A Data-Characteristic-Aware Latent Factor Model for Web Services QoS Prediction

Di Wu, *Member, IEEE*, Xin Luo, *Senior Member, IEEE*, Mingsheng Shang, Yi He, Guoyin Wang, *Senior Member, IEEE*, and Xindong Wu, *Fellow, IEEE*

Abstract—How to accurately predict unknown quality-of-service (QoS) data based on observed ones is a hot yet thorny issue in Web service-related applications. Recently, a latent factor (LF) model has shown its efficiency in addressing this issue owing to its high accuracy and scalability. An LF model can be improved by identifying user and service neighborhoods based on user and service geographical information. However, such information can be difficult to acquire in most applications with the considerations of information security, identity privacy, and commercial interests in a real system. Besides, the existing LF model-based QoS predictors mostly ignore the reliability of given QoS data where noises commonly exist to cause accuracy loss. To address the above issues, this paper proposes a data-characteristic-aware latent factor (DCALF) model to implement highly accurate QoS predictions, where 'data-characteristic-aware' indicates that it can appropriately implement QoS prediction according to the characteristics of given QoS data. Its main idea is two-fold: a) it detects the neighborhoods and noises of users and services based on the dense LFs extracted from the original sparse QoS data, b) it incorporates a density peaks-based clustering method into its modeling process for achieving the simultaneous detections of both neighborhoods and noises of QoS data. With such designs, it precisely represents the given QoS data in spite of their sparsity, thereby achieving highly accurate predictions for unknown ones. Experimental results on two QoS datasets generated by real-world Web services demonstrate that the proposed DCALF model outperforms state-of-the-art QoS predictors, making it highly competitive in addressing the issue of Web service selection and recommendation.

Index Terms—Web Service, quality-of-service, QoS, latent factor analysis, density peak, data-characteristic-aware, missing data, big data, topological neighborhood, noise data, service selection, data science

1 Introduction

Web service is a software component used to exchange data between two software systems over a network [1], [2]. In the era of the World Wide Web, more and more service providers serve their customers through Web service, resulting in a rapid expansion of online services [3], [4]. Unsurprisingly, among the huge number of Web services, many of them provide similar functionality to users. In such a context, how to select appropriate Web services from a huge candidate set and recommend them to potential users becomes a hot yet thorny issue [5], [6], [7].

Quality-of-Service (QoS), which describes Web services' non-functional characteristics (e.g., response time and throughput) [8], [9], is recognized as an important criterion to evaluate the quality of Web services with similar functionality [1], [3], [10], [11]. With reliable QoS data, the optimal Web services can be efficiently selected and recommended to potential users. Warming-up test is often adopted to acquire QoS data. However, it is expensive and time-consuming [5], [12], [13]. As a result, generating predictions for unknown QoS data based on historical ones becomes highly important [12], [13], [14], [15], [16].

Collaborative filtering (CF) [17], [18], [19], [20], [21], [22], [23], [24], [25] is one of the most popular and successful approaches to QoS prediction [12], [13], [14], [15], [16], [26], [27], [28], [29], [30], [31]. A CF-based QoS predictor is commonly defined on a user-service QoS matrix [12], [13], [14], [15], [16], [26], [27], [28], [29], [30], [31], where each column denotes a specified service, each row denotes a specified user, and each entry denotes an observed QoS record (e.g., response time and throughput) produced by a specified user's invoking on a specified service. Since the number of services can be huge in a real system, it becomes impossible for a user to invoke all provided services. As a result, such a user-service QoS matrix is highly sparse with numerous

- *Di Wu is with the Chongqing Engineering Research Center of Big Data Application for Smart Cities, and Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China and also with the University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: wudi@cigit.ac.cn.*
- *Xin Luo is with the School of Computer Science and Technology, Dongguan University of Technology, Dongguan, Guangdong 523808, China and also with the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China. E-mail: luoxin21@gmail.com.*
- *Mingsheng Shang is with the Chongqing Engineering Research Center of Big Data Application for Smart Cities, and Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China. E-mail: msshang@cigit.ac.cn.*
- *Yi He is with the University of Louisiana at Lafayette, Lafayette, LA 70503 USA. E-mail: yi.he1@louisiana.edu.*
- *Guoyin Wang is with the Department of Chongqing Key Laboratory of Computational Intelligence, Chongqing University of Posts and Telecommunications, Chongqing 400065, China. E-mail: wanggy@cqupt.edu.cn.*
- *Xindong Wu is with the Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Hefei 230009, China and also with the Mininglamp Academy of Sciences, Mininglamp Technology, Beijing 100084, China. E-mail: xwu@hfut.edu.cn.*

Manuscript received 3 Dec. 2018; revised 3 July 2020; accepted 31 July 2020. Date of publication 5 Aug. 2020; date of current version 29 Apr. 2022. (Corresponding author: Xin Luo.)
Recommended for acceptance by S. Sadiq.
Digital Object Identifier no. 10.1109/TKDE.2020.3014302

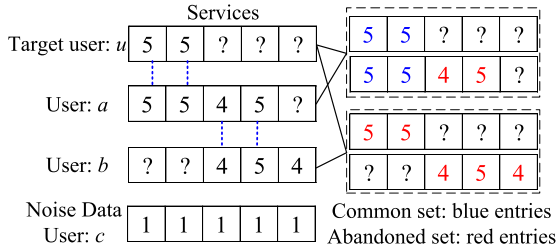


FIG. 1. Unreliable neighborhoods achieved on raw QoS data.

missing entries. Hence, a CF-based QoS predictor works by analyzing such a sparse user-service QoS matrix to predict its missing data based on its observed ones.

Among various CF-based QoS predictors, a latent factor (LF) model-based one is widely investigated and adopted owing to its high scalability and accuracy [15], [16], [26], [27], [28], [29], [31], [32], [33], [34]. Currently, state-of-the-art QoS predictors are primarily based on an LF-based model [15], [16], [26], [31]. They improve the base LF-based model by identifying the neighborhoods of users and services on the historical QoS records plus additional geographical information. However, they suffer the following limitations:

(a) They mostly identify the neighborhoods based on the common sets defined on the raw sparse user-service QoS matrix, which can be inaccurate. For example, Fig. 1 depicts that many observed QoS records (red entries) are abandoned when finding the common sets among users, which makes target user u and user b have no common QoS records. Hence, although users u and b should be similar owing to their similarity with user a in common, such similarity cannot be correctly identified on the raw sparse user-service QoS matrix.
(b) They directly adopt the given QoS data to implement QoS prediction without the considerations of their reliability. Unfortunately, noises (unreliable QoS data) caused by invoking failures or accidents usually exist [29], [35]. For example, the historical records of user c shown in Fig. 1 are noises, which can greatly impair the performance of a resultant model.
(c) They utilize additional geographical information to identify the geographical neighborhoods of users and services. However, geographical information can be unavailable when considering identity privacy, information security, and commercial interest. Besides, geographical neighborhoods can be influenced by unexpected factors like information facilities, routing policies, network throughput, and time of invocation.

To address the above limitations, this paper proposes a data-characteristic-aware latent factor (DCALF) model. It has the following specific designs:

(a) It detects the neighborhoods and noises of QoS data on the dense LFs extracted from the original sparse QoS data,
(b) It incorporates a density peaks-based clustering (DPClust) [36] method into its modeling process to detect both neighborhoods and noises from QoS data simultaneously, and
(c) It appropriately implements QoS prediction according to the characteristics of QoS data.

With such the designs, a DCALF model precisely represents a sparse user-service QoS matrix to achieve highly accurate QoS prediction. Main contributions of this work include:

(a) A DCALF model that precisely identifies the neighborhoods among users/services based on QoS data only, as well as achieves highly accurate predictions for unobserved QoS data;
(b) Algorithm design and analyses for a DCALF model; and
(c) Detailed empirical studies conducted on two real QoS datasets demonstrate that a DCALF model is highly efficient in addressing the problem of QoS prediction.

Note that a DCALF model is significantly different from state-of-the-art QoS predictors in the following aspects:

(a) It detects unreliable QoS data hidden in given ones,
(b) It precisely models the user/service neighborhoods based on QoS data only without any information loss, and
(c) It is not limited by any additional information.

To the authors' best knowledge, such efforts are never encountered in any previous study.

Section 2 gives the preliminaries. Section 3 presents a DCALF model. Section 4 provides experimental results and analyses. Section 5 discusses related studies. Finally, Section 6 concludes this paper.

