A Data-Aware Latent Factor Model for Web Service QoS Prediction

Di Wu[1], Xin Luo[1(✉)], Mingsheng Shang[1], Yi He[3], Guoyin Wang[2(✉)], and Xindong Wu[3]

[1] Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China
{wudi,luoxin21,msshang}@cigit.ac.cn
[2] Chongqing Key Laboratory of Computational Intelligence, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
wanggy@ieee.org
[3] University of Louisiana at Lafayette, Lafayette 70503, USA
{yi.hel,xwu}@louisiana.edu

Abstract. Accurately predicting unknown quality-of-service (QoS) data based on historical QoS records is vital in web service recommendation or selection. Recently, latent factor (LF) model has been widely and successfully applied to QoS prediction because it is accurate and scalable under many circumstances. Hence, state-of-the-art methods in QoS prediction are primarily based on LF. They improve the basic LF-based models by identifying the neighborhoods of QoS data based on some additional geographical information. However, the additional geographical information may be difficult to collect in considering information security, identity privacy, and commercial interests in real-world applications. Besides, they ignore the reliability of QoS data while unreliable ones are often mixed in. To address these issues, this paper proposes a data-aware latent factor (DALF) model to achieve highly accurate QoS prediction, where 'data-aware' means DALF can easily implement the predictions according to the characteristics of QoS data. The main idea is to incorporate a density peaks based clustering method into an LF model to discover the neighborhoods and unreliable ones of QoS data. Experimental results on two benchmark real-world web service QoS datasets demonstrate that DALF has better performance than the state-of-the-art models.

1 Introduction

Web services are software components used to exchange data between two software systems over a network [1]. In this era of the Internet, there are numerous online web services [2]. How to select optimal ones from a large candidate set and recommend them to potential users becomes a hot yet thorny issue [3].

Quality-of-Service (QoS) is essential for addressing such an issue because it is a significant factor to evaluate the performance of web services [1, 2, 4]. Once QoS data of candidate web services are obtained, reliable ones can be selected and recommended

Q. Yang et al. (Eds.): PAKDD 2019, LNAI 11439, pp. 384–399, 2019.
https://doi.org/10.1007/978-3-030-16148-4_30

to potential users accordingly. Conducting warming-up tests is an important way to acquire QoS data. However, it is economically expensive [3, 5, 6].

Alternatively, QoS prediction is another widely used way to acquire QoS data [5–9]. Its principle is to predict unknown QoS data based on historical records and/or other information. Collaborative filtering (CF), which has been successfully applied to e-commerce recommendation systems [10, 11], is frequently adopted to implement QoS prediction [5–9, 12–17]. CF-based QoS prediction is developed based on a user-service QoS matrix [5–9, 12–17], where each column denotes a specified web service, each row denotes a specified user, and each entry stands for a historical QoS record produced by a specified user invoking a specified web service. Such a matrix is sparse [5–9, 12–17]. Thus, how to accurately predict the missing data of the sparse user-service QoS matrix based on its known ones is the key to achieve CF-based QoS prediction.

Among CF-based QoS prediction methods, latent factor (LF)-based models are more widely adopted [8, 9, 12–15, 17]. Originated from matrix factorization (MF) techniques [3, 10], an LF-based model works by building a low-rank approximation to the given user-service QoS matrix based on its known data only. It maps both users and services into the same low-dimensional LF space, trains desired LFs on the known data, and then predicts the missing data heavily relying on these resultant LFs [18].

Since LF-based model has the powerful ability on QoS prediction, the state-of-the-art methods in this area are primarily based on LF [8, 9, 12, 17]. They improve the basic LF-based models by identifying the neighborhoods of QoS data based on historical QoS records plus some additional geographical information. However, these geography-LF-based models have the following drawbacks:

(a) They adopt a common set on raw QoS data to identify the neighborhoods. Since the raw user-service QoS matrix can be very sparse, resultant common sets of users/services are commonly too small to identify the neighborhoods precisely. For example, Fig. 1 shows that many known data (red entries) are abandoned in finding the common sets among users, making the resultant neighborhoods lack reliability.
(b) They ignore the data reliability. Unreliable QoS data or called noises collected from malicious users (e.g., badmouthing a specific service) are often mixed up with the reliable ones [15]. Their QoS prediction accuracy would be impaired instead of being improved if they employ the unreliable QoS data.
(c) Additional geographical information can be difficult to gather in considering identity privacy, information security, and commercial interest. Moreover, geographical similarities can be influenced by unexpected factors like information facilities, routing policies, network throughput, and time of invocation.

To address the above drawbacks, this paper proposes a data-aware latent factor (DALF) model to achieve highly accurate QoS prediction. The main idea is to incorporate a density peaks based clustering method (DPClust) [19] into an LF model to discover the characteristics of QoS data, which can guide DALF to implement QoS prediction appropriately. The main contributions of this work include:

(a) We propose a method that can simultaneously identify a neighborhood for a user or a web service and detect the unreliable QoS data existed in the known ones.

(b) We theoretically analyze DALF and design its algorithm.
(c) We conduct detailed experiments on two benchmark real-world web service QoS datasets to evaluate DALF and compare it with the state-of-the-art models.

To the best of our knowledge, this work is never encountered in any previous works because (i) it can not only identify the neighborhoods but also detect the unreliable QoS data, (ii) it builds reliable neighborhoods solely based on a given QoS matrix but considering its full information, and (iii) it does not require any additional information.

