-art matrix factorization techniques, which aim to minimize a mean squared error, tend to only explain the conditional mean response time or throughput between a user and a web service, which may deviate from the "most probable" value due to the existence of outliers. Therefore, using conditional mean QoS estimates to select and recommend top web services for each user may

978-1-5090-5336-0/17/$31.00 ©2017 IEEE

lead to significant biases.

In this paper, we propose *Quantile Matrix Factorization* (QMF) and its extension QMF+ for web service QoS estimation for the purpose of personalized web service recommendation. Our contributions are manifold. *First*, we propose the concept of *Quantile Matrix Factorization* by replacing the least squares objective function in traditional matrix factorization with an asymmetric loss function similar to that in Quantile Regression [12]. The new objective function enables us to estimate the conditional *median* QoS metrics, which can better explain the central tendency of skewed data, or to estimate a certain *percentile* of the QoS value of interest. *Second*, although the QMF problem has a non-smooth quantile objective, we propose a simple yet efficient Iteratively Reweighted Least Squares (IRLS) algorithm to efficiently solve a smooth approximation of the QMF problem. *Finally*, we extend the proposed QMF model to QMF+, such that additional side attributes from both users and web services, such as countries and ASes, can be taken into account to enhance QoS estimation accuracy and web service ranking performance, especially for cases with extremely sparse past observations.

Through extensive evaluation based on a real-world QoS measurement dataset between 5825 web services and 339 users distributed worldwide, we show that our proposed algorithms greatly outperform several state-of-the-art personalized web service recommendation schemes, including probabilistic matrix factorization [13] and collaborative estimation [11], in terms of making personalized selection of the best k web services for each user and in terms of ranking these top k services.

The rest of the paper is organized as follows. We formulate the quantile matrix factorization problem for web service QoS estimation in Sec. II. In Sec. III, we present an iteratively reweighted least squares (IRLS) algorithm to solve a smoothed approximation of quantile matrix factorization. In Sec. IV, we further extend our model to incorporate the side attribute information of both users and web services to enhance QoS estimation accuracy. We evaluate our proposed algorithms in Sec. V on a real-world dataset and discuss the related literature in Sec. VI. Finally, the paper is concluded in Sec. VII.

II. Problem Description

Suppose we have a set m users and a set of n web services. Let $M \in \mathbb{R}^{m\times n}$ denote the matrix of QoS values between all the users and servers, where the entry $M_{u,i}$ represents the latency (or throughput) between user u and service i. Assume we have observed the QoS value between some user-service pairs. Let Ω denote the set of all such measured (u,i) pairs. The problem is to infer the missing QoS entries in M only based on the partially observed values $\Omega = \{M_{u,i}|\text{QoS value of } (u,i) \text{ has been observed}\}$.

A. Matrix Factorization for Web Service Recommendation

One effective and popular method that has been successfully adopted in network latency estimation based on partial observations is matrix factorization (e.g., [7], [8]). Matrix factorization solves a non-convex optimization problem, assuming that each user i has a latent feature vector $u_i \in \mathbb{R}^r$ and each service j has a latent feature vector $v_j \in \mathbb{R}^r$. Let $U := [u_1, \ldots, u_m]^\top$ and $V := [v_1, \ldots, v_n]^\top$. Then, the matrix factorization problem is to find two such tall matrices U and V, with $r \ll \{m, n, |\Omega|\}$ by solving

$$\min_{U\in\mathbb{R}^{m\times r}, V\in\mathbb{R}^{n\times r}} \sum_{(i,j)\in\Omega} \mathcal{L}(M_{i,j}, \hat{M}_{i,j}) \quad \text{s.t.} \quad \hat{M} = UV^\top.$$

The most commonly used loss function in matrix factorization is the squared loss $(M_{i,j} - u_i^\top v_j)^2$, with which the problem is to minimize the mean squared error (MSE):

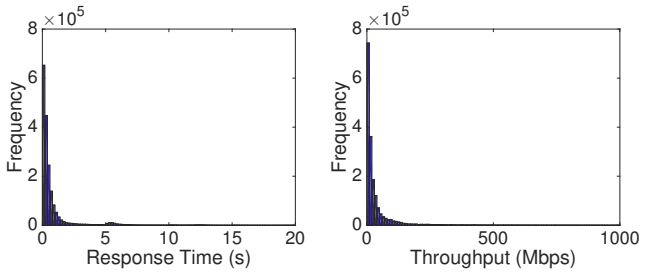
$$\begin{aligned} \min_{U,V} \quad & \frac{1}{2}\sum_{(i,j)\in\Omega} (M_{i,j} - \hat{M}_{i,j})^2 + \frac{\lambda}{2}\|U\|_F^2 + \frac{\lambda}{2}\|V\|_F^2 \\ \text{s.t.} \quad & \hat{M} = UV^\top, \forall U \in \mathbb{R}^{m\times r}, V \in \mathbb{R}^{n\times r}, \end{aligned} \tag{1}$$

where $\|\cdot\|_F$ denotes the matrix Frobenius norm. The term $\frac{\lambda}{2}\|U\|_F^2$ and $\frac{\lambda}{2}\|V\|_F^2$ are usually called regularizer in order to encourage simple models to avoid over-fitting issues (which make the model fit observed entries well but drift away from unknown ones). Like linear regression, solving (1) actually aims to produce an optimal solution $\hat{M}$ such that $\hat{M}_{i,j} = u_i^\top v_j$ estimates the *conditional mean* of the observation $M_{i,j}$. For symmetric noise following a Gaussian distribution, the conditional mean is the most efficient estimator.

However, for skewed or heavy-tailed noises, the conditional mean can be far away from the central area where $M_{i,j}$ is distributed, and thus is not representative of the underlying most frequent value of $M_{i,j}$. In these cases, we need to develop new techniques to better characterize the median and tail behavior of observations, beyond conditional mean estimates.

To get an intuitive idea about the distributions of typical web service QoS data, in Fig. 1(a) and Fig. 1(b), we plot the response times and throughput values between 339 service users and 5825 web services distributed all around the world from a publicly available dataset [10]. Here, the response time is the duration between the time that a service user sends a request and the time that he/she has received the corresponding response, including both the service latency and network latency. We can see that both response times and throughput are highly skewed. In Fig. 1(a) we can see that 90% of measured response times are smaller than 1.7s, yet the largest measured latency is 19.9s. More importantly, the mean response time is 0.9s which is much larger than the median 0.32s, around which most response times are distributed. Similarly, in Fig. 1(b), the mean throughput is 47.56 kbps, while the median is only 13.95 kbps and 90% of measured throughput values are smaller than 103.70 kbps while the largest measured value is 1000 kbps. This also shows that the throughput is highly skewed and heavy-tailed.