2 Preliminaries

2.1 Problem of a User-Dependent QoS Predictor

This section formally defines a user-dependent QoS predictor. A small example is illustrated in Fig. S1 in the Supplementary File of this paper, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/tkde.2020.3014302. As shown in Fig. S1(a) available online, five services (s_1, s_2, s_3, s_4, s_5) are invoked by four users (u_1, u_2, u_3, u_4) with the solid lines representing users' observed invocations on corresponding services. Such invocation history can be modeled into a user-service QoS matrix R as shown in Fig. S1(b) available online. Note that each known entry in R denotes the QoS record of a specific user-dependent QoS metric (e.g., response time) by a specific user on a specific service, while the question marks indicate that the corresponding user-service QoS records have not been observed yet. Hence, a QoS predictor is essentially built on R's known data set to predict its unknown data as shown in Fig. S1(c) available online, which can be defined as in [33], [34]:

Definition 1 Problem of a User-Dependent QoS Predictor. *Given a user set* U *and a service set* S. *A user-service QoS matrix* $R^{|U|\times|S|}$ *has its each element* $\mathrm{r_{u,s}}$ *describe user* u's $(u \in U)$ *historical QoS record of a specific user-dependent QoS metric on service* s $(s \in S)$. *Let* $\mathrm{R_K}$ *and* $\mathrm{R_U}$ *respectively denote* R's *known and unknown entry sets. A user-dependent QoS predictor aims to predict* $\mathrm{R_U}$ *based on* $\mathrm{R_K}$ *as precisely as possible.*

With Definition 1 and inferences in Section 2.1, next we explicitly define the problem of a user-dependent QoS predictor, which is the main concern of this study.

2.2 An LF-Based QoS Predictor

An LF model is widely adopted to implement a user-dependent QoS predictor owing to its high scalability and accuracy [15], [16], [26], [27], [28], [29], [31], [32], [33], [34]. Thus, an LF-based QoS predictor can be defined as:

Definition 2 An LF-Based QoS Predictor. *Given a user-service QoS matrix $R^{|U|\times|S|}$ and the LF dimension* f, *an LF model builds LF matrices $P^{|U|\times f}$ and $Q^{f\times|S|}$ to achieve* R*'s rank-*f *approximation $\hat{R}$by minimizing the distance between* R *and $\hat{R}$on* $\mathrm{R_K}$ *with $\hat{R} = PQ$.*

Note that P and Q are actually interpreted as the user and item LF matrices, respectively. From Definition 2, we see that an LF model works by minimizing a certain distance between R and $\hat{R}$on R_K. With the commonly adopted Euclidean euclidean distance, such a minimization objective is formulated by [16], [17], [18]:

$$\begin{aligned}&\arg\min_{P,\,Q}\varepsilon(P,\ Q)\\&=\frac{1}{2}\sum_{(u,s)\in R_K}\left(r_{u,s}-\hat{r}_{u,s}\right)^2=\frac{1}{2}\sum_{(u,s)\in R_K}\left(r_{u,s}-\sum_{k=1}^{f}p_{u,k}q_{k,s}\right)^2,\end{aligned}\tag{1}$$

where $p_{u,k}$ and $q_{k,s}$ denote specific entries in P and Q, respectively. According to prior research [16], [17], [37], it is important to integrate regularization terms like an L_2 norm-based one into (1) to improve its generality:

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

where $\boldsymbol{\lambda}$ is the regularization coefficient.

With Definition 2 and inferences in Section 2.2, we define an LF-based QoS predictor and its benchmark learning objective. In this study, we aim to address the problem of a user-dependent QoS predictor via building an extended LF-based QoS predictor.

2.3 A DPClust Algorithm

A DPClust algorithm characterizes its cluster centers according to data density [36]. Given a dataset $X = \{x_1, x_2, \ldots, x_G\}$, for each data point x_i where $i \in \{1, 2, \ldots, G\}$, its local density $\boldsymbol{\rho}_i$ is given by a kernel function like a cut-off or a Gaussian kernel. With the former, $\boldsymbol{\rho}_i$ is given as:

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

where $d_{i,j}$ is the distance between x_i and x_j, d_c is the cutoff constant, respectively. On the other hand, with a Gaussian kernel $\boldsymbol{\rho}_i$ is formulated by:

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

Note that for robustness [36], d_c is set as follows.

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

where *Vec* is a vector obtained by sorting all $d_{i,j}$ in ascending order, P_{Vec} is the percentage denoting the average percentage of neighbors of all data points. According to [36], P_{Vec} is usually set in the [1%, 2 percent%] interval.

For each x_i, $\boldsymbol{\delta}_i$ is the minimum distance between x_i and any other data point with higher local density:

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

Thus, cluster centers are recognized as the data points with anomalously large $\boldsymbol{\rho}$ and $\boldsymbol{\delta}$.

Fig. 2 gives a simple example of DPClust. Fig. 2a shows 26 data points embedded in a two-dimensional space. After computing $\boldsymbol{\rho}$ and $\boldsymbol{\delta}$ for all data points, the decision graph [36] that is the plot of $\boldsymbol{\delta}$ as a function of $\boldsymbol{\rho}$ for each data point can be drawn as in Fig. 2b. Then the DPClust algorithm recognizes the blue solid and the pink solid points in Fig. 2b as cluster centers. The three black hollow data points have a relatively small $\boldsymbol{\rho}$ and a large $\boldsymbol{\delta}$ because they are outliers. Thus, a DPClust algorithm is also able to detect outliers by computing outlier factor $\boldsymbol{\gamma}_i$ for each x_i,

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

Note that (7) indicates that an outlier has an anomalously small value of $\boldsymbol{\gamma}$.

In this section, we briefly introduce the principle and properties of a DPClust algorithm. For more details please refer to [36]. To be shown next, the proposed DCALF model will adopt a DPClust algorithm to identify the user/service neighbors and outliers based on LFs extracted from a sparse user-service QoS matrix, thereby achieving highly accurate predictions for missing user-dependent QoS data.

3 A DCALF Model

Fig. 3 depicts the flowchart of the proposed DCALF model, where a DCALF model essentially consists of three steps. Step 1 is to extract LF matrices P for users and Q for services from a given R with high sparsity. Step 2 is to identify neighborhoods of QoS data and detect unreliable QoS data with a DPClust algorithm. Specially, P is adopted to identify neighborhoods of users and detect unreliable users, and Q is adopted to address the same tasks on services, respectively. Step 3 is to predict the unobserved QoS data in R. Next we present these three steps in detail.

3.1 Step 1: LF Extraction

In this part, we aim to extract P and Q based on R_K. To do so, we apply SGD to (2). Considering the instant loss on a single element $r_{u,s}$, we have:

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

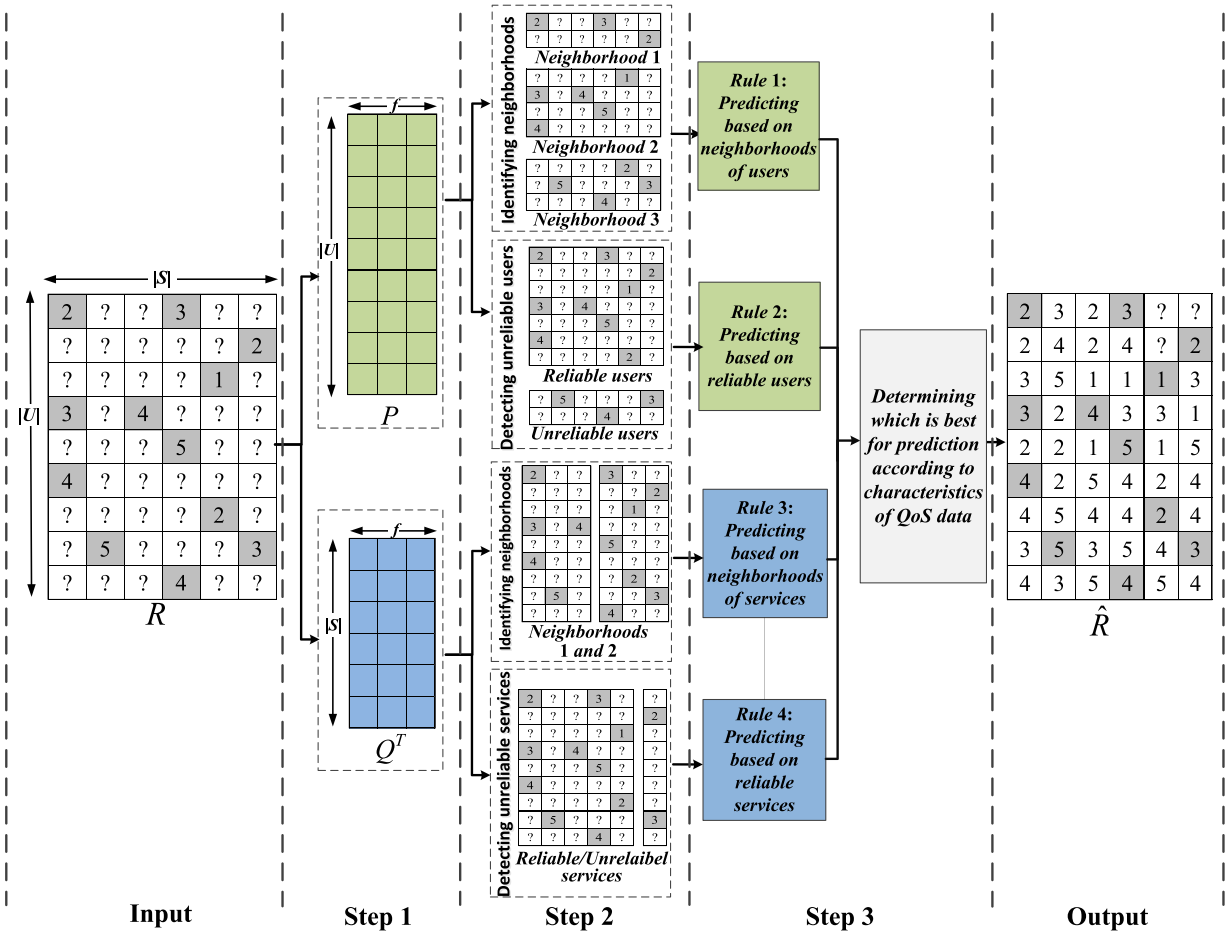


FIG. 2. Flowchart of a DCALF model.