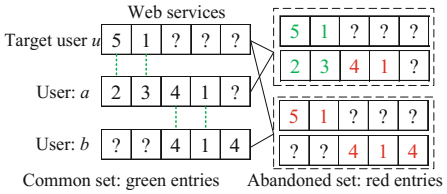

Fig. 1. The dilemma in building neighborhoods based on common sets defined on raw QoS data. (Color figure online)

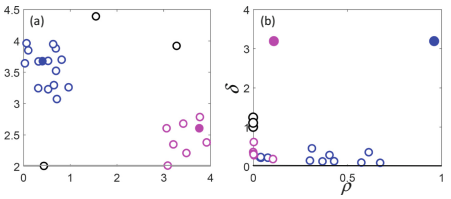

Fig. 2. The example of DPClust: (a) data distribution; (b) decision graph for data in (a); different colors correspond to different clusters. (Color figure online)

2 Preliminaries

2.1 LF Model

The QoS data is a user-service QoS matrix R defined as *Definition* 1 [9–11].

Definition 1. Given a user set U and a web service set S; let R be a $|U| \times |S|$ matrix where each element $r_{u,s}$ describes a user u's ($u \in U$) experience on a web service s ($s \in S$). R_K and R_U indicate the known and unknown entry sets of R respectively. R usually is a sparse matrix with $|R_K| \ll |R_U|$.

Definition 2. Given R, U, S, and f; given a $|U| \times f$ matrix P for U and an $f \times |S|$ matrix Q for S; $\hat{R}$ is R's rank-f approximation based on R_K under the condition of $f \ll \min(|U|, |S|)$. An LF model is to seek for P and Q to obtain $\hat{R}$ and error $\sum_{(u,s)\in RK} (r_{u,s} - \hat{r}_{u,s})^2$ is minimized. $\hat{R}$ is given by $\hat{R} = PQ$ where each element $\hat{r}_{u,s}$ is the prediction for each $r_{u,s}$ of R, $u \in U$ and $s \in S$. f is the dimension of LF space. P and Q are the LF matrices for users and web services respectively.

According to *Definition* 2, the loss function for LF model is [9–11]:

$$\arg\min_{P,Q} \varepsilon(P, Q) = \frac{1}{2} \sum_{(u,s)\in R_K} \left(r_{u,s} - \sum_{k=1}^{f} p_{u,k} q_{k,s} \right)^2 . \tag{1}$$

As analyzed in [9, 10, 18], it is important to integrate the Tikhonov regularization into (1) to improve its generality as follow:

$$\arg\min_{P,Q} \varepsilon(P, Q) = \frac{1}{2}\sum_{(u,s)\in R_K}\left(r_{u,s} - \sum_{k=1}^{f} p_{u,k}q_{k,s}\right)^2 + \frac{\lambda}{2}\sum_{(u,s)\in R_K}\left(\sum_{k=1}^{f} p_{u,k}^2 + \sum_{k=1}^{f} q_{k,s}^2\right), \tag{2}$$

where λ is the regularization controlling coefficient. By minimizing (2) with an optimizer, e.g., stochastic gradient descent (SGD), P and Q are extracted from R.

2.2 DPClust Algorithm

DPClust is a clustering algorithm based on the idea that cluster centers are characterized by a higher density than their neighbors and by a relatively large distance from data points with higher densities [19]. We employ DPClust to develop DALF because it can not only find the characteristics of data but also spotted the outliers.

Given a dataset $X = \{x_1, x_2, \ldots, x_G\}$, for each data point $x_i, i \in \{1, 2, \ldots, G\}$, its local density ρ_i is computed via cut-off kernel or Gaussian kernel. Cut-off kernel is as follow:

$$\rho_i = \sum_{j=1, j\neq i}^{N} \Phi(d_{i,j} - d_c), \qquad \Phi(t) = \begin{cases} 1 & t<0 \\ 0, & others \end{cases} \tag{3}$$

where $d_{i,j}$ is the distance between data points x_i and x_j and the number of all the $d_{i,j}$ is $G \times (G-1)/2$, and d_c is a cutoff distance with a fixed value. Gaussian kernel is as follow:

$$\rho_i = \sum_{j=1, j\neq i}^{G} e^{-\left(\frac{d_{i,j}}{d_c}\right)^2} \tag{4}$$

For a robust computing of ρ_i, d_c can be set as [19, 20]:

$$Vec = sort\left(d_{ij}\right),\ d_c = Vec(\lfloor P_{Vec} \times G \times (G-1)/2 \rfloor) \tag{5}$$

where Vec is a vector obtained by sorting all the $d_{i,j}$ in ascending order, P_{Vec} is a percentage denoting the average percentage of neighbors of all the data points. According to [19, 20], P_{Vec} is usually set around 1% to 2% as a rule of thumb.

For each data point x_i, δ_i is the minimum distance between x_i and any other data point with higher local density:

$$\delta_i = \begin{cases} \min_{j:\rho_i<\rho_j}(d_{i,j}), & others \\ \max_j(d_{ij}), & \forall j,\ \rho_i \geq \rho_j \end{cases} \tag{6}$$

Then, cluster centers are recognized as data points for which the value of ρ and δ are anomalously large.

Figure 2 is a simple example for illustrating DPClust. Figure 2(a) shows 26 data points embedded in the two-dimensional space. After computing ρ and δ for all the data points, the decision graph can be drawn in Fig. 2(b). Then, we can easily recognize the blue and pink solid data points as the cluster centers. Note that the three black hollow data points are the outliers and have a relatively small ρ and a large δ, which means that DPClust can also detect the outliers by computing outlier factor γ_i for each x_i as follow:

$$\gamma_i = \rho_i / \delta_i. \tag{7}$$

Formula (7) indicates that an outlier has an anomalously small value of γ.

3 The Proposed DALF Model

Figure 3 depicts the flowchart of DALF that has three parts. Part 1 is extracting LF matrices P for users and Q for services. Part 2 is identifying neighborhoods of QoS data and detecting unreliable QoS data by employing DPClust algorithm. Concretely, P is used to identify neighborhoods of users and detect unreliable users, and Q is used to identify neighborhoods of services and detect unreliable services. Part 3 is predicting the unknown entries in R based on Part 2. There are four prediction strategies in Part 3. The characteristics of QoS data will determine which one or more are appropriate to implement predictions. Next, we give the detailed descriptions on the three parts.