Such a significant discrepancy between the mean and the median implies that the conventional matrix factorization minimizing mean squared error (MSE) may not be suitable for estimating web service QoS metrics, since it tends to yield a mean estimation conditioned on the observations,



(a) Response Time (b) Throughput

Fig. 1. Histograms of response times and throughput between 5825 web services and 339 service users. Both QoS metrics are highly skewed.

while in reality QoS data will most likely concentrate around their median values. Moreover, the use of MSE-based matrix factorization can not estimate a certain percentile of the QoS values.

B. From Quantile Regression to Quantile Matrix Factorization

We now extend the idea of quantile regression to the case of matrix factorization, and we call such a new approach *quantile matrix factorization*.

Quantile regression estimates conditional quantile functions. More formally, the τ-th quantile of a real-valued random variable Z is defined as

$$Q_\tau(\mathrm{Z}) = \inf\{z : \Pr(\mathrm{Z} \le z) \ge \tau\},$$

among which the median is given by $Q_{0.5}(\cdot)$. If the random variable Z is conditional on some other variables or parameters, the quantile calculated here is called the *conditional quantile*.

A useful key observation [12] in quantile regression is

$$Q_\tau(\mathrm{Z}) = \arg\min_\beta \mathbb{E}[\rho_\tau(\mathrm{Z} - \beta)],$$

where

$$\rho_\tau(z) := z(\tau - \mathbb{1}(z < 0))$$

is called the *check loss function*, with $\mathbb{1}(z < 0)$ equal to 1 if $z < 0$ and equal to 0 otherwise. The rationale behind the above equation is that, for any $0 < \tau < 1$, minimizing $\rho_\tau(z)$ would push the lowest τ fraction to lie below z and $1 - \tau$ fraction to lie above z.

Given the above observation, we can estimate the quantile of a random variable Z from its sample observations $z_1, \ldots, z_N$ by solving the following optimization problem [12]:

$$\hat{Q}_\tau(\mathrm{Z}) = \arg\min_\beta \sum_{i=1}^{N} \rho_\tau(z_i - \beta). \tag{2}$$

Now we introduce a quantile-regression-like objective into the matrix factorization problem by replacing the squared loss in the original matrix factorization by a quantile check loss function. Under this setting, we can formulate the *Quantile Matrix Factorization* (QMF) problem as

$$\begin{aligned} \min_{U,V} \quad & \sum_{(i,j)\in\Omega} \rho_\tau(M_{i,j} - \hat{M}_{i,j}) + \frac{\lambda}{2}\|U\|_F^2 + \frac{\lambda}{2}\|V\|_F^2 \\ \text{s.t.} \quad & \hat{M} = UV^\top, U \in \mathbb{R}^{m\times r}, V \in \mathbb{R}^{n\times r}, \end{aligned} \tag{3}$$

which aims to produce an optimal solution $\hat{M}$ such that $\hat{M}_{i,j} = u_i^\top v_j$ estimates the τ-th *quantile* of the observation $M_{i,j}$. When $\tau = 0.5$, we are essentially estimating the median of each unobserved $M_{i,j}$.

Quantile regression has proven to be a powerful tool and well accepted method in economics, ecology and statistics. The above quantile-regression-like formulation has two unique strengths:

First, the QMF formulation in problem 3 can shift estimates from mean statistics to conditional medians or other quantiles of the observations. This is important in practice. The conventional MSE-based MF, as discussed above, is to minimize the mean squared error and actually estimates the mean of each $M_{i,j}$ given the observations. However, when the data is not Gaussian, this method is not guaranteed to be optimal, especially when the data are skewed or heavy-tailed.

Second, by focusing on different levels τ of quantiles, we can obtain a more complete picture of the data than the conventional MSE-based methods. Actually, the choice of τ depends on the practitioner's interests. If we are interested in the medians, we can set $\tau = 0.5$, and we can get the first and third quartiles by setting $\tau = 0.25$ and 0.75, respectively. If we are interested in the tail property, e.g., 10-percentile response time, we can set $\tau = 0.1$. In more complicated cases, we can even estimate a specific confidence interval for each $M_{i,j}$, e.g., a 90% confidence interval of each $M_{i,j}$ can be estimated by solving problem 3 under $\tau = 0.05$ to get a lower bound and then under $\tau = 0.95$ to get an upper bound.

Because of these two strengths, the formulation in problem 3 is particularly suitable for our web service recommendation task. First, as shown in Fig. 1(a) and 1(b), the response time and throughput are highly skewed. Secondly, we can utilize QMF to help a user narrow down the search of web services by estimating extreme quantile values. QMF in our web service recommender system can also help users exclude the worst services with long response time and low throughput, which can not be achieved by the traditional MSE-based estimator.

III. Algorithms

When it comes to directly optimizing the *non-smooth* quantile loss function, choices of efficient algorithms are limited. Meanwhile, the non-smooth characteristic also hinders its theoretical investigation. Many researchers use a smooth function to approximate the quantile loss function, e.g., [14]. In this paper, we use the same strategy to find a smooth approximation to the quantile objective and then provide an Iteratively Reweighted Least Square (IRLS) algorithm to solve the smoothed approximation.

For the residual $R_{i,j} := M_{i,j} - \hat{M}_{i,j}$, we have the following inequalities:

$$\rho_\tau(R_{i,j}) \le \sqrt{\rho_\tau^2(R_{i,j}) + \eta^2} \le \frac{1}{2}\left(\frac{\rho_\tau^2(R_{i,j}) + \eta^2}{z_{i,j}} + z_{i,j}\right)$$

for $\eta > 0, z_{i,j} > 0$. For the first inequality, we can choose a small constant η related to the sample size $|\Omega|$, e.g., $\eta = 1/\log(|\Omega|)$. Then the gap between the first two terms will

Algorithm 1 IRLS for smoothed quantile matrix factorization.