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 as follows:

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

By repeating (9) on R_K iteratively to achieve a stationary point, P and Q are extracted from R.

3.2 Step 2: Neighborhood and Outlier Detection

Since P and Q respectively describe user and service characteristics hidden in R_K, we can identify neighborhoods of involved users and services as well as to detect unreliable QoS data based on them. We adopt the parameter $\boldsymbol{\alpha}$ to denote the ratio of unreliable QoS data.

3.2.1 User Neighborhoods and Outliers

With extracted P, user u's local density $\boldsymbol{\rho}_u$ is computed via a cut-off kernel as follows,

$$\rho_u = \sum_{u'=1,u'\neq u}^{|U|} \Phi\left(d_{u,u'} - d_U\right),\ \Phi(t) = \begin{cases} 1 & t<0 \\ 0, & otherwise, \end{cases} \tag{10}$$

where d_U is the cutoff constant 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', respectively. Here we compute $d_{u,u'}$ with the Euclidean euclidean distance by:

$$d_{u,u'} = \sqrt{\sum_{k=1}^{f}\left(p_{u,k} - p_{u',k}\right)^2}. \tag{11}$$

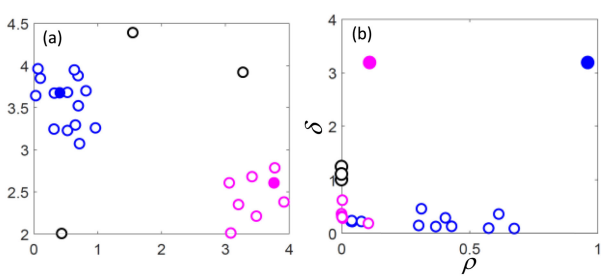


Fig. 3. The example of DPClust: (a) data distribution; (b) decision graph for data in (a); different colors correspond to different clusters.

Note that $\boldsymbol{\rho}_u$ also can be computed via a Gaussian kernel by:

$$\rho_u = \sum_{u'=1,u'\neq u}^{|U|} e^{-\left(\frac{d_{u,u'}}{d_U}\right)^2} \tag{12}$$

According to (5), d_U is computed by:

$$Vec = sort\big(\ d_{u,u'}\big),\ d_U = Vec(\lfloor P_{Vec} \times |U| \times (|U|-1)/2 \rfloor). \tag{13}$$

Then the minimum distance $\boldsymbol{\delta}_u$ of user u between itself and any other user with higher local density is computed by:

$$\delta_u = \begin{cases} \max_{u'}(d_{u,u'}), & \forall u',\ \rho_u \geq \rho_{u'}, \\ \min_{u':\rho_u<\rho_{u'}}(d_{u,u'}), & otherwise. \end{cases} \tag{14}$$

Finally, the outlier factor $\boldsymbol{\gamma}_u$ of user u is computed by:

$$\gamma_u = \rho_u/\delta_u. \tag{15}$$

As discussed in Section 2.2, user clusters and outliers can be detected via computing ρ_u, δ_u, and $\gamma_u, \forall u \in U$. Note that we let different clusters denote different user neighborhoods, and outliers represent unreliable users. Thus, we can identify user neighborhoods and unreliable users based on P by (10), (11), (12), (13), (14), (),(15). After that, the original R can be separated into N small matrices $\{RU\,1, RU\,2, \ldots, RU\,N\}$ representing user neighborhoods, or separated into two matrices $\{RU\,r, RU\,u\}$, where $RU\,r$ and $RU\,u$ respectively denote the reliable and unreliable users. Here the ratio of unreliable QoS data $\boldsymbol{\alpha}$ is computed by:

$$\alpha = \big|R_u^U\big|/\big(\big|R_r^U\big| + \big|R_u^U\big|\big). \tag{16}$$

3.2.2 Service Neighborhoods and Outliers

After extracting Q following Section 3.1, for each service s, its local density $\boldsymbol{\rho}_s$ is computed via a cut-off kernel and a Gaussian kernel as follows,

$$\rho_s = \sum_{s'=1,s'\neq s}^{|S|} \Phi\big(d_{s,s'} - d_S\big),\ \Phi(t) = \begin{cases} 1 & t<0 \\ 0, & others \end{cases} \tag{17}$$

$$\rho_s = \sum_{s'=1,s'\neq s}^{|S|} e^{-\left(\frac{d_{s,s'}}{d_S}\right)^2}, \tag{18}$$

where s' is another service different from s, $d_{s,s'}$ is the distance between services s and s', and d_S is the cutoff distance for services, respectively. With the Euclidean euclidean distance, d_S is given as:

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

$$Vec = sort\big(\ d_{s,s'}\big),\ d_S = Vec(\lfloor P_{Vec} \times |S| \times (|S|-1)/2 \rfloor). \tag{20}$$

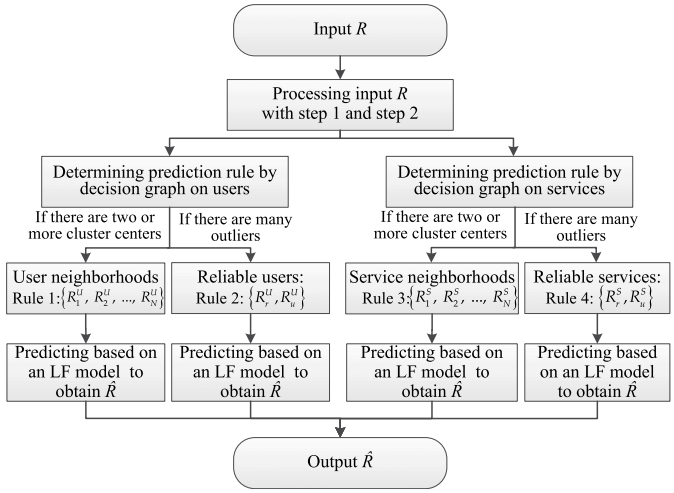


Fig. 4. Flowchart of prediction rule selection in a DCALF model.

So the minimum distance $\boldsymbol{\delta}_s$ between itself and any other service with higher local density is computed by:

$$\delta_s = \begin{cases} \max_{s'}(d_{s,s'}), & \forall s',\ \rho_s \geq \rho_{s'}, \\ \min_{s':\rho_s<\rho_{s'}}(d_{s,s'}), & otherwise. \end{cases} \tag{21}$$

Then, the outlier factor $\boldsymbol{\gamma}_s$ is computed by:

$$\gamma_s = \rho_s/\delta_s. \tag{22}$$

Finally, the neighborhoods of services and unreliable services can be detected based on $\boldsymbol{\rho}_s$, $\boldsymbol{\delta}_s$, and $\boldsymbol{\gamma}_s$. More specifically, R can be separated into N matrices $\{RS\,1, RS\,2, \ldots, RS\,N\}$ representing service neighborhoods or two matrices $\{RS\,r, RS\,u\}$ representing reliable and unreliable services. In a service-oriented perspective, the ratio of unreliable QoS data $\boldsymbol{\alpha}$ is computed by

$$\alpha = \big|R_u^S\big|/\big(\big|R_r^S\big| + \big|R_u^S\big|\big). \tag{23}$$

3.3 Step 3: Prediction

Based on Steps 1-2, we can accurately predict the missing data in R based on obtained $\{RU\,1, RU\,2, \ldots, RU\,N\}, \{RU\,r, RU\,u\}, \{RS\,1, RS\,2, \ldots, RS\,N\}$, and $\{RS\,r, RS\,u\}$. Note that each matrix set can be adopted to achieve separate prediction rules. However, which one should be adopted? In this study, we choose a prediction rule according to the characteristics of the given data.

Fig. 4 depicts the flowchart of determining a prediction rule for a DCALF model. In the beginning, we process the input R with steps 1 and 2. Afterward, we determine the prediction rule based on the decision graphs from both perspectives of users and services. In the following, we provide a small example to illustrate this decision process.

Fig. S2 in the Supplementary File of this paper available online illustrates the four kinds of decision graph corresponding to four prediction rules. Considering the user perspective, if there are two or more cluster centers as shown in Fig. S2(a) available online, then there are two or more neighborhoods of users. Thus, we choose prediction rule 1, i.e., predicting based on neighborhoods of users. If there are

many outliers (red rectangles) as shown in Fig. S2(b) available online, then there are many unreliable users. Thus, we choose prediction rule 2, i.e., predicting based on reliable users. On the other hand, the decision process is similar when considering from the service perspective. As shown in Fig. S2(c) available online, if there are two or more cluster centers (service side), we choose prediction rule 3 to work based on service neighborhoods. As shown in Fig. S2(d) available online, if there are many outliers (red rectangles, service side), we choose prediction rule 4 to work based on reliable services.