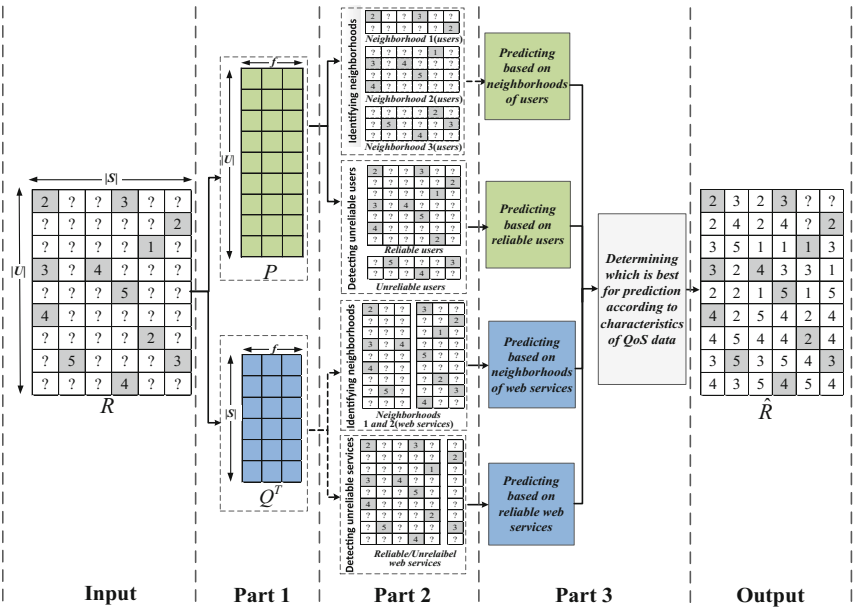

Fig. 3. Flowchart of the proposed DALF

3.1 Extracting LF Matrices for Users and Services

This part aims to extract LF matrices P for users and Q for services from R based on an LF model. We apply SGD to (2) to consider instant loss on a single element $r_{u,s}$:

$$\varepsilon_{u,s} = \frac{1}{2}\left(r_{u,s} - \sum_{k=1}^{f} p_{u,k} q_{k,s}\right)^2 + \frac{\lambda}{2}\left(\sum_{k=1}^{f} p_{u,k}^2 + \sum_{k=1}^{f} q_{k,s}^2\right) \tag{8}$$

Then, LFs involved in (8) are trained by moving them along the opposite of the stochastic gradient of (8) with respect to each single LF, i.e., we make

$$On\ r_{u,s},\ for\ k = 1 \sim f: \begin{cases} p_{u,k} \leftarrow p_{u,k} + \eta q_{k,s}\left(r_{u,s} - \sum_{k=1}^{f} p_{u,k} q_{k,s}\right) - \lambda \eta p_{u,k}, \\ q_{k,s} \leftarrow q_{k,s} + \eta p_{u,k}\left(r_{u,s} - \sum_{k=1}^{f} p_{u,k} q_{k,s}\right) - \lambda \eta q_{k,s}. \end{cases} \tag{9}$$

After LFs are trained on all the elements in R_K by computing (9), P and Q are extracted. For ease of formulation, we use the function (10) to represent extracting P and Q from R based on an LF model as follow.

$$\{P, Q\} = F^{LF}(P,\ Q|R) \tag{10}$$

3.2 Identifying Neighborhoods of QoS Data and Detecting Unreliable QoS Data

Since LF matrices P and Q respectively reflect the users and services characteristics hidden in R, we can identify neighborhoods of QoS data and detect unreliable QoS data based on them. Here, we use parameter α to denote the ratio of unreliable QoS data.

A. **With respect to users**

This section explains how to identify a neighborhood for a user and detect unreliable users based on P. Here P is seen as the dataset of users. For each user u, its local density ρ_u is computed via cut-off kernel as:

$$\rho_u = \sum_{u'=1, u' \neq u}^{|U|} \Phi\left(d_{u,u'} - d_U\right),\ \Phi(t) = \begin{cases} 1 & t < 0 \\ 0, & others \end{cases} \tag{11}$$

where d_U is the cutoff distance with respect to users, u' denotes another user that is different from user u, $d_{u,u'}$ denotes the distance between users u and u'. Here we compute $d_{u,u'}$ with Euclidean distance as:

$$d_{u,u'} = \sqrt{\sum_{k=1}^{f} \left(p_{u,k} - p_{u',k}\right)^2} \tag{12}$$

ρ_u also can be computed via Gaussian kernel as:

$$\rho_u = \sum_{u'=1, u' \neq u}^{|U|} e^{-\left(\frac{d_{u,u'}}{d_U}\right)^2} \tag{13}$$

According to (5), d_U is computed as:

$$Vec = sort\left(d_{u,u'}\right),\ d_U = Vec(\lfloor P_{Vec} \times |U| \times (|U| - 1)/2 \rfloor) \tag{14}$$

Then, the minimum distance δ_u of user u between itself and any other user with higher local density is computed as:

$$\delta_u = \begin{cases} \min_{u':\rho_u < \rho_{u'}}(d_{u,u'}), & others \\ \max_{u'}(d_{u,u'}), & \forall u',\ \rho_u \geq \rho_{u'} \end{cases} \tag{15}$$

Finally, the outlier factor γ_u of user u can be computed as:

$$\gamma_u = \rho_u / \delta_u \tag{16}$$

Based on all the ρ_u, δ_u, and γ_u, we can discover the user dataset P's clusters and outliers. Here clusters represent neighborhoods of users, outliers represent unreliable users. By computing (11)–(16), the original R can be separated into N matrices $\{R_1^U, R_2^U, \ldots, R_N^U\}$, where each matrix $R_n^U, n \in \{1, 2, \ldots, N\}$, denotes a neighborhood of users; or separated into two matrices $\{R_r^U, R_u^U\}$, where R_r^U denotes the reliable users and R_u^U denotes the unreliable users. Here the ratio of unreliable QoS data α is computed by:

$$\alpha = \left|R_u^U\right| / \left(\left|R_r^U\right| + \left|R_u^U\right|\right). \tag{17}$$