1: **Input**: $\mathcal{P}_\Omega$, target rank r
2: **Parameter**: Smooth constant $\eta > 0$, maximum number of iterations T
3: Initialize $U^{(0)}$ and $V^{(0)}$ with random values
4: **for** $t = 0$ to $T-1$ **do**
5: **for** $i = 1, \ldots, m$ **do**
6: $u_i^{(t+1)} \leftarrow \arg\min_{u \in \mathbb{R}^r} h_i(u; V^{(t)}, Z^{(t)})$
7: **end for**
8: **for** $j = 1, \ldots, n$ **do**
9: $v_j^{(t+1)} \leftarrow \arg\min_{v \in \mathbb{R}^r} h_j(v; U^{(t)}, Z^{(t)})$
10: **end for**
11: $M^{(t+1)} \leftarrow U^{(t+1)} V^{(t+1)\top}$
12: **for** $(i,j) \in \Omega$ **do**
13: $R_{i,j}^{(t+1)} \leftarrow M_{i,j} - u_i^{(t+1)\top} v_j^{(t+1)}$
14: $z_{i,j}^{(t+1)} \leftarrow \sqrt{\rho_\tau^2(R_{i,j}^{(t+1)}) + \eta^2}$
15: **end for**
16: **end for**
17: **Output**: $M \leftarrow U^{(T)} V^{(T)\top}$.

diminish as p increases. When $z_{i,j} = \sqrt{\rho_\tau^2(R_{i,j}) + \eta^2}$, the last term is minimized and the second inequality becomes the equality. Both the last two terms are smooth approximations of the quantile loss function and under certain conditions, the approximations could be as close as possible. Motivated by this fact, we solve the following problem as a smooth approximation of quantile matrix factorization:

$$\begin{aligned} \min_{U,V} \quad & \sum_{(u,i)\in\Omega} \sqrt{\rho_\tau^2(M_{i,j} - \hat{M}_{i,j}) + \eta^2} + \frac{\lambda}{2}\|U\|_F^2 + \frac{\lambda}{2}\|V\|_F^2 \\ \text{s.t.} \quad & \hat{M} = UV^\top, \forall U \in \mathbb{R}^{m\times r}, V \in \mathbb{R}^{n\times r}. \end{aligned} \tag{4}$$

Considering the fact that both the number of users and web services can be very large in the real world, the optimization algorithm used should have simple steps in each iteration, with intermediate results that are easy to store. One very popular family of such algorithms is the Block-Coordinate Descent (BCD) method, which alternately minimizes each coordinate or variable, i.e., each row of U or V will be updated alternately in a sequential order or in parallel. However, since minimizing a square root loss function directly is not an easy task, we can try to minimize an approximate function instead. Under some conditions, minimizing an approximate function can even have performance similar to minimizing the original loss function. The Iteratively Reweighted Least Square (IRLS) algorithm [15] is such an example, and has recently been extended to a framework called block successive upper-bound minimization (BSUM) [16].

Motivated by IRLS and BSUM, we propose an algorithm that iteratively minimizes the following two functions for each user i and web service j:

$$h_i(u; V, Z) := \frac{1}{2} \sum_{j\in\Omega(i)} \frac{\rho_\tau^2(M_{i,j} - u^\top v_j)}{z_{i,j}} + \frac{\lambda}{2}\|u\|_2^2,$$

$$h_j(v; U, Z) := \frac{1}{2} \sum_{i\in\Omega(j)} \frac{\rho_\tau^2(M_{i,j} - u_i^\top v)}{z_{i,j}} + \frac{\lambda}{2}\|v\|_2^2,$$

where $\Omega(i)$ denotes the set of observed services for user i, $\Omega(j)$ denotes the set of observed users for service j, and $\|\cdot\|_2$ denotes the ℓ_2 norm. The full procedure of our algorithm is shown in Algorithm 1. Note that according to Theorem 2 in [16], we can conclude that Algorithm 1 can converge to stationary points.

Now we present our approach to solve the above two sub-problems (shown in Steps 6 and 9) by rewriting them as Quadratic Programming (QP) problems. Then, we can use some standard solvers to solve QP. For simplicity, we only rewrite Step 6.

Suppose we have observed l web services for user i. For residual $R_{i,j}$, we can extract its positive component $R_{i,j}^+ := \max(R_{i,j}, 0)$ and its negative component $R_{i,j}^- := -\min(R_{i,j}, 0)$. Then, we can rewrite the corresponding term in $h_i(u; V, Z)$ as

$$\frac{\rho_\tau^2(R_{i,j})}{z_{i,j}} = \tau^2 (R_{i,j}^+)^2 / z_{i,j} + (1-\tau)^2 (R_{i,j}^-)^2 / z_{i,j} \tag{5}$$

$$= \tau^2 (S_{i,j}^+)^2 + (1-\tau)^2 (S_{i,j}^-)^2, \tag{6}$$

where we denote $S_{i,j}^+ := R_{i,j}^+ / \sqrt{z_{i,j}}$ and $S_{i,j}^- := R_{i,j}^- / \sqrt{z_{i,j}}$. All such $S_{i,j}^+$ and $S_{i,j}^-$ form vectors $s^+ \in \mathbb{R}_+^l$ and $s^- \in \mathbb{R}_+^l$, respectively. We then denote $b_j = M_{i,j} / \sqrt{z_{i,j}}$ and $v_j' = v_j / \sqrt{z_{i,j}}$ for $j \in \Omega(i)$. Then, we have

$$S_{i,j}^+ - S_{i,j}^- = (M_{i,j} - v_j^\top u) / \sqrt{z_{i,j}} = b_j - v_j'^\top u.$$

We can finally convert the sub-problem in Step 6 into the following QP problem:

$$\begin{aligned} \min_{u, s^+, s^-} \quad & \tau^2 \|s^+\|_2^2 + (1-\tau)^2 \|s^-\|_2^2 + \frac{\lambda}{2}\|u\|_2^2 \\ \text{s.t.} \quad & s_j^+ - s_j^- = b_j - v_j'^\top u, \forall j \in \Omega(i), \\ & u \in \mathbb{R}^r, s^+, s^- \in \mathbb{R}_+^l \end{aligned} \tag{7}$$

IV. Quantile Matrix Factorization with Side Attributes

The previous section provides a new quantile matrix factorization approach to complete the QoS parameter matrix with some measured entries $M_{i,j}$. In this section, we further enhance the QoS prediction accuracy by exploiting some readily obtainable side attributes of both users and web services such as the location and ISP information. The challenge here is how to extend the quantile matrix factorization model (3) to seamlessly incorporate context information of users and web services. We propose a new formulation, called *QMF+*, that incorporates user and server attributes into the proposed QMF model, and then describe our modified algorithm to solve QMF+.