Next, we give the computing formulas for the four prediction rules. Note that we adopt the following general function to represent the extraction of P and Q from R,

$$\{P, Q\} = F^{LF}(P,\ Q|R). \tag{24}$$

Prediction Rule 1. Predicting based on neighborhoods of users. We adopt the set of $\{RU\,1, RU\,2, \ldots, RU\,N\}$ to obtain $\hat{R}$as follows,

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

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

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

Prediction Rule 2. Predicting based reliable users. We adopt the set of $\{RU\,r, RU\,u\}$ to obtain $\hat{R}$in the following form,

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

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

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

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

Prediction Rule 3. Predicting based on neighborhoods of services. We adopt the set of $\{RS\,1, RS\,2, \ldots, RS\,N\}$ to obtain $\hat{R}$as follows,

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

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

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

Prediction Rule 4. Predicting based reliable services. We adopt the set of $\{RS\,r, RS\,u\}$ to obtain $\hat{R}$as follows,

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

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

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

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

3.4 Algorithm Design and Analysis

3.4.1 Algorithms

Note that A DCALF model relies on four algorithms, i.e., Algorithm 1 that extracts LF matrices (ELFM), Algorithm 2 that identifies user neighborhoods and unreliable users (INU-DUU), Algorithm 3 that identifies service neighborhoods and unreliable services (INS-DUS), and Algorithm 4 that generates predictions. Note that Algorithms 1—4 respectively correspond to methods mentioned in Sections 3.1, 3.2.1, 3.2.2, and 3.3. They are designed in Tables S1—S4 in the Supplementary File of this paper available online.

Considering Algorithm 1, its computational complexity is

$$\begin{aligned} T &= \Theta(1) + N_{mtr} \times (|R_K| \times f \times 2 \times \Theta(1) + \Theta(1)) + \Theta(1) \\ &\approx \Theta(N_{mtr} \times |R_K| \times f), \end{aligned} \tag{37}$$

where N_{mtr} is the max-training-round. Then the complexity of Algorithm 2 is given by

$$\begin{aligned} T &= |U| \times (|U| - 1)/2 \times \Theta(f) + \Theta(|U|^2) \\ &\quad + |U| \times \Theta(|U| - 1) + |U| \times \Theta(|U| + 1) + 5\Theta(1) \\ &\approx \Theta(|U|^2 \times f). \end{aligned} \tag{38}$$

For Algorithm 3, its complexity is formulated by

$$\begin{aligned} T &= |S| \times (|S| - 1)/2 \times \Theta(f) + \Theta(|S|^2) \\ &\quad + |S| \times \Theta(|S| - 1) + |S| \times \Theta(|S| + 1) + 5\Theta(1) \\ &\approx \Theta(|S|^2 \times f). \end{aligned} \tag{39}$$

Thus, based on (37), (38), (39), Algorithm 4's computational complexity is formulated by

$$\begin{aligned} T &= \Theta(N_{mtr} \times |R_K| \times f) + \Theta(|U|^2 \times f) + \Theta(|S|^2 \times f)\Theta(1) \\ &\quad + \Theta(N_{mtr} \times |R_K| \times f) + \Theta(1) \\ &\approx \Theta\Big(\big(|U|^2 + |S|^2\big) \times f\Big) + \Theta(N_{mtr} \times |R_K| \times f). \end{aligned} \tag{40}$$

Note that (40) denotes DCALF's complexity since the costs of Algorithms 1-3 are included in that of Algorithms 4.

3.4.2 An Illustrative Example

To illustrate how DCALF implements QoS prediction accurately based on Algorithms 1-4, we give an illustrative example as shown in Fig. 5a. Given the user-service QoS matrix which is the same as that in Fig. 1, we first adopt Algorithm 1 to extract LF matrix P based on all the observed QoS records of the given matrix. Here we set $f = 2$. Then, we draw the 2D distribution of the extracted LF matrix P, as shown in Fig. 5b, where we find that users u, a, and b are close and user c is far away from them, which means that the extracted P well represents the characteristics hidden in the QoS records.

Next, we put P into Algorithm 2 to identify user neighborhoods and unreliable users. The decision graph of P is shown in Fig. 5c, where we find that users u, a, and b belong to a cluster and user c is an outlier. Hence, based on Algorithms 1-2, DCALF identifies that user b and target user u are similar while user c is unreliable.

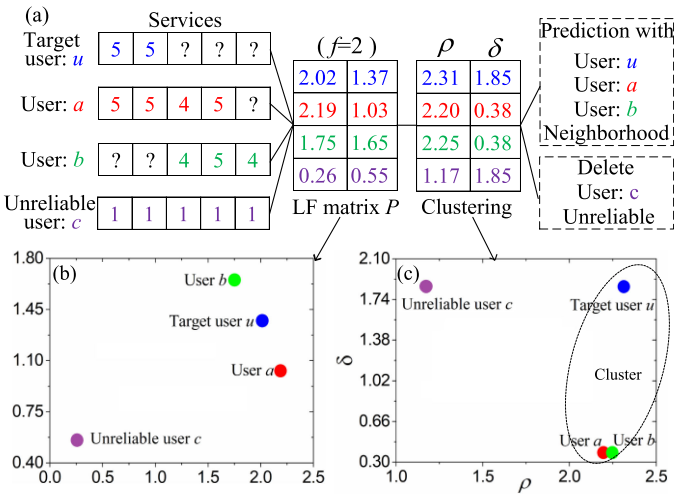


Fig. 5. A small example of making QoS predictions by DCALF, (a) illustration, (b) 2D distribution of P, and (c) decision graph of P.

Finally, we combine users u, a, and b as a neighborhood and put them into Algorithm 4 to implement QoS prediction. In this example, DCALF is modeled with respect to users. If we model it with respect to services, we adopt Algorithm 3 instead of Algorithm 2, thereby identifying service neighborhoods and unreliable services based on Q.

From this example, we see that DCALF can correctly identify the neighborhood of users u, a, and b and detect unreliable user c, which cannot be achieved by prior models developed on common sets as shown in Fig. 1. The main reason is that DCALF discovers the neighborhoods and the unreliable ones of QoS data based on the dense LFs extracted from all observed data in a sparse user-service QoS matrix. From (40), we see that DCALF has an extra computational burden caused by its steps 1 and 2 when compared with a base LF model. According to [38], [39], an LF model can be parallelized by using an alternating SGD algorithm. Hence, it is expected that DCALF's computational efficiency can be further improved by parallelization with alternating SGD. We will address this issue in the future.

4 Experiments and Results

4.1 Datasets

Two benchmark datasets, which are real-world Web service QoS data collected by the WS-Dream system and frequently used in prior studies [3], [5], [14], [15], [16], [26], [27], [28], [29], [31], are selected to conduct the experiments. The first dataset (D1) is the Response Time that contains 1,873,838 records and the second one (D2) is the Throughput that contains 1,831,253 records. These records are generated by 339 users on 5,825 services. For both two datasets, different testing cases are designed to validate the involved models' performance. Table 1 summarizes the properties of all testing cases, where the column 'Density' is computed by $(|\Lambda|/(|U| \times |S|)) \times 100\%$ with Λ consisting of the training data. A cross-validation strategy is employed to conduct the experiments more objectively.

4.2 Evaluation Protocol

In QoS prediction, the main task is to predict the unobserved QoS data based on the observed ones. According to the predicted QoS data, we can know the users' unobserved experiences on services. Hence, this work mainly cares about the closeness of prediction to the ground truth. To evaluate the prediction accuracy of DCALF, mean absolute error (MAE) and root mean squared error (RMSE), which are frequently adopted in QoS prediction [15], [16], [26], [27], [28], [29], [31], are computed by

$$MAE = \left(\sum_{(w,j)\in\Gamma} \left|r_{w,j} - \hat{r}_{w,j}\right|_{abs}\right) \Big/ |\Gamma|,$$
$$RMSE = \sqrt{\left(\sum_{(w,j)\in\Gamma} \left(r_{w,j} - \hat{r}_{w,j}\right)^2\right) \Big/ |\Gamma|}.$$

where Γ indicates the testing set and $|\cdot|_{abs}$ denotes the absolute value of a given number. The lower MAE and RMSE denote higher prediction accuracy. All experiments are run on a personal computer (PC) with a 3.7 GHz i7 central processing unit (CPU) and 64 GB random access memory (RAM).

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

4.3 Prediction Rule Selection

This section illustrates how a DCALF model selects its prediction rule according to the characteristics of D1.4 and D2.4. Firstly, we execute steps 1 and 2 on D1.4 and D2.4 to achieve their decision graphs for users and services as shown in Figs. 6, 7. Considering the decision graphs for users as shown in Figs. 6a and 7a, we observe that there are two cluster centers on D1.4 and three cluster centers on D2.4. This phenomenon indicates that the users of 1.4 and D2.4 could be separated into two and three neighborhoods, respectively. Hence, we can build a DCALF model with Prediction Rule 1 provided according to the inferences in Section 3.3.