B. With respect to services

This section explains how to identify a neighborhood for a service and detect unreliable services based on Q. Here Q is seen as the dataset of services. For each service s, its local density ρ_s, minimum distance δ_s between itself and any other service with higher local density, and outlier factor γ_s can be computed by (18)–(23).

$$\rho_s = \sum_{s'=1, s' \neq s}^{|S|} \Phi\left(d_{s,s'} - d_S\right),\ \Phi(t) = \begin{cases} 1 & t < 0 \\ 0, & others \end{cases} \tag{18}$$

$$d_{s,s'} = \sqrt{\sum_{k=1}^{f} \left(q_{k,s} - q_{k,s'}\right)^2} \tag{19}$$

$$\rho_s = \sum_{s'=1, s' \neq s}^{|S|} e^{-\left(\frac{d_{s,s'}}{d_S}\right)^2} \tag{20}$$

$$Vec = sort\left(d_{s,s'}\right),\ d_S = Vec(\lfloor P_{Vec} \times |S| \times (|S| - 1)/2 \rfloor) \tag{21}$$

$$\delta_s = \begin{cases} \min_{s':\rho_s < \rho_{s'}}(d_{s,s'}), & others \\ \max_{s'}(d_{s,s'}), & \forall s',\ \rho_s \geq \rho_{s'} \end{cases} \tag{22}$$

$$\gamma_s = \rho_s / \delta_s \tag{23}$$

where s' is another service that is different from service s, $d_{s,s'}$ is the distance between services s and s', d_S is the cutoff distance for services. Similarly, the original R also can be separated into N matrices $\{R_1^S, R_2^S, \ldots, R_N^S\}$, where each matrix $R_n^S, n \in \{1, 2, \ldots, N\}$, denotes a neighborhood of services; or separated into two matrices $\{R_r^S, R_u^S\}$, where R_r^S and R_u^S denote the reliable and unreliable services respectively. Here, α is computed by

$$\alpha = \left|R_u^S\right| / \left(\left|R_r^S\right| + \left|R_u^S\right|\right). \tag{24}$$

3.3 Prediction

After Sect. 3.2, we can accurately predict the missing data in R by employing the four matrices sets of $\{R_1^U, R_2^U, \ldots, R_N^U\}$, $\{R_r^U, R_u^U\}$, $\{R_1^S, R_2^S, \ldots, R_N^S\}$, and $\{R_r^S, R_u^S\}$ respectively. Each matrices set can be used to implement the prediction, but which one is the best? This is determined by the characteristics of QoS data and please refer to Sect. 4.3 to see an example. Next, we respectively explain how to implement the prediction based on the four matrices sets and formula (10).

First, if matrices set $\{R_1^U, R_2^U, \ldots, R_N^U\}$ is used to predict, the computing formulas are

$$for\ n = 1 \sim N:\ \{P_n^U,\ Q_n^U\} = F^{LF}\left(P_n^U,\ Q_n^U \middle| R_n^U\right), \tag{25}$$

$$for\ n = 1 \sim N:\ \hat{R}_n^U = P_n^U Q_n^U, \tag{26}$$

$$\hat{R} = \hat{R}_1^U \cup \hat{R}_2^U \cup \ldots \cup \hat{R}_N^U. \tag{27}$$

Second, if matrices set $\{R_r^U, R_u^U\}$ is employed to predict, the computing formulas are

$$\{P_r^U, Q_r^U\} = F^{LF}\left(P_r^U, Q_r^U \middle| R_r^U\right), \tag{28}$$

$$\hat{R}_r^U = P_r^U Q_r^U, \tag{29}$$

$$\hat{R} = \hat{R}_r^U \oplus PQ|_{row}. \tag{30}$$

where $\hat{R}_r^U \oplus PQ|_{row}$ indicates that using $\hat{R}_r^U$ to replace the corresponding rows of the matrix product of PQ.

Third, if matrices set $\{R_1^S, R_2^S, \ldots, R_N^S\}$ is used to predict, the computing formulas are

$$for\ n = 1 \sim N:\ \{P_n^S, Q_n^S\} = F^{LF}\left(P_n^S, Q_n^S \middle| R_n^S\right), \tag{31}$$

$$for\ n = 1 \sim N:\ \hat{R}_n^S = P_n^S Q_n^S, \tag{32}$$

$$\hat{R} = \hat{R}_1^S \cup \hat{R}_2^S \cup \ldots \cup \hat{R}_N^S. \tag{33}$$

Fourth, if matrices set $\{R_r^S, R_u^S\}$ is employed to predict, the computing formulas are

$$\{P_r^S, Q_r^S\} = F^{LF}\left(P_r^S, Q_r^S \middle| R_r^S\right), \tag{34}$$

$$\hat{R}_r^S = P_r^S Q_r^S, \tag{35}$$

$$\hat{R} = \hat{R}_r^S \oplus PQ|_{column}. \tag{36}$$

where $\hat{R}_r^S \oplus PQ|_{column}$ indicates that using $\hat{R}_r^S$ to replace the corresponding columns of the matrix product of PQ.