Suppose we have collected some user attributes as *explicit user features* and collected some web service attributes as *explicit service features*. Let $x_i \in \mathbb{R}^p$ and $y_j \in \mathbb{R}^q$ denote the explicit feature vectors for user i and web service j, respectively. We assume that the QoS value between user i

and service j depends on the inner product of latent feature vectors, i.e., $u_i^\top v_j$, as well as on their explicit features via a bilinear product, i.e., $x_i^\top W y_j$. In other words, the QoS value between user i and web service j is modeled as

$$\hat{M}_{i,j} = u_i^\top v_j + x_i^\top W y_j.$$

Let $X = [x_1, \ldots, x_m]^\top$ and $Y = [y_1, \ldots, y_n]^\top$. Given the explicit user features X, explicit service features Y, and a subset of observed QoS values $M_{i,j}$, the problem of estimating the τ-th quantiles of QoS metrics with side attributes (QMF+) is to solve the following optimization problem:

$$\begin{aligned} \min_{U,V,W} \quad & \sum_{(i,j)\in\Omega} \rho_\tau(M_{i,j} - \hat{M}_{i,j}) + \frac{\lambda}{2}\|U\|_F^2 + \frac{\lambda}{2}\|V\|_F^2 \\ \text{s.t.} \quad & \hat{M} = UV^\top + XWY^\top, \\ & U \in \mathbb{R}^{m\times r}, V \in \mathbb{R}^{n\times r}, W \in \mathbb{R}^{p\times q}. \end{aligned} \tag{8}$$

Note that for web service recommendation, the number of explicit user features p and the number of explicit service features q are usually small relative to the number of users and services. Hence, there is no need to have a regularizer for W.

To solve (8), we can extend Algorithm 1 to alternately minimize W, U and V, and then update the residuals $R_{i,j}$ and the corresponding weights $z_{i,j}$ for all known entries $(i,j) \in \Omega$. Note that updating entries of U and V is exactly the same as what we did in Algorithm 1, where $M_{i,j}$ is substituted by $M'_{i,j} := M_{i,j} - x_i^\top W y_j$ in QMF+.

We now describe how to update W. Define $\tilde{M} := M - UV^\top$ and

$$H(W; U, V, Z) := \frac{1}{2} \sum_{(i,j)\in\Omega} \frac{\rho_\tau^2(\tilde{M}_{i,j} - x_i^\top W y_j)}{z_{i,j}}.$$

We can minimize the above function efficiently in two common cases. First, if both p and q are relatively small, we can directly solve W by the same technique that we used to solve for U and V. Let the residual $\tilde{R}_{i,j} := \tilde{M}_{i,j} - x_i^\top W y_j$, and denote $\tilde{S}^+_{i,j} := \tilde{R}^+_{i,j}/\sqrt{z_{i,j}}$ and $\tilde{S}^-_{i,j} := \tilde{R}^-_{i,j}/\sqrt{z_{i,j}}$. All such $\tilde{S}^+_{i,j}$ and $\tilde{S}^-_{i,j}$ can also form vectors $\tilde{s}^+ \in \mathbb{R}_+^{|\Omega|}$ and $s^- \in \mathbb{R}_+^{|\Omega|}$, respectively, and we can finally convert it into a QP problem like (7).

Another common case is when both explicit user features and explicit service features are sparse, i.e., when all x_i's and y_j's have few nonzero entries. This happens when the explicit features are generated from some categorical variables. For example, suppose the users are from p countries in total and the web services are from q countries in total. We can generate their explicit features $x_i \in \mathbb{R}^p$ and $y_j \in \mathbb{R}^q$ by setting $x_{i,p'} = 1$ if user i is from country p' and $x_{i,p'} = 0$ otherwise (the same applies to $y_{j,q'}$). In this case, only one entry in each x_i (or y_j) is nonzero. Letting $\mathcal{W}(p', q')$ denote the set $\{(i,j) : x_{i,p'} = 1, y_{j,q'} = 1\}$, we have

$$H(W; U, V, Z) = \sum_{p'=1}^{p}\sum_{q'=1}^{q} \frac{1}{2} \sum_{(i,j)\in\mathcal{W}(p',q')} \frac{\rho_\tau^2(\tilde{M}_{i,j} - W_{p',q'})}{z_{i,j}},$$

where

$$W_{p',q'} := \arg\min_w \frac{1}{2} \sum_{(i,j)\in\Omega} \frac{\rho_\tau^2(\tilde{M}_{i,j} - w)}{z_{i,j}}.$$

The minimization problem above is a special case of subproblem (7) with $r = 1$. If there is more than one attribute (e.g., not only countries but also ISPs), we can divide W into several blocks and update each block alternately using the above procedure.

In other more complicated cases, we may need some additional constraints on W to be able to solve the problem. For example, we can further assume that W is also low-rank, and problem (8) can be regarded as a variation of *factorization machines* [17]. However, further discussions on more general explicit features are beyond the scope of this paper. And we refer interested readers to the literature for details.

V. Performance Evaluation

We evaluate our QMF and QMF+ methods on a publicly available dataset, which contains response time and throughput records between 339 users and 5825 web services distributed worldwide, made available by a previous study [10]. The dataset also contains explicit user features and service features, including the country, autonomous system (AS), IP address, latitude and longitude.

A. Experimental Setup

We will first evaluate the performance of QMF and QMF+ in comparison to existing web service recommendation schemes in terms of two performance metrics, namely, *NDCG@k* (Normalized Discounted Cumulative Gain at top k) and *Precision@k* (Precision at top k). These are two popular performance metrics for evaluating recommendation and ranking effectiveness. Instead of evaluating the relative/absolute pairwise prediction errors, these two metrics compare the gap between the *predicted order* and the *observed order*. Precision@k measures how many true top-k services in the observation are correctly predicted by an algorithm. Formally, let $\mathcal{P}_k(i)$ be the *predicted* set of top k services in terms of a QoS metric, and $\mathcal{P}^*_k(i)$ be the *observed* set, and Precision@$k(i)$ for user i is defined as

$$\text{Precision@}k(i) = \frac{1}{k} \sum_{j\in\mathcal{P}_k(i)} \mathbb{1}(j \in \mathcal{P}^*_k(i)).$$

Given a predicted ranked list of services π_i for user i, the NDCG@k metric is defined as

$$\text{NDCG@}k(i) := \frac{\text{DCG@}k(i, \pi_i)}{\text{DCG@}k(i, \pi_i^*)},$$

where

$$\text{DCG@}k(i, \pi_i) = rel(i, \pi_i(1)) + \sum_{j=2}^{k} \frac{rel(i, \pi_i(j))}{\log_2 j},$$

where the value $rel(i, \pi_i(j))$ is the relevance of the service $\pi_i(j)$. In our experiments, we simply use the observed QoS metrics as the relevance values.