On the other hand, considering the decision graphs for services as shown in Figs. 6b and 7b, we observe that there is only one cluster center on both D1.4 and D2.4. This phe-

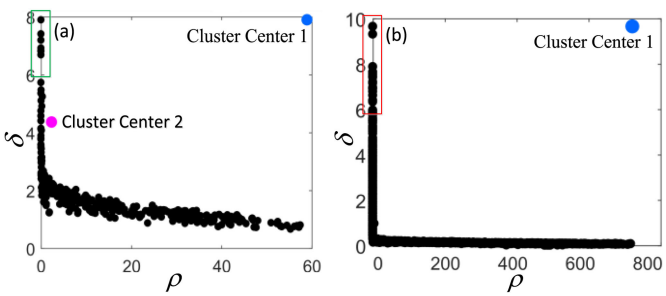


Fig. 6. Decision graph on D.1.4 in (a) users, (b) services.

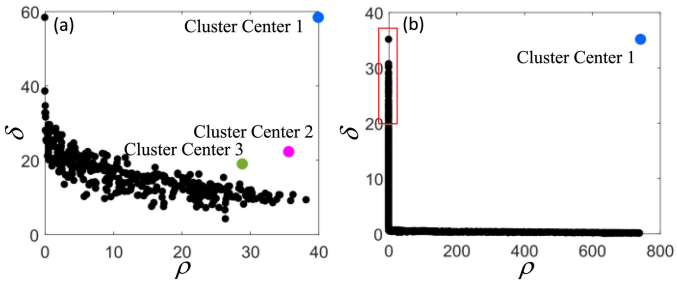


FIG. 7. Decision graph on D.2.4 in (a) users, (b) services.

nomenon indicates that the involved services do not have reliable neighborhoods. Moreover, we see that there are many outlier services (marked by the red rectangles) in both D1.4 and D2.4, which means that there are many unreliable services. Hence, based on the analyses in Section 3.3, we clearly see that a DCALF model with Prediction Rule 4 is also appropriate to implement QoS prediction on these two datasets.

Thus, in the following experiments, we focus on validating the performance of DCALF models based on Prediction Rules 1 and 4. Hereafter, we name a DCALF model with Prediction Rule 1 as the DCALF-A, and a DCALF model with Prediction Rule 4 as the DCALF-B, respectively.

4.4 Hyper-Parameters Sensitivity Tests

4.4.1 In DCALF-A (Prediction Rule 1)

As analyzed in Section 3, two model parameters, i.e., f and $\boldsymbol{\lambda}$, have major effects on the performance of a DCALF-A model. Hence, in this set of experiments, we perform sensitivity tests on it with respect to f and $\boldsymbol{\lambda}$.

Effects of f. In this set of experiments, for validating the effects of f, the other parameters are set as $\boldsymbol{\eta} = 0.01$ for D1, $\boldsymbol{\eta} = 0.0001$ for D2, $\boldsymbol{\lambda} = 0.01$, and $P_{Vec} = 2$ percent%, uniformly. The experimental results that f increases from 10 to 320 are recorded in Figs. S3 and S4 in the Supplementary File of this paper available online. According to prior research [16], [17], [37], an LF model's representation learning ability on a sparse matrix commonly increases as the LF dimension increases. In a DCALF-A model we observe similar phenomena, i.e., its RMSE and MAE tend to decrease as f increases. However, as f increases over 80, its MAE/RMSE decreasing trend becomes slow, or its MAE/RMSE even increases on some testing cases as shown in Figs. S3 and S4 available online. Note that f decides the rank of the low-rank approximation generated by a DCALF model to the target sparse matrix. From these results, we see that for a DCALF-A model, as f increases over the actual rank of the target HiDS matrix, it can suffer from accuracy loss. Meanwhile, as f increases, the time cost of DCALF increases linearly as analyzed in Section 3.4. Hence, relatively small f around 80 should be considered for balancing the time cost and prediction accuracy in practice.

Effects of $\boldsymbol{\lambda}$. In this set of experiments, for validating the effects of $\boldsymbol{\lambda}$, we set the other parameters as $\boldsymbol{\eta} = 0.01$ for D1, $\boldsymbol{\eta} = 0.0001$ for D2, $f = 20$, and $P_{Vec} = 2\%$, uniformly. The MAE and the RMSE of DCALF as $\boldsymbol{\lambda}$ increases are recorded in Figs. S5 and S6 in the Supplementary File of this paper available online respectively. Note that we have tested a larger range of $\boldsymbol{\lambda}$ on D2 than that on D1 because D2's numerical scale is much larger than that of D1, which leads to a magnitude difference in parameter selection. From Figs. S5 and S6 available online, we clearly see that it is not necessary to apply a heavy regularization effect to a DCALF-A model. As illustrated in Section 3, a DCALF model adopts a DPClust algorithm in Step 2 to detect the clusters and outliers in the target sparse data, which requires that LFs extracted in Step 1 precisely represent the target data. On the other hand, as $\boldsymbol{\lambda}$ increases the generality of the LF model achieved in Step 1 increases, which actually means that it somehow deviates from the given data. Once these two effects are not well balanced, accuracy loss will be resulted as shown in Figs. S5 and S6 available online.

However, it should be further mentioned that as $\boldsymbol{\lambda} = 0$, a DCALF-A model cannot converge on both D1 and D2 (which cannot be depicted in Figs. S5 and S6 available online since the MAE and RMSE becomes infinity). Thus, regularization is absolutely necessary in DCALF-A. Based on these results, we conclude that the light regularization effect should be applied to a DCALF-A model to ensure its prediction accuracy for missing QoS data.

4.4.2 In DCALF-B (Prediction Rule 4)

Considering a DCALF-B model, its performance is mainly affected by three parameters, i.e., $\boldsymbol{\alpha}$, f, and $\boldsymbol{\lambda}$. Thus, in the following, we aim to validate their effects.

Impacts of $\boldsymbol{\alpha}$. In this set of experiments, for validating the effects of $\boldsymbol{\alpha}$, the other parameters in DCALF-B are set as $\boldsymbol{\lambda} = 0.01$, $\boldsymbol{\eta} = 0.01$ for D1, $\boldsymbol{\eta} = 0.0001$ for D2, $f = 20$, and $P_{Vec} = 2\%$, uniformly. The MAE and the RMSE of DCALF as $\boldsymbol{\alpha}$ increases are presented in Figs. S7 and S8 in the Supplementary File of this paper available online respectively. On D1, the MAE and RMSE decreases at first and then increases in general as $\boldsymbol{\alpha}$ increases. The lowest MAE and RMSE are obtained when $\boldsymbol{\alpha}$ is around 0.3 in MAE and 0.15 in RMSE, as shown in Figs. S7 (a) and S8(a) available online. On D2, DCALF has the lowest MAE and RMSE when $\boldsymbol{\alpha}$ is around 0.1, as shown in Figs. S7 (b) and S8(b) available online. According to these results, it appears that more services on D1 are detected as unreliable ones by DCALF than on D2. Overall, these results indicate that the prediction accuracy of DCALF can be improved by adopting reliable services to train.

Impacts of f. The results are shown in Figs. S9 and S10 in the Supplementary File of this paper available online. Since these results are very similar to that in *Section 4.4.1.1)*, they are not described in detail for the sake of soundness. Please refer to *Section 4.4.1.1)* to see more details and analyses.

Impacts of $\boldsymbol{\lambda}$. The results are shown in Figs. S11 and S12 in the Supplementary File of this paper available online. Similarly, these results are not described in detail for the sake of soundness because they are very similar to that in *Section 4.4.1.2)* Please refer to *Section 4.4.1.2)* to see more details and analyses.

4.5 Performance Comparison

In this section, we compare a DCALF model with eight state-of-the-art QoS predictors in terms of prediction accuracy and computational efficiency. Details of compared QoS predictors are summarized in Table 2. Note that as analyzed in Sections 4.3 and 4.4, there are two versions for DCALF, which are also introduced in Table 2.

TABLE 2
Descriptions of All the Comparison Models.

Model	Description
BLF	The base LF model-based model proposed in 2009 [37], which consists of two situations, i.e., with and without linear biases. Note that we pick the best one as the QoS predictor when comparing on each test case.
RSNMF	The regularized single element dependent non-negative matrix factorization (MF) model proposed in 2016 [5], which is designed for the QoS predictor by incorporating regularization into a non-negative MF.
LN-LFM	The personalized LF-based QoS predictor proposed in 2014 [40], which improved the base LF model by incorporating the latent neighbor information.
NIMF	The neighborhood-integrated QoS predictor proposed in 2013 [41], which extends MF by employing the information of similar users.
NAMF	The network-aware MF-based QoS predictor proposed in 2016 [16], which is a geography-MF-based model.
GeoMF	The improved MF-based QoS predictor proposed in 2017 [15] based on geographical neighborhoods of QoS data, which is a geography-MF-based model.
LMF-PP	The location-MF-based QoS predictor proposed in 2018 [26], which needs additional geographical information.
AutoRec	The DNNs-based QoS predictor proposed in 2015 [42], which is an autoencoder [43] framework for CF and consists of I-AutoRec and U-AutoRec. Note that U-AutoRec is chosen as a rival model owing to its better performance in QoS prediction.
DCALF-A	A DCALF model with Prediction Rule 1 given in Section 3.3.
DCALF-B	A DCALF model with Prediction Rule 4 given in Section 3.3.