3.4 Algorithm Design and Analysis

DALF relies on four algorithms. Algorithm 1 is extracting LF matrices (ELFM), Algorithm 2 is computing QoS data with respect to users (U-QoS), Algorithm 3 is computing QoS data with respect to services (S-QoS), and Algorithm 4 is Prediction. Their pseudo codes and time cost of each step are given in Algorithms 1–4. For Algorithms 1–3, their computational complexities are $\Theta(N_{mtr} \times |R_K| \times f)$, $\Theta\left(|U|^2 \times f\right)$, and $\Theta\left(|S|^2 \times f\right)$, respectively, where N_{mtr} is the maximum training round. For Algorithm 4, its computational complexity is $\Theta\left(\left(|U|^2 + |S|^2\right) \times f\right) + \Theta(N_{mtr} \times |R_K| \times f)$, which is the total computational complexity of DALF.

4 Experimental Results

4.1 Datasets

Two benchmark datasets, which are real-world web service QoS data collected by the WS-Dream system (https://github.com/wsdream/wsdream-dataset) and frequently used in prior researches [2, 3, 7–9, 12–15, 17], are selected to conduct the experiments. First dataset (D1) is the Response Time that contains 1,873,838 records and second dataset (D2) is the Throughput that contains 1,831,253 records. These records are generated by 339 users on 5,825 web services. For both two datasets, different test cases are designed to evaluate the performance of DALF. Table 1 summarizes the properties of all the test cases, where column 'Density' denotes the density of the training matrix.

Algorithm 1. ELFM

	Input: R; **Output:** P, Q	**Cost**
1	initializing f, λ, η, N_{mtr}=*max-training-round*	$\Theta(1)$
2	**while** $t \leq N_{mtr}$ && not converge	$\times N_{mtr}$
3	**for** each known entry $r_{u,s}$ in R // $r_{u,s} \in R_K$	$\times \lvert R_K \rvert$
4	**for** k=1 to f	$\times f$
5	computing $p_{u,k}$ according to (9)	$\Theta(1)$
6	computing $q_{k,s}$ according to (9)	$\Theta(1)$
7	**end for**	--
8	**end for**	--
9	t=t+1	$\Theta(1)$
10	**end while**	--
11	**return** P, Q	$\Theta(1)$

Algorithm 2. U-QoS

	Input: P, R; **Output:** $\{R_1^U, R_2^U, \ldots, R_N^U\}$, $\{R_r^U, R_e^U\}$	**Cost**
1	**for** u=1 to $\lvert U \rvert$	$\times \lvert U \rvert$
2	**for** u'= u+1 to $\lvert U \rvert$	$\times(\lvert U \rvert$-1)/
3	computing $d_{u,u'}$ according to (12)	$\Theta(f)$
4	**end for**	--
5	**end for**	--
6	computing d_U according to (14)	$\Theta(\lvert U \rvert^2)$
7	**for** u=1 to $\lvert U \rvert$	$\times \lvert U \rvert$
8	computing ρ_u according to (11) or (13)	$\Theta(\lvert U \rvert$-1
9	**end for**	--
10	**for** u=1 to $\lvert U \rvert$	$\times \lvert U \rvert$
11	computing δ_u according to (15)	$\Theta(\lvert U \rvert)$
12	computing γ_u according to (16)	$\Theta(1)$
13	**end for**	--
14	clustering P according to all the ρ_u and δ_u	$\Theta(1)$
15	separating R into $\{R_1^U, R_2^U, \ldots, R_N^U\}$ according to step 14	$\Theta(1)$
16	detecting unreliable users according to all γ_u	$\Theta(1)$
17	separating R into $\{R_r^U, R_e^U\}$ according to step 16	$\Theta(1)$
18	**return:** $\{R_1^U, R_2^U, \ldots, R_N^U\}$, $\{R_r^U, R_e^U\}$	$\Theta(1)$

Algorithm 3. S-QoS

	Input: Q, R; **Output:** $\{R_1^S, R_2^S, \ldots, R_N^S\}$, $\{R_r^S, R_e^S\}$	**Cost**
1	**for** s=1 to $\lvert S \rvert$	$\times \lvert S \rvert$
2	**for** s'= s+1 to $\lvert S \rvert$	$\times(\lvert S \rvert$-1)/2
3	computing $d_{s,s'}$ according to (19)	$\Theta(f)$
4	**end for**	--
5	**end for**	--
6	computing d_S according to (21)	$\Theta(\lvert S \rvert^2)$
7	**for** s=1 to $\lvert S \rvert$	$\times \lvert S \rvert$
8	computing ρ_s according to (18) or (20)	$\Theta(\lvert S \rvert$-1)
9	**end for**	--
10	**for** s=1 to $\lvert S \rvert$	$\times \lvert S \rvert$
11	computing δ_s according to (22)	$\Theta(\lvert I \rvert)$
12	computing γ_s according to (23)	$\Theta(1)$
13	**end for**	--
14	clustering Q according to all the ρ_s and δ_s	$\Theta(1)$
15	separating R into $\{R_1^S, R_2^S, \ldots, R_N^S\}$ according to step 14	$\Theta(1)$
16	detecting unreliable users according to all the γ_s	$\Theta(1)$
17	separating R into $\{R_r^S, R_e^S\}$ according to step 16	$\Theta(1)$
18	**return:** $\{R_1^S, R_2^S, \ldots, R_N^S\}$, $\{R_r^S, R_e^S\}$	$\Theta(1)$

Algorithm 4. Prediction

	Input: R; **Output:** $\hat{R}$	**Cost**
1	Calling **Algorithm 1**	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
2	Calling **Algorithm 2**	$\Theta(\lvert U \rvert^2 \times f)$
3	Calling **Algorithm 3**	$\Theta(\lvert S \rvert^2 \times f)$
4	determining which matrices set is best for prediction	$\Theta(1)$
5	**if** $\{R_r^U, R_2^U, \ldots, R_N^U\}$ is best for prediction	
6	**for** n=1 to N	
7	computing $\hat{R}_n^U$ according to (25) and (26)	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
8	**end for**	
9	computing $\hat{R}$ according to (27)	
10	**else if** $\{R_r^U, R_e^U\}$ is best for prediction	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
11	computing $\hat{R}$ according to (28)—(30)	
12	**else if** $\{R_r^S, R_2^S, \ldots, R_N^S\}$ is best for prediction	
13	**for** n=1 to N	
14	computing $\hat{R}_n^S$ according to (31) and (32)	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
15	**end for**	
16	computing $\hat{R}$ according to (33)	
17	**else** computing $\hat{R}$ according to (34)—(36)	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
18	**end if**	
19	**return:** $\hat{R}$	$\Theta(1)$

Table 1. Properties of all the designed test cases.