TABLE I
RANKING PERFORMANCE COMPARISON OF RESPONSE TIME AND THROUGHPUT ON NDCG@k AND PRECISION@k (LARGER VALUE INDICATES HIGHER RANKING PERFORMANCE). HERE N@k INDICATES NDCG@k AND P@k INDICATES PRECISION@k

		Sampling Rate 1%				Sampling Rate 10%				Sampling Rate 30%			
Data	Method	N@10	N@100	P@10	P@100	N@10	N@100	P@10	P@100	N@10	N@100	P@10	P@100
RT	QMF, 0.1	0.217	0.341	0.008	0.107	0.572	0.638	0.036	0.391	0.568	0.650	0.040	0.413
	QMF, 0.25	0.354	0.420	0.016	0.165	**0.615**	0.694	**0.042**	0.473	**0.592**	0.703	0.032	0.490
	QMF, 0.5	**0.409**	**0.453**	**0.017**	**0.184**	0.595	**0.715**	0.037	**0.520**	0.563	**0.713**	**0.042**	**0.523**
	PMF	0.135	0.130	0.013	0.036	0.085	0.165	0.013	0.044	0.059	0.212	0.018	0.062
	QMF+,0.1	0.365	0.487	0.037	0.276	0.593	0.667	0.046	0.434	0.673	0.679	0.073	0.457
	QMF+,0.25	0.467	0.528	0.043	0.337	**0.674**	0.714	0.066	0.512	**0.684**	0.725	0.061	**0.535**
	QMF+,0.5	**0.563**	**0.543**	**0.113**	**0.375**	0.614	**0.728**	**0.127**	**0.525**	0.662	**0.739**	**0.139**	0.534
	LoRec	0.083	0.140	0.013	0.040	0.062	0.186	0.008	0.046	0.079	0.166	0.013	0.044
TP	QMF-0.1	0.272	0.252	0.016	0.048	0.442	0.521	0.017	0.146	0.586	0.670	0.026	0.279
	QMF-0.25	0.548	0.472	0.059	0.155	0.650	0.696	0.033	0.313	0.743	**0.779**	0.067	**0.429**
	QMF-0.5	**0.601**	**0.586**	**0.078**	**0.222**	**0.756**	**0.779**	**0.075**	**0.429**	**0.779**	0.774	**0.103**	0.410
	PMF	0.020	0.052	0.005	0.012	0.041	0.058	0.010	0.018	0.079	0.095	0.016	0.025
	QMF+,0.1	0.332	0.301	0.022	0.030	0.501	0.557	0.047	0.201	0.662	0.740	0.064	0.383
	QMF+,0.25	**0.385**	0.342	0.029	0.085	0.559	**0.608**	0.053	**0.245**	0.704	**0.771**	0.077	**0.433**
	QMF+,0.5	0.384	**0.369**	**0.037**	**0.096**	**0.616**	0.604	**0.068**	0.244	**0.740**	0.770	**0.099**	0.432
	LoRec	0.020	0.043	0.006	0.012	0.031	0.061	0.015	0.021	0.058	0.080	0.011	0.023

We also evaluate the recovery accuracy of QMF and QMF+ in comparison to several state-of-the-art web service recommendation schemes, using the relative estimation errors (REs) on missing entries, which are defined as $|M_{i,j}-\hat{M}_{i,j}|/M_{i,j}$ for $(i,j) \notin \Omega$. In particular, we compare the following schemes:

- **Algorithm 1 (QMF) and its extension (QMF+)**: for each algorithm we set three quantiles $\tau = 0.1, 0.25, 0.5$, which represent the 10% quantile, the first quartile, and the median, respectively.
- **PMF (Probabilistic Matrix Factorization)**: a widely-used implementation of the matrix factorization model [18], which has been used to predict the response times of web services [13] with a loss function of MSE.
- **LoRec** [11]: a state-of-the-art web service recommender system which employs the location information to enhance QoS prediction accuracy. This collaborative-filtering-based system predicts QoS metric of a user-service pair by aggregating the observed values from similar users, where the similarity between users is defined according to their geo-locations and observed QoS metrics.

The user-service matrices in the real world are typically very sparse, since a user may have ever connected to only a small number of web services. We randomly choose a subset of observed values in the user-service matrix with different sampling rate, which is the ratio of the number of known entries in M to the number of all the entries. In particular, we randomly set 1%, 10% and 30% of the matrix entries as observed. Our algorithms and other baseline algorithms are employed to estimate the missing QoS values and predict the ranking of web services for each user in the descending order of the corresponding QoS values. For QMF and QMF+, we set the dimension of latent feature vectors to $r = 10$, and the smooth constant $\eta = 1/\log(|\Omega|)$ as we have discussed in Sec. II. For PMF, we also set the dimension of latent feature vectors as $r = 10$. For QMF+ and LoRec, we only consider the country in the explicit features of both users and web services, since the baseline scheme LoRec only considers location information.