4.5.1 Comparison of Prediction Accuracy

To draw a fair comparison, we adopt the following settings: a) the LF dimension is set as $f = 20$ for all QoS predictors except for AutoRec because it is a DNNs-based model, b) the other parameters involved in the compared state-of-the-art QoS predictors are set according to their original papers, and c) for DCALF, its other parameters are set as $\boldsymbol{\alpha} = 0.2$ and $\boldsymbol{\eta} = 0.01$ on D1, $\boldsymbol{\alpha} = 0.1$ and $\boldsymbol{\eta} = 0.0001$ on D2, $\boldsymbol{\lambda} = 0.01$ and $P_{Vec} = 2\%$ on both D1 and D2.

The comparison results are presented in Tables 3 and S5 in the Supplementary File of this paper available online. To better understand them, the win/loss counts of DCALF-A/DCALF-B versus other QoS predictors are summarized in the third/second-to-last row of Tables 3 and S5 available online. From Tables 3 and S5 available online, we find that a) DCALF-A evidently achieves a lower MAE/RMSE than the other QoS predictors on D2 while it does not achieve that on D1, and b) DCALF-B achieves a lower MAE/RMSE than all the compared QoS predictors on most cases. The reasons for the above phenomenon can be explained from two aspects as shown in Figs. 6, 7: a) the neighborhoods of users are not very clear on D1 but clear on D2, and b) the unreliable services are very clear on both D1 and D2. Hence, predicting missing data of D1 based on neighborhoods of users (with DCALF-A) can be inappropriate, while predicting missing data of D2 based on reliable services (DCALF-B) is highly feasible.

Besides, to analyze the significance of the comparison results, we perform the Friedman test [44], [45], [46] in MAE and RMSE recorded in Tables 3 and S5 available online respectively. Note that a Friedman test is very effective in validating the performance of multiple models on multiple datasets. The Friedman statistical results are recorded in the last row of Tables 3 and S5 available online, where the hypothesis that these comparison QoS predictors have significant differences with a significance level of 0.05 is accepted. From the Friedman statistical results, we observe that DCALF-B has the lowest Friedman Rank values among all the QoS predictors, which means that it achieves the highest prediction accuracy among its peers.

For checking whether or not DCALF-B has significantly higher prediction accuracy than its peers, we further perform the Wilcoxon signed-ranks test between DCALF-B and each compared QoS predictor one by one in both MAE and RMSE, which are recorded in Tables 3 and S5 available online. Note that the Wilcoxon signed-ranks test is a non-parametric pairwise comparison method [47], [48], [49]. It has three indicators, i.e., $R+$, $R-$, and p-value. Large $R+$ denotes high prediction accuracy, large $R-$ denotes low prediction accuracy, and the p-value denotes the significance level. The statistical results are recorded in Tables 4 and S6 in the Supplementary File of this paper available online, where the accepted hypotheses under the significance level $= 0.1$ are highlighted. From Tables 4 and S6 available online, we have the following findings:

(a) With MAE as the evaluation metric, all the hypotheses are accepted, indicating that DCALF-B achieves significantly higher prediction accuracy than its peers do with MAE as the evaluation metric, as recorded in Table 4. For instance, from Table 3 we clearly see that DCALF-B achieves around 4.76%-12.61 percent% lower MAE than AutoRec does on all the testing cases.

(b) With RMSE as the evaluation metric, three hypotheses are not accepted, i.e., DCALF-B vs. LN-LFM, DCALF-B vs. NIMF, and DCALF-B vs. GeoMF. However, from Table S6 available online we clearly see that DCALF-B also achieves much higher $R+$ rankings than LN-LFM, NIMF, and GeoMF do, indicating that DCALF-B has better prediction accuracy than them in RMSE. This observation is also supported by the numerical results summarized in Table S5 available online.

4.5.2 Comparison of Computational Efficiency

Table S7 in the Supplementary File of this paper available online summarizes the computational costs of compared QoS predictors. Note that AutoRec is not included in Table S7 available online because its computational cost is extremely high due to the DNNs-based learning strategy [50]. Meanwhile, the computational cost of DCALF-A and DCALF-B models is the same according to Section 3.4.

TABLE 3
Comparison Results in MAE, Including Win/Loss Counts Statistic and Friedman Test, Where • and ✪ Respectively Indicate that DCALF-A and DCALF-B Have A Lower MAE Than Comparison Models.

Test Cases	BLF	RSNMF	LN-LFM	NIMF	NAMF	GeoMF	LMF-PP	AutoRec	DCALF	
									DCALF-A	DCALF-B
D1.1	0.5561•✪	0.5438✪	0.5501•✪	0.5502•✪	0.5465•✪	0.5305✪	0.5285✪	0.5467•✪	0.5457	0.5127
D1.2	0.4944•✪	0.4868•✪	0.4889•✪	0.4842✪	0.4976•✪	0.4827✪	0.4725✪	0.5055•✪	0.4857	0.4544
D1.3	0.4691•✪	0.4492✪	0.4611✪	0.4508✪	0.4625✪	0.4495✪	0.4472✪	0.4598✪	0.4642	0.4346
D1.4	0.4531•✪	0.4371✪	0.4424✪	0.4346✪	0.4360✪	0.4366✪	0.4260✪	0.4482✪	0.4520	0.4246
D2.1	18.9776•✪	21.4302•✪	18.6512•✪	17.7153•	20.2104•✪	24.7465•✪	18.3091•	21.3118•✪	17.6576	18.6237
D2.2	16.1924•✪	17.2305•✪	16.0634•✪	15.5264•✪	17.0126•✪	22.4728•✪	15.9125•✪	17.0310•✪	15.3595	15.3430
D2.3	14.9279•✪	14.6880•✪	14.7664•✪	14.2146✪	15.6547•✪	17.7908•✪	14.7450•✪	15.0156•✪	14.3836	14.0664
D2.4	14.3061•✪	14.3654•✪	14.1264•✪	13.5638✪	14.6482•✪	16.2852•✪	14.1033•✪	14.2265•✪	13.6697	13.5491
•Win/Loss	8/0	5/3	6/2	3/5	6/2	4/4	4/4	6/2	—	—
✪Win/Loss	8/0	8/0	8/0	7/1	8/0	8/0	7/1	8/0	—	—
Friedman Rank*	8	6.125	6.25	3.75	7.375	6.875	3	7.625	4.625	**1.375**

A lower Friedman Rank value indicates a higher prediction accuracy.

In Table S7 available online, N_{mtr} is the max iteration count, K_1 is the number of nearest neighbors for a user, and K_2 is the number of nearest neighbors for a service. Based on Table S7 available online, we conclude that: a) BLF, RSNMF, and LN-LFM have the lowest computational complexity because they are the basic LF-based QoS predictors without considering the neighborhoods or noises of QoS data, and b) DCALF's computational complexity is lower than that of NIMF, NAMF, GeoMF, and LMF-PP because we commonly have $f \ll \min\{|U|, |S|\}$ in a real system.

Moreover, we have tested the CPU running time of compared QoS predictors. Fig. 8 records the results on D1 and Fig. S13 in the Supplementary File of this paper available online records the results on D2, where we find that a) BLF, RSNMF, and LN-LFM have relatively less CPU running time than their peers, b) DCALF's CPU running time is less than or comparable to that of NIMF, NAMF, GeoMF, and LMF-PP, and c) AutoRec takes the most CPU running time among all the QoS predictors. Note that these findings are consistent with what has been concluded from Table S7 available online.

4.5.3 Summary of Performance Comparison

Based on the experimental results, we summarize that:

(a) DCALF model achieves higher prediction accuracy for missing QoS data than compared state-of-the-art QoS predictors. In particular, a DCALF-B model achieves significantly higher prediction accuracy than its peers including a DCALF-A model. Such results indicate that the data characteristic-aware prediction rule selection strategy mentioned in Section 3 plays a vital role in the prediction accuracy of DCALF models. And

(b) A DCALF model's computational efficiency is competitive with that of its peers.

TABLE 4
Wilcoxon Signed-Ranks Test in MAE
With Significance Level = 0.1.