Dataset	No.	Density	Training data	Testing data
D1	D1.1	5%	93,692	1,780,146
	D1.2	10%	187,384	1,686,454
	D1.3	15%	281,076	1,592,762
	D1.4	20%	374,768	1,499,070
D2	D2.1	5%	91,563	1,739,690
	D2.2	10%	183,125	1,648,128
	D2.3	15%	274,689	1,556,564
	D2.4	20%	366,251	1,465,002

4.2 Evaluation Protocol

To evaluate the prediction quality of DALF, mean absolute error (MAE) is computed by $MAE = \left(\sum_{(w,j)\in\Gamma}\left|r_{w,j} - \hat{r}_{w,j}\right|_{abs}\right)/|\Gamma|$, where Γ denotes the testing set.

4.3 Prediction According to the Characteristics of QoS Data

This section illustrates how to predict the unknown QoS data according to its characteristics. First, extracting LF matrices P for users and Q for services on D1.4 and D2.4 respectively. Then, computing ρ, δ, and γ for each user or service. After that, the decision graphs for D1.4 and D2.4 can be drawn in Figs. 4 and 5. With respect to users, we observe that there are two cluster centers on D1.4 and three cluster centers on D2.4, which means that users of D1.4 and D2.4 could be separated into two and three neighborhoods respectively. With respect to services, there is only one cluster center on both D1.4 and D2.4, which means that there are no neighborhoods in services. Besides, we can find that there are many outliers (red rectangles) in both D1.4 and D2.4 with respect to services, which means that there are many unreliable services. Similar results are obtained on the other test cases. Thus, we conclude that predicting based on neighborhoods of users or reliable services is the best strategy for the eight test cases. In addition, we have conducted some experiments to verify that these two prediction strategies have better performance than the other two. For saving space, we never show their results.

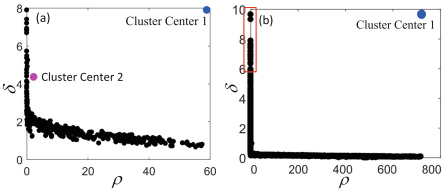

Fig. 4. The decision graph for D1.4 with respect to: (a) users, (b) services. (Color figure online)

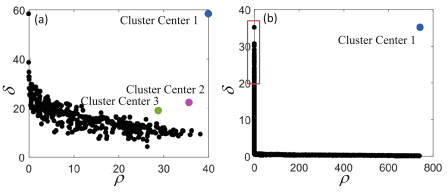

Fig. 5. The decision graph for D2.4 with respect to: (a) users, (b) services. (Color figure online)

4.4 Predicting Based on Neighborhoods of Users

A. Impacts of f

The parameters are set as $\eta = 0.01$ for D1, $\eta = 0.0001$ for D2, $\lambda = 0.01$, and $P_{Vec} = 2\%$, uniformly. Figure 6 shows the experimental results when f increases from 10 to 320. Since the higher dimension of LF space has better representation learning ability, DALF has a lower MAE as f increases. However, as f increases over 80, the MAE tends to decrease slightly or even increase. One reason is that with $f = 80$, DALF's representation learning ability is strong enough to precisely represent the test cases. As a result, continuous increase of f after 80 cannot bring significant improvement in prediction accuracy.

B. Impacts of λ

This set of experiments sets the parameters as $\eta = 0.01$ for D1, $\eta = 0.0001$ for D2, $f = 20$, and $P_{Vec} = 2\%$, uniformly. Figure 7 records the MAE as λ increases. We test a larger range of λ on D2 than on D1 because D2 has a much larger range of value than D1. The MAE decreases at first as λ increases in general on all the test cases. However, it then increases when λ grows over the optimal threshold, which means that DALF may be greatly impacted by the regularization terms.

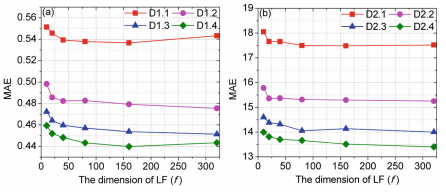

Fig. 6. MAE of DALF with different f predicting based on neighborhoods of users: (a) D1, (b) D2.

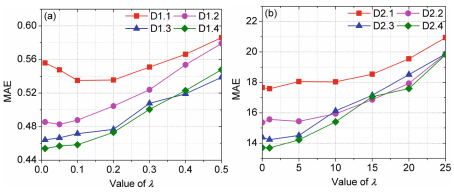

Fig. 7. MAE of DALF as λ increases predicting based on neighborhoods of users: (a) D1, (b) D2.

4.5 Predicting Based on Reliable Services

A. Impacts of α

The parameters are set as $\lambda = 0.01$, $\eta = 0.01$ for D1, $\eta = 0.0001$ for D2, $f = 20$, and $P_{Vec} = 2\%$, uniformly. Figure 8 is the measured MAE as α increases. On D1, the MAE decreases at first and then increases in general as α increases. The lowest MAE is obtained when α around to 0.3. On D2, the results are more complicated. Concretely, DALF has the lowest MAE when $\alpha = 0.05$ or 0.1. According to these observations, it seems that more services on D1 are detected as unreliable ones than that on D2. Overall, these results validate that the prediction accuracy of DALF can be improved by employing reliable services to train.