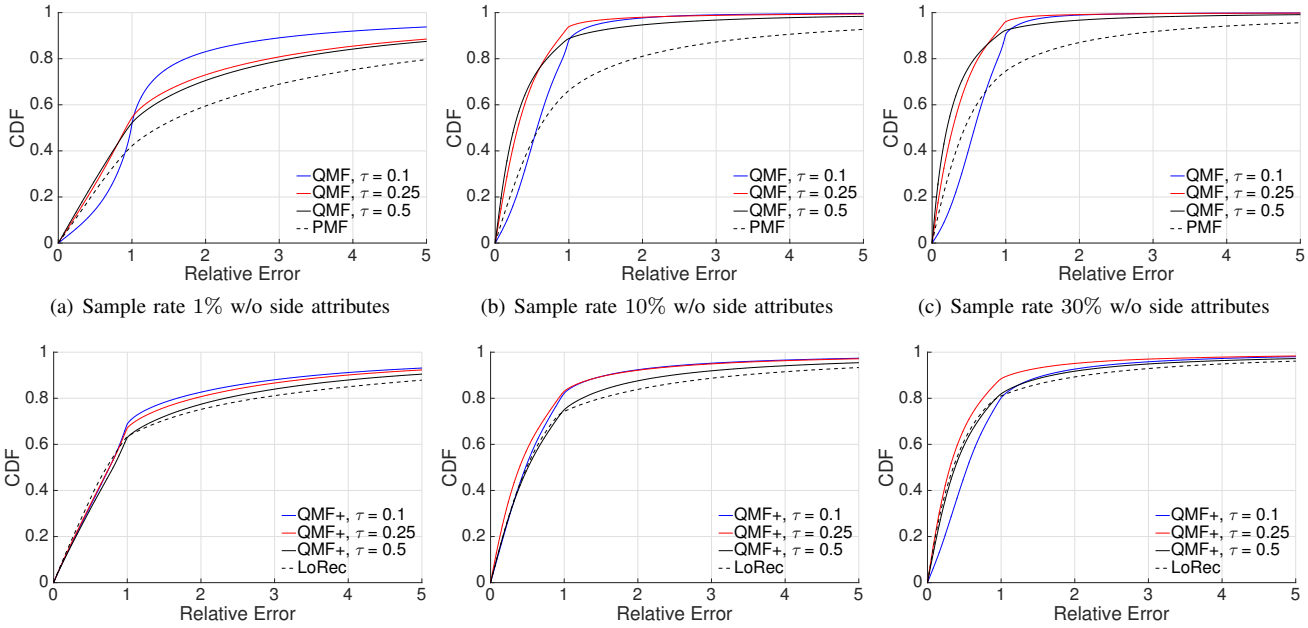
B. Ranking performance

Table I shows the ranking performance in NDCG and precision of response time and throughput under the sampling rates of 1%, 10% and 30%. In this table we focus on four metrics: NDCG@10, NDCG@100, Precision@10 and Precision@100. We boldface the best performance for each column in the Table I.

Compared with PMF, which minimizes MSE, Algorithm 1 obtains better prediction accuracy for both response time and throughput under all settings of τ. Since the data is highly skewed, our QMF algorithm can estimate different quantiles which are closer to the center of distribution. In particular, we observe highest NDCG and precision scores under $\tau = 0.5$ in most cases. This implies that estimating median is more robust than estimating mean. Similarly, the QMF+ method which exploits geo-location information outperforms LoRec under all settings of τ.

By comparing results among QMF and QMF+, it is clear that the prediction accuracy of QMF+ is further improved over QMF by exploiting both users and servers location as side attributes on response time dataset. In particular, we can see that the improvements of QMF+ are more significant in low sampling rate 1% than high sampling rates 10% and 30%. When the sampling rate is low, side attributes can help to provide some initial estimation. Generally speaking, the number of explicit user features is less than that of users, and so for services. In addition, we can see that the ranking performance results from QMF+ methods under all τ's are closer than those from QMF methods, which shows that geo-location information can help to improve the robustness.

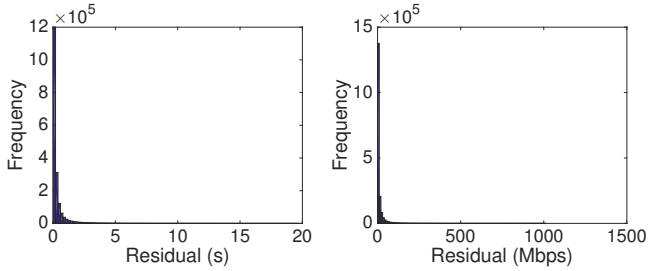
For both QMF and QMF+, we can see that the ranking results under $\tau = 0.1$ are worse than results under $\tau = 0.25$ and $\tau = 0.5$, since the quantile $\tau = 0.1$ is more extreme than $\tau = 0.25$ and $\tau = 0.5$ and the estimation under such



(a) Sample rate 1% w/o side attributes (b) Sample rate 10% w/o side attributes (c) Sample rate 30% w/o side attributes

(d) Sample rate 1% + side attributes (e) Sample rate 10% + side attributes (f) Sample rate 30% + side attributes

Fig. 2. The CDFs of relative estimation errors of response time on the missing values with sample rate 1%, 10% and 30%.



(a) Response Time (b) Throughput

Fig. 3. Histograms of residuals via MSE minimization

quantile needs more data. When explicit user features and explicit service features are incorporated, they can provide initial estimation and it leads to decrease the demand of samples and lead to better performance. This is confirmed in Table I, in which we can see that QMF+ always outperforms QMF for $\tau = 0.1$, and the improvement of QMF+ from QMF under the sampling rate 1% is higher than that under the sampling rate of 10% and 30%.