Comparison	$R+$	$R-$	p-value
DCALF-B vs BLF	36	0	**0.0039**
DCALF-B vs RSNMF	36	0	**0.0039**
DCALF-B vs LN-LFM	36	0	**0.0039**
DCALF-B vs. NIMF	28	8	**0.0977**
DCALF-B vs. NAMF	36	0	**0.0039**
DCALF-B vs. GeoMF	36	0	**0.0039**
DCALF-B vs. LMF-PP	31	5	**0.0391**
DCALF-B vs. AutoRec	36	0	**0.0039**

4.6 Discussions

4.6.1 High Prediction Accuracy of a DCALF Model

A DCALF model consists of three steps, where step 1 and step 2 extract the information of the neighborhoods and the outliers of users/services. The extracted information is utilized in step 3 to achieve highly accurate QoS prediction. Without step 1 and step 2, a DCALF model degrades to a base LF model (BLF). Therefore, if we compare DCALF with BLF, we see how the extracted information boosts a DCALF model's prediction accuracy. From Tables 3 and S5 available online, we see that a DCALF-A model achieves much lower MAE/RMSE than a BLF model does, indicating that DCALF utilizes the information of identified neighborhoods of users to improve an LF model's QoS prediction accuracy, where the most significant accuracy gain is around 6.96 percent%. Besides, when comparing DCALF-B with BLF, we can conclude that DCALF improves an LF

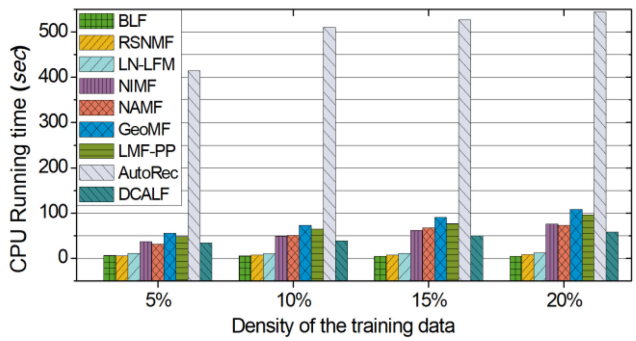


Fig. 8. CPU running time of comparison models on D1.

model's QoS prediction accuracy by utilizing the information of unreliable services, where the most significant accuracy gain is around 8.09 percent%. Therefore, steps 1 and 2 are essential for a DCALF model to achieve highly accurate QoS predictions.

4.6.2 QoS Issues in Web Service

A QoS model aims to evaluate the quality of Web services with similar functionality [51], [52]. So far, various QoS models are proposed [51], [52]. They concern various QoS properties including accessibility, capacity, response time, throughput, availability, interoperability, robustness, authentication, confidentiality, cost, and reputation [53]. Among them, some are user-dependent like response time, throughput, and availability [54]. Since different users commonly have different Internet connections and heterogeneous environments, the user-dependent QoS data can vary significantly from user to user. Hence, a QoS model developed on the user-dependent QoS data is likely to achieve more accurate selections from a large candidate Web services for a specific user. Warming-up test and QoS prediction are two main approaches to acquiring the user-dependent QoS data. However, Warming-up tests are usually difficult and even impractical due to the following reasons: a) testing all services is time-consuming and expensive, and b) it is impossible for a user to invoke all services from a large candidate set. Hence, QoS prediction is a vital issue in Web services.

In this paper, we aim at implementing QoS prediction based on historical QoS data by the proposed DCALF model. The adopted two datasets consist of real-world Web service QoS data, where the true QoS invocation records reflect the real Internet connections and heterogeneous environments. Hence, the QoS prediction results obtained on the adopted datasets are representative. In the experiments, although DCALF is only evaluated on the user-dependent QoS properties of response time and throughput, it is also compatible with other user-dependent QoS properties.

5 Related Work

Many QoS properties are user-dependent [53], like response time and throughput. Considering a specific service-based application scene like online product-oriented Web services for videos, user-dependent QoS data are mainly determined by users' invoking environment [54]. Hence, it becomes common [54] that if two users' historical QoS data are similar, their invoking environments are probably similar, making them probable to experience similar QoS in the future. From this point of view, a CF approach that essentially works by modeling the similarity among users and services [55], [56], [57], [58], [59] becomes feasible for QoS prediction [53], [54]. So far, a CF-based QoS predictor has achieved great progress in providing an efficient solution for many service-based applications, like a cloud computing-based application [60] and a multimedia service-based application [61].

Considering existing CF-based QoS predictors, an LF-based one is highly popular and widely investigated. For instance, Luo *et al*. [5] propose a QoS predictor by incorporating regularization and non-negative constraints into an LF model. Yu *et al*. [40] combine an LF model and a latent neighbor model to achieve highly accurate QoS predictions. Zheng *et al*. [41] propose an extended LF model that incorporates the information of user similarity into the LF extraction process. Zhu *et al*. [28] propose an adaptive LF model that is able to perform online QoS prediction via data transformation and online learning.

Generally speaking, when users are located in the same area, they probably possess similar invoking environments [40]. Hence, a user's geographical location is considered a key factor influencing user-dependent QoS data [16], [54], which is widely adopted as additional input for an LF model [15], [16], [26], [31], [63]. Kumar and Anouncia [62] propose to improve the accuracy of QoS-based Web service selection with the consideration of it. Chen *et al*. [15] propose an LF-based QoS predictor with the incorporation of geographical neighborhoods. Zheng *et al*. [16] propose a network-aware LF-based QoS predictor that considers the user network map. Yu *et al*. [63] propose a hybrid QoS predictor that combines LF analysis and network location-aware neighbor selection. Ryu *et al*. [26] propose to adopt the location information of both users and providers in an LF-based QoS predictor. Feng and Huang [31] propose a neighborhood-aware LF-based QoS predictor by systematically modeling geographical information, sample set diversity, and platform context.

However, a DCALF model proposed in this study possesses its own significance when compared with the abovementioned models in the following aspects:

(a) It builds the neighborhoods among users and services based on their LF matrices that precisely represent the known data of a target sparse QoS matrix, rather than on this target matrix itself. Thus, all observed information in the target sparse matrix is fully utilized to build highly robust neighborhoods. In comparison, existing neighborhood-aware LF-based QoS predictors mostly build user and item neighborhoods based on the raw sparse QoS matrix directly, which only utilizes a very small part of its observed data as illustrated in Fig. 1. Such neighborhoods are far less robust than those built by a DCALF model.

(b) It incorporates a DPClust algorithm into the modeling process to carefully detect the neighbors and noises among users/services, thereby enhancing its prediction accuracy and robustness. In comparison, existing LF-based QoS predictors do not have such designs.

(c) With the prediction rules presented in Section 3.3, it achieves significant accuracy gain over state-of-the-art QoS predictors [5], [15], [16], [26], [37], [40], [41], [42] as reported in Section 4.5. Note that such accuracy gain is achieved purely based on QoS data only, while a DCALF is also compatible with additional information like the widely-adopted geography information [15], [16], [26]. From this point of view, a DCALF model provides researchers and engineers with a novel and efficient approach to utilizing neighborhood information when addressing the problem of QoS prediction.

On the other hand, the above mentioned QoS predictors including the proposed DCALF model are all defined on static QoS data. In a dynamic environment, however, QoS

data experienced by the same user on the same service further becomes time-dependent, i.e., they vary over time [54]. According to a recent survey regarding user-dependent QoS predictors by Syu *et al*. [69], how to correctly model temporal dynamics when addressing dynamic QoS data becomes an emerging issue. Zhang *et al*. propose a non-negative tensor factorization-based QoS predictor for such purposes [64]. Other representative models of this kind include a spatial-temporal-based one [65], a dynamic CF-based one [66], a sparsity-tolerant-based one [67], and a privacy-preserving-based one [68]. Thus, we are encountering a highly interesting problem: how to make a DCALF model precisely grasp temporal dynamics hidden in dynamic QoS data? We plan to answer it in the future.

6 Conclusions

This paper proposes a data-characteristic-aware latent-factor (DCALF) model for generating highly accurate QoS predictions. Based on the presented methodology, experimental results and analyses, we have the conclusions that a) a DCALF model can precisely detect the neighborhoods and noises among users/services based on observed QoS data only, b) a DCALF model is data-characteristic-aware owing to its flexibility in selecting appropriate prediction rules according to the characteristics of observed QoS data, and c) a DCALF model achieves significant accuracy gains when comparing with state-of-the-art QoS predictors.

The proposed DCALF model adopts the standard SGD algorithm to acquire LFs from the sparse user-service QoS matrix, where its computational efficiency can be further improved via a parallel SGD algorithm [38], [39]. Meanwhile, a temporal dynamics-incorporated extension of DCALF is further desired. We will address these issues in the future.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under grant 2016YFB 1000901, in part by the National Natural Science Foundation of China under grants 61702475, 61772493, 91746209 and 61902370, in part by the Natural Science Foundation of Chongqing (China) under grants cstc2019jcyjjqX0013 and cstc2019jcyjmsxmX0578, in part by the Guangdong Province Universities and College Pearl River Scholar Funded Scheme (2019), in part by the CAS "Light of West China" Program, and in part by the Pioneer Hundred Talents Program of Chinese Academy of Sciences. Di Wu and Xin Luo are co-first authors of this paper.