B. Impacts of f and λ

Since these results are very similar to that in Sects. 4.4(A) and (B), they are not described in detail for saving space. Please refer to Sects. 4.4(A) and (B).

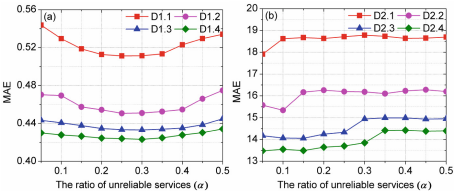

Fig. 8. MAE of DALF as α increases predicting based on reliable services: (a) D1, (b) D2.

Table 2. Descriptions of all the compared models.

Models	Descriptions
BLF	Basic LF model proposed in 2009 [18]
RSNMF	Improved LF-based model proposed in 2016 [3]
NIMF	Improved LF-based model proposed in 2013 [21]
NAMF	Geography-LF-based model proposed in 2016 [9]
GeoMF	Geography-LF-based model proposed in 2017 [8]
LMF-PP	Geography-LF-based model proposed in 2018 [12]
AutoRec	The DNN-based model proposed in 2015 [22]
DALF-1	Predicting based on neighborhoods of users
DALF-2	Predicting based on reliable services

Table 3. MAE of all the compared models on each test case.

TestCases	BLF	RSNMF	NIMF	NAMF	GeoMF	LMF-PP	AutoRec	DALF	
								DALF-1	DALF-2
D1.1	0.5561	0.5438	0.5502	0.5465	0.5305	0.5285	0.5467	0.5457	**0.5114**
D1.2	0.4944	0.4868	0.4842	0.4976	0.4827	0.4725	0.5055	0.4857	**0.451**
D1.3	0.4691	0.4492	0.4508	0.4625	0.4495	0.4472	0.4598	0.4642	**0.4331**
D1.4	0.4531	0.4371	0.4346	0.436	0.4366	0.426	0.4482	0.452	**0.4232**
D2.1	18.9776	21.4302	17.7153	22.736	24.7465	18.3091	21.3118	**17.6576**	17.9117
D2.2	16.1924	17.2305	15.5264	17.9148	22.4728	15.9125	17.031	**15.3595**	15.5734
D2.3	14.9278	14.6879	14.2146	15.9876	17.7908	14.745	15.0156	14.3836	**14.1739**
D2.4	14.3061	14.3654	13.5638	14.7462	16.2852	14.1033	14.2265	13.6697	**13.4772**

Table 4. Statistical results of prediction accuracy with a significance level of 0.05.

Comparison	DALF vs BLF	DALF vs. RSNMF	DALF vs. NIMF	DALF vs. NAMF	DALF vs. GeoMF	DALF vs. LMF-PP	DALF vs. AutoRec
p-value	**0.0039**	**0.0039**	**0.0039**	**0.0039**	**0.0039**	**0.0039**	**0.0039**

Note that the best one between DALF-1 and DALF-2 for each test case is selected to conduct statistical analysis

4.6 Comparisons Between DALF and Other Models

We compare DALF with state-of-the-art models on prediction accuracy and computational complexity. They are three LF-based models, three geography-LF-based models, and one deep neural network (DNN)-based model, and described in Table 2.

On prediction accuracy, the dimension of LF is set as $f = 20$ for all models except for AutoRec (because AutoRec is a DNN-based model) to conduct the fair comparisons. Besides, all other parameters of the compared models are set according to their original papers. There are two situations for DALF to conduct the comparisons. They are also marked in Table 2. Meanwhile, the other parameters for DALF are set as: $\alpha = 0.3$ and $\eta = 0.01$ for D1, $\alpha = 0.05$ and $\eta = 0.0001$ for D2, $\lambda = 0.01$, and $P_{Vec} = 2\%$, uniformly.

The compared results are shown in Table 3. Besides, the Wilcoxon signed-ranks test [23], as a nonparametric pairwise comparison procedure, is adopted to conduct statistical test. The results are recorded in Table 4. From them, we have two findings:

(a) DALF has significantly better prediction accuracy than the other models. For example, it has around 5.27%–17.15% lower MAE than AutoRec on all test cases.
(b) DALF-1 has much higher MAE than DALF-2 on D1, while they have similar MAE on D2. Figure 4 shows that neighborhoods of users are not very clear on D1 but clear on D2, and unreliable services can be easily detected on both D1 and D2. These findings mean that predicting based on neighborhoods of users is better for D2 than for D1, and predicting based on reliable services are appropriate for both D1 and D2.

On computational complexity, AutoRec is not compared because it is DNN-based model with extremely high computational cost [24]. Table 5 concludes the computational complexities for all the models, where K_1 and K_2 are the number of nearest neighbors for a user and for a service respectively. From it, we have two conclusions:

(a) BLF and RSNMF have lowest computational complexity because they are basic LF-based models and never consider neighborhood or unreliable factors of QoS data.
(b) DALF's computational complexity is lower than or at least comparable to that of the geography-LF-based models because f is much smaller than $|U|$ and $|S|$.

Table 5. The computational complexities of all the compared models.

Model	Complexity
BLF [18]	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
RSNMF [3]	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
NIMF [21]	$\Theta(\lvert U \rvert^2 \times \lvert S \rvert) + \Theta(N_{mtr} \times \lvert R_K \rvert \times f \times K_1^2)$
NAMF [9]	$\Theta(\lvert U \rvert^2) + \Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
GeoMF [8]	$\Theta(\lvert U \rvert^2 \times \lvert S \rvert + \lvert S \rvert^2 \times \lvert U \rvert) + \Theta(N_{mtr} \times \lvert R_K \rvert \times f^2 \times (K_1 + K_2))$
LMF-PP [12]	$\Theta(\lvert U \rvert^2 \times \lvert S \rvert + \lvert S \rvert^2 \times \lvert U \rvert) + \Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
DALF	$\Theta((\lvert U \rvert^2 + \lvert S \rvert^2) \times f) + \Theta(N_{mtr} \times \lvert R_K \rvert \times f)$

5 Conclusions

We propose a data-aware latent-factor (DALF) model to achieve highly accurate QoS prediction. The main idea is to incorporate a density peaks based clustering method (DPClust) into a latent factor (LF)-based model to improve the prediction accuracy. Empirical studies on two benchmark real-world web service QoS datasets demonstrate that: (i) DALF can discover the characteristics of QoS data only based on the user-service QoS matrix, (ii) DALF is a data-aware model because it can easily choose the appropriate strategies to implement prediction according to the characteristics of QoS data, and (iii) DALF has better performance than state-of-the-art models in QoS prediction. Finally, an open challenge for DALF is how to combine the two respects of users and services to further improve it. We plan to address this challenge in our future work.