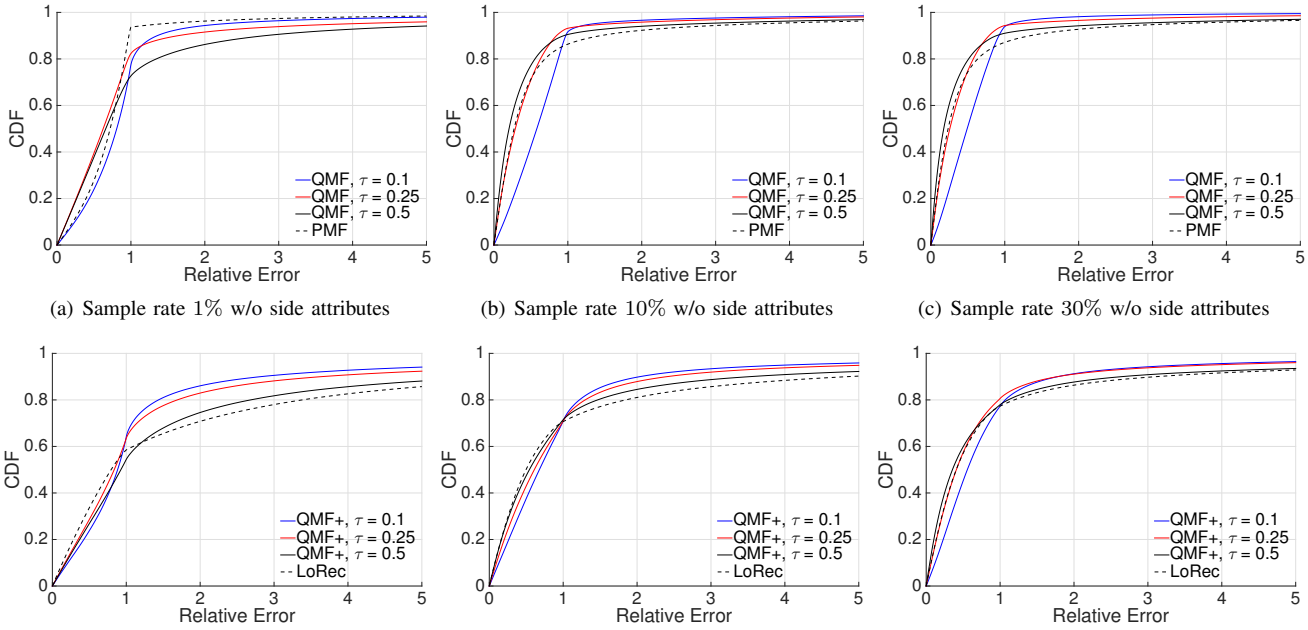
Now let's see why our QMF is better in terms of ranking. We did top $r = 10$ singular value decomposition (SVD) for both response time matrix and throughput matrix and we plot the results in Fig. 3. It is well known that top r SVD of M provides its best approximation of rank r in terms of Frobenius norm [19], which is actually the MSE. We plot the residuals of such method in Fig. 3. In these figures, 90% of residuals of response time are smaller than 0.8, while the largest residual can be as large as 19.73. Also, 90% of residuals of throughput are smaller than 30.75, but the largest residual is 1011. And now it is clear to see that in these two datasets, the residuals are still highly skewed. Then we can conclude that if we use the conditional mean for ranking web services, the results are not accurate, because centers of these two datasets are distributed far away from their conditional means.

C. Recovery accuracy

We plot the relative estimation errors on missing response time in Fig. 2(a)-2(c), and on missing throughput in Fig. 4(a)-4(c), respectively, under three settings of QMF and PMF. In Fig. 2(a)-2(c), we can see that PMF is inferior to QMF under $\tau = 0.25$ and 0.5 because PMF is targeting minimizing MSE to estimate the conditional mean, but the highly skewness of response time distribution leads the mean to be far away from the center. For QMF with $\tau = 0.1$, we can see that there are fewer small relative errors and large relative errors than others, and these errors concentrate on a small range, especially under low sampling rate as shown in Fig. 2(a). For the throughput results in Fig. 4(a)-4(c), our algorithm is only slightly better than PMF. We also compare QMF+ under the same setting with LoRec plotted in Fig. 2(d)-4(f) and Fig. 4(d)-4(f). We can observe similar results that LoRec is inferior to QMF+ under $\tau = 0.25$ and 0.5.

D. Impact of the latent feature dimension

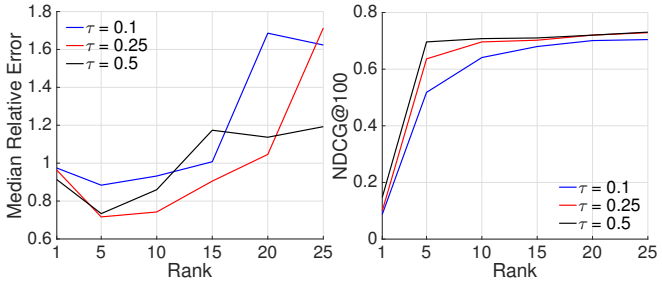
We further investigate the impact of r, the dimension of user and service latent feature vectors. In this experiment, we set the sampling rate to be 10% and compare the median of relative estimation errors, called *median relative error*, and the NDCG@100 score. We test the impact of the dimension r on both QMF and QMF+ for the response time dataset, and plot the results in Fig. 5 and Fig. 6, respectively. We can clearly see that both the ranking precision and the estimation precision increase as the latent feature dimension increases. However, when $r > 15$, the ranking performance almost stops increasing. In addition, a higher dimension introduces higher



(a) Sample rate 1% w/o side attributes (b) Sample rate 10% w/o side attributes (c) Sample rate 30% w/o side attributes

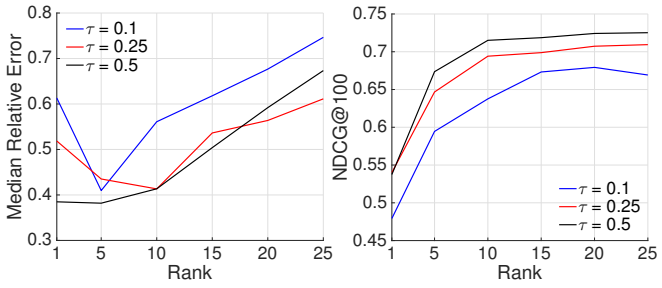
(d) Sample rate 1% + side attributes (e) Sample rate 10% + side attributes (f) Sample rate 30% + side attributes

Fig. 4. The CDFs of relative estimation errors of throughput on the missing values with sample rate 1%, 10% and 30%.



(a) Median Relative Error (b) NDCG@100

Fig. 5. Impact of dimension of latent vectors on QMF in terms of median relative error and NDCG@100.



(a) Median Relative Error (b) NDCG@100

Fig. 6. Impact of dimension of latent vectors on QMF+ in terms of median relative error and NDCG@100.

computational cost. Therefore, we suggest that r should be set in the range 5–15. The trend shown in these two figures also holds for other ranking metrics, e.g., precision$k$, and for other estimation error metrics, e.g., mean relative errors. The results on the throughput dataset are similar and omitted due to the space limit.

VI. Related Work

Early work on Internet latency estimation was mainly based on network coordinate systems (NCSs), which embed Internet hosts into a coordinate system such as a Euclidean space and predict the latency between a pair of hosts by calculating their corresponding distance [20] in the embedded space. However, most existing NCSs (e.g., Vivaldi [6], GNP [21]) use Euclidean embedding and thus cannot model triangle inequality violation (TIV), which has been widely reported [22], [23] on the Internet.

A popular approach to QoS prediction for web services is collaborative filtering or collaborative prediction, typically represented by the neighbor-based method, including the user-based and the item-based approaches, which calculates similarities between users or between services based on partially observed QoS measurements. For example, a user-based approach has been proposed in [24] that predicts the missing QoS of a user-service pair based on the QoS experiences of other similar users in the past, where user similarities are computed according to their QoS values to the same web services. A hybrid collaborative QoS estimation method has been proposed in [25], which combines the user-based and item-based approaches. Furthermore, [11] proposes a method which exploits the location information as well as the observed QoS parameters for service recommendation.