References

[1] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Wsrec: A collaborative filtering based web service recommender system," in *Proc. IEEE Int. Conf. Web Serv.*, 2009, pp. 437–444.

[2] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "A survey on trust and reputation models for web services: Single, composite, and communities," *Decis. Support Syst.*, vol. 74, pp. 121–134, 2015.

[3] Z. Zheng, Y. Zhang, and M. R. Lyu, "Distributed QoS evaluation for real-world Web services," in *Proc. 2010 IEEE Int. Conf. Web Services Serv.*, 2010.

[4] C. Mao, J. Chen, D. Towey, J. Chen, and X. Xie, "Search-based QoS ranking prediction for web services in cloud environments," *Future Gener. Comput. Syst.*, vol. 50, pp. 111–126, 2015.

[5] X. Luo, M. Zhou, Y. Xia, Q. Zhu, A. C. Ammari, and A. Alabdulwahab, "Generating highly accurate predictions for missing QoS data via aggregating nonnegative latent factor models," *IEEE Trans. Neural Networks Netw. Learn. Syst.*, vol. 27, no. 3, pp. 524–537, Mar. 2016.

[6] M. Alrifai, and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition," in *Proc. 18th Int. Conf. World Wide Web*, 2009.

[7] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data mining with big data," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 97–107, Jan. 2014.

[8] D. A. Menascé, "QoS issues in Web services," *IEEE Internet Comput.*, vol. 6, no. 6, pp. 72–75, Nov./Dec. 2002.

[9] C. Jatoth, G. Gangadharan, and R. Buyya, "Computational intelligence based QoS-aware web service composition: A systematic literature review," *IEEE Trans. Services Comput.*, vol. 10, no. 3, pp. 475–492, May/Jun. 2017.

[10] D. Geebelen *et al.*, "QoS prediction for web service compositions using kernel-based quantile estimation with online adaptation of the constant offset," *Inf. Sci.*, vol. 268, pp. 397–424, 2014.

[11] Y. Ma, S. Wang, P. C. Hung, C.-H. Hsu, Q. Sun, and F. Yang, "A highly accurate prediction algorithm for unknown web service QoS values," *IEEE Trans. Services Comput.*, vol. 9, no. 4, pp. 511–523, Jul./Aug. 2016.

[12] X. Chen, X. Liu, Z. Huang, and H. Sun, "Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation," in *Proc. 2010 IEEE Int. Conf. Web Services*, 2010.

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

[14] K. Lee, J. Park, and J. Baik, "Location-based web service QoS prediction via preference propagation for improving cold start problem," in *Proc. 2015 IEEE Int. Conf. Web Services Serv.*, 2015.

[15] Z. Chen, L. Shen, F. Li, and D. You, "Your neighbors alleviate cold-start: On geographical neighborhood influence to collaborative web service QoS prediction," *Knowl.-Based Syst.*, vol. 138, pp. 188–201, 2017.

[16] M. Tang, Z. Zheng, G. Kang, J. Liu, Y. Yang, and T. Zhang, "Collaborative web service quality prediction via exploiting matrix factorization and network map," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 1, pp. 126–137, Mar. 2016.

[17] X. Luo, M. Zhou, S. Li, Z. You, Y. Xia, and Q. Zhu, "A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method," *IEEE Trans. Neural Networks Netw. Learn. Syst.*, vol. 27, no. 3, pp. 579–592, Mar. 2016.

[18] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *Acm Comput. SurveysSurv.*, vol. 47, no. 1, pp. 1–45, 2014.

[19] H. Chen, and J. Li, "Learning multiple similarities of users and items in recommender systems," in *Proc. 2017 IEEE Int. Conf. Data Mining*, 2017.

[20] H. Zhao, Q. Yao, J. T. Kwok, and D. L. Lee, "Collaborative filtering with social local models," in *Proc. 2017 IEEE Int. Conf. Data Mining*, 2017.

[21] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," in *Proc. 2006 SIAM Int. Conf. Data Mining*, pp. 549–553, 2006.

[22] J. M. Hernández-Lobato, N. Houlsby, and Z. Ghahramani, "Probabilistic matrix factorization with non-random missing data," in *Proc. 31st Int. Conf. Mach. Learn., JMLR*, 2014.

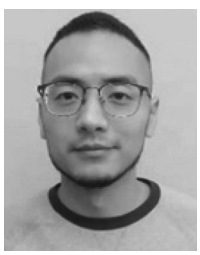
[23] Y. Cai, H. Leung, Q. Li, J. Tang, and J. Li, "TyCo: owards Typicality-based collaborative filtering recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 3, pp. 766–779, Mar. 2014.

[24] Y. Koren, and R. Bell, "Advances in collaborative filtering," in *Recommender Systems Handbook*, . Berlin, Germany: Springerpp. 77–118: Springer, 2015, pp. 77–118.

[25] Y. Yang, Z. Zheng, X. Niu, M. Tang, Y. Lu, and X. Liao, "A location-based factorization machine model for web service QoS prediction," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2018.2876532.

[26] D. Ryu, K. Lee, and J. Baik, "Location-based web service QoS prediction via preference propagation to address cold start problem," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2018.2821686.

[27] H. Wu, K. Yue, B. Li, B. Zhang, and C.-H. Hsu, "Collaborative QoS prediction with context-sensitive matrix factorization," *Future Gener. Comput. Syst.*, vol. 82, pp. 669–678, 2018.
[28] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, "Online qos prediction for runtime service adaptation via adaptive matrix factorization," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 10, pp. 2911–2924, Oct. 2017.
[29] C. Wu, W. Qiu, Z. Zheng, X. Wang, and X. Yang, "QoS prediction of web services based on two-phase k-means clustering," in *Proc. 2015 IEEE Int. Conf. Web Services Serv.*, 2015.
[30] A. Liu *et al.*, "Differential private collaborative web services QoS prediction," *World Wide Web*, vol. 22, 2018, doi: 10.1007/s11280-018-0544-7.
[31] Y. Feng, and B. Huang, "Cloud manufacturing service QoS prediction based on neighbourhood enhanced matrix factorization," *J. Intell. Manuf.*, vol. 31, pp. 1649–1660, 2018, doi: 10.1007/s10845-018-1409-8.
[32] J. Wu, L. Chen, Y. Feng, Z. Zheng, M. C. Zhou, and Z. Wu, "Predicting quality of service for selection by neighborhood-based collaborative filtering," *IEEE Trans. Syst. , Man , Cybern. , Syst.*, vol. 43, no. 2, pp. 428–439, Mar. 2013.
[33] X. Luo *et al.*, "Incorporation of efficient second-order solvers into latent factor models for accurate prediction of missing QoS data," *IEEE Trans. Cybern.*, vol. 48, no. 4, pp. 1216–1228, Apr. 2018.
[34] X. Luo, M. Zhou, Z. Wang, Y. Xia, and Q. Zhu, "An effective QoS estimating scheme via alternating direction method-based matrix factorization," *IEEE Trans. Services Comput.*, vol. 12, no. 4, pp. 503–518, Jul.–Aug. 2019.
[35] W. Qiu, Z. Zheng, X. Wang, X. Yang, and M. R. Lyu, "Reputation-aware qos value prediction of web services," in *Proc. IEEE 2013 Int. Conf. Services Comput.* IEEE, 2013, pp. 41–48.
[36] A. Rodriguez, and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.
[37] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.
[38] H. Li, K. Li, J. An, and K. Li, "MSGD: A novel matrix factorization approach for large-scale collaborative filtering recommender systems on GPUs," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 7, pp. 1530–1544, Jul. 2018.
[39] X. Luo, H. Liu, G. Gou, Y. Xia, and Q. Zhu, "A parallel matrix factorization based recommender by alternating stochastic gradient decent," *Eng. Appl. Artif. Intell.*, vol. 25, no. 7, pp. 1403–1412, 2012.
[40] D. Yu, Y. Liu, Y. Xu, and Y. Yin, "Personalized QoS prediction for web services using latent factor models," in *Proc. 2014 IEEE Int. Conf. Services Serv. Comput.*, IEEE, 2014.
[41] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Collaborative web service QoS prediction via neighborhood integrated matrix factorization," *IEEE Trans. Services Comput.*, vol. 6, no. 3, pp. 289–299, Third quarter 2013.
[42] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders meet collaborative filtering," in *Proc. 24th ACM Int. Conf. World Wide Web*, ACM, 2015, pp. 111–112.
[43] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, 2015, Art. no. 436.
[44] A. Benavoli, G. Corani, and F. Mangili, "Should we really use post-hoc tests based on mean-ranks?," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 152–161, 2016.
[45] A. Dong, F.-L. Chung, and S. Wang, "Semi-supervised classification method through oversampling and common hidden space," *Inf. Sci.*, vol. 349, pp. 216–228, 2016.
[46] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, no. Jan, pp. 1–30, 2006.
[47] D. Wu, X. Luo, G. Wang, M. Shang, Y. Yuan, and H. Yan, "A highly accurate framework for self-labeled semisupervised classification in industrial applications," *IEEE Trans. Ind. Inform.*, vol. 14, no. 3, pp