Acknowledgments. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB1000900, and in part by the National Natural Science Foundation of China under Grants 61702475, 91646114 and 61772096.

References

1. Zheng, Z., Ma, H., Lyu, M.R., King, I.: WSRec: a collaborative filtering based web service recommender system. In: Proceeding of 2009 IEEE International Conference on Web Services, pp. 437–444. IEEE (2009)
2. Zheng, Z., Zhang, Y., Lyu, M.R.: Distributed QoS evaluation for real-world web services. In: Proceeding of 2010 IEEE International Conference on Web Services, pp. 83–90. IEEE (2010)
3. Luo, X., Zhou, M., Xia, Y., Zhu, Q., Ammari, A.C., Alabdulwahab, A.: Generating highly accurate predictions for missing QoS data via aggregating nonnegative latent factor models. IEEE Trans. Neural Netw. Learn. Syst. **27**(3), 524–537 (2016)
4. Geebelen, D., et al.: QoS prediction for web service compositions using kernel-based quantile estimation with online adaptation of the constant offset. Inf. Sci. **268**, 397–424 (2014)
5. Chen, X., Liu, X., Huang, Z., Sun, H.: RegionKNN: a scalable hybrid collaborative filtering algorithm for personalized web service recommendation. In: Proceeding of 2010 IEEE International Conference on Web Services, pp. 9–16. IEEE (2010)
6. Zheng, Z., Ma, H., Lyu, M.R., King, I.: Qos-aware web service recommendation by collaborative filtering. IEEE Trans. Serv. Comput. **4**(2), 140–152 (2011)
7. Lee, K., Park, J., Baik, J.: Location-based web service QoS prediction via preference propagation for improving cold start problem. In: Proceeding of 2015 IEEE International Conference on Web Services, pp. 177–184. IEEE (2015)
8. Chen, Z., Shen, L., Li, F., You, D.: Your neighbors alleviate cold-start: on geographical neighborhood influence to collaborative web service QoS prediction. Knowl.-Based Syst. **138**, 188–201 (2017)
9. Tang, M., Zheng, Z., Kang, G., Liu, J., Yang, Y., Zhang, T.: Collaborative web service quality prediction via exploiting matrix factorization and network map. IEEE Trans. Netw. Serv. Manag. **13**(1), 126–137 (2016)

10. Luo, X., Zhou, M., Li, S., You, Z., Xia, Y., Zhu, Q.: A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method. IEEE Trans. Neural Netw. Learn. Syst. **27**(3), 579–592 (2016)
11. Shi, Y., Larson, M., Hanjalic, A.: Collaborative filtering beyond the user-item matrix: a survey of the state of the art and future challenges. ACM Comput. Surv. **47**(1), 1–45 (2014)
12. Ryu, D., Lee, K., Baik, J.: Location-based web service QoS prediction via preference propagation to address cold start problem. IEEE Trans. Serv. Comput. (2018)
13. Wu, H., Yue, K., Li, B., Zhang, B., Hsu, C.-H.: Collaborative QoS prediction with context-sensitive matrix factorization. Future Gener. Comput. Syst. **82**, 669–678 (2018)
14. Zhu, J., He, P., Zheng, Z., Lyu, M.R.: Online QoS prediction for runtime service adaptation via adaptive matrix factorization. IEEE Trans. Parallel Distributed Syst. **28**(10), 2911–2924 (2017)
15. Wu, C., Qiu, W., Zheng, Z., Wang, X., Yang, X.: QoS prediction of web services based on two-phase k-means clustering. In: Proceeding of 2015 IEEE International Conference on Web Services, pp. 161–168. IEEE (2015)
16. Liu, A., et al.: Differential private collaborative Web services QoS prediction. World Wide Web 1–24 (2018, in Press)
17. Feng, Y., Huang, B.: Cloud manufacturing service QoS prediction based on neighbourhood enhanced matrix factorization. J. Intell. Manuf. 1–12 (2018)
18. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
19. Rodriguez, A., Laio, A.: Clustering by fast search and find of density peaks. Science **344** (6191), 1492–1496 (2014)
20. Wu, D., et al.: Self-training semi-supervised classification based on density peaks of data. Neurocomputing **275**, 180–191 (2018)
21. Zheng, Z., Ma, H., Lyu, M.R., King, I.: Collaborative web service QoS prediction via neighborhood integrated matrix factorization. IEEE Trans. Serv. Comput. **6**(3), 289–299 (2013)
22. Sedhain, S., Menon, A.K., Sanner, S., Xie, L.: AutoRec: autoencoders meet collaborative filtering. In: Proceedings of the 24th International Conference on World Wide Web, pp. 111–112 (2015)
23. Wu, D., Luo, X., Wang, G., Shang, M., Yuan, Y., Yan, H.: A highly accurate framework for self-labeled semi supervised classification in industrial applications. IEEE Trans. Ind. Inf. **14** (3), 909–920 (2018)
24. Zhou, Z.-H., Feng, J.: Deep forest: towards an alternative to deep neural networks. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence (2017)