Another type of methods, namely matrix completion and matrix factorization, has recently gained enormous success in solving both collaborative filtering recommendation problems [26] and network latency estimation problems. In matrix factorization, the QoS matrix is factorized into two latent feature matrices, one for users and another for services, and each QoS value is estimated by the inner product of the latent feature

vectors of the corresponding user and service. For network latency prediction, DMFSGD [7] successfully uses matrix factorization to estimate the latencies between Internet hosts and achieves a lower prediction error than network coordinate systems. A hybrid method combining Euclidean embedding and matrix factorization has been proposed in [8] to predict latencies between personal devices including mobile devices. The matrix factorization approach has also been adopted for web service recommendation in [13].

However, all existing matrix factorization techniques attempt to minimize the mean squared error (MSE) and thus only estimate the mean QoS statistics, which can be far away from the center of the distribution of response times (or throughput), if the distribution is heavy-tailed. The proposed QMF solves this issue by estimating the *median* QoS values, which better model the most frequent QoS values between users and services and are therefore more robust to outliers. In the meantime, we can also obtain a full picture of QoS estimates by setting different levels of quantiles.

VII. Concluding Remarks

In this paper, we propose a personalized web service recommendation method based on the effective estimation of response times and throughput. Since it is impractical to measure such QoS values for all pairs of user and web service, we estimate the unknown values from partially observed ones via matrix completion. Compared with existing popular matrix factorization approaches, which aim at minimizing the mean squared error and actually estimate the conditional means of the QoS values, we propose the Quantile Matrix Factorization (QMF) which novelly combines the quantile regression technique and matrix factorization. We propose an efficient algorithm based on Iterative Reweighted Least Squares (IRLS) to solve QMF and further extend our model to take explicit attributes on both the user and web service sides into account. Extensive evaluations based on a real-world dataset of web service QoS measurements show that QMF significantly outperforms several state-of-the-art QoS prediction and recommendation algorithms based on matrix factorization or collaborative filtering, especially in terms of excluding services with the highest response times and selecting the best services. Furthermore, the prediction and ranking performance of QMF and QMF+ is particularly robust to the skewed QoS data.

References

[1] M. Alrifai, D. Skoutas, and T. Risse, "Selecting skyline services for QoS-based web service composition," in *Proc. ACM WWW*, 2010, pp. 11–20.

[2] A. Klein, F. Ishikawa, and S. Honiden, "Towards network-aware service composition in the cloud," in *Proc. ACM WWW*, 2012, pp. 959–968.

[3] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient algorithms for web services selection with end-to-end QoS constraints," *ACM Transactions on the Web (TWEB)*, vol. 1, no. 1, p. 6, 2007.

[4] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized QoS prediction forweb services via collaborative filtering," in *Proc. IEEE ICWS*, July 2007, pp. 439–446.

[5] Y. Zhang, Z. Zheng, and M. Lyu, "Exploring latent features for memory-based QoS prediction in cloud computing," in *Proc. IEEE Symp. Rel. Syst. (SRDS)*, Oct 2011, pp. 1–10.

[6] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," in *Proc. ACM SIGCOMM*, 2004.

[7] Y. Liao, W. Du, P. Geurts, and G. Leduc, "DMFSGD: A decentralized matrix factorization algorithm for network distance prediction," *IEEE/ACM Trans. Netw. (TON)*, vol. 21, no. 5, pp. 1511–1524, 2013.

[8] B. Liu, D. Niu, Z. Li, and H. V. Zhao, "Network latency prediction for personal devices: Distance-feature decomposition from 3D sampling," in *Proc. IEEE INFOCOM*, 2015, pp. 307–315.

[9] R. Sun and Z.-Q. Luo, "Guaranteed matrix completion via nonconvex factorization," in *Proc. IEEE FOCS*, 2015, pp. 270–289.

[10] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of real-world web services," *IEEE Trans. Service Comput.*, vol. 7, no. 1, pp. 32–39, 2014.

[11] X. Chen, Z. Zheng, Q. Yu, and M. R. Lyu, "Web service recommendation via exploiting location and QoS information," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1913–1924, 2014.

[12] R. Koenker, *Quantile regression*. Cambridge university press, 2005, no. 38.

[13] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Collaborative web service qos prediction via neighborhood integrated matrix factorization," *IEEE Trans. Service Comput.*, vol. 6, no. 3, pp. 289–299, 2013.

[14] C. Chen, "A finite smoothing algorithm for quantile regression," *J. Comp. Graph. Stat.*, 2012.

[15] A. Beck, "On the convergence of alternating minimization for convex programming with applications to iteratively reweighted least squares and decomposition schemes," *SIAM J. Optim.*, vol. 25, no. 1, pp. 185–209, 2015.

[16] M. Razaviyayn, M. Hong, and Z.-Q. Luo, "A unified convergence analysis of block successive minimization methods for nonsmooth optimization," *SIAM J. Optim.*, vol. 23, no. 2, pp. 1126–1153, 2013.

[17] S. Rendle, "Factorization machines," in *Proc. IEEE ICDM*, 2010, pp. 995–1000.

[18] A. Mnih and R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2007, pp. 1257–1264.

[19] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[20] B. Donnet, B. Gueye, and M. A. Kaafar, "A survey on network coordinates systems, design, and security," *Commun. Surveys Tuts.*, vol. 12, no. 4, 2010.

[21] T. E. Ng and H. Zhang, "Predicting internet network distance with coordinates-based approaches," in *Proc. IEEE INFOCOM*, vol. 1, 2002.

[22] J. Ledlie, P. Gardner, and M. I. Seltzer, "Network coordinates in the wild." in *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, vol. 7, 2007.

[23] S. Lee, Z.-L. Zhang, S. Sahu, and D. Saha, "On suitability of euclidean embedding of internet hosts," in *Proc. ACM SIGMETRICS*, vol. 34, no. 1, 2006.

[24] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized QoS prediction for web services via collaborative filtering," in *Proc. IEEE ICWS*, 2007, pp. 439–446.

[25] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware web service recommendation by collaborative filtering," *IEEE Trans. Service Comput.*, vol. 4, no. 2, pp. 140–152, 2011.

[26] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proc. ACM KDD*, Las Vegas, Nevada, August, 2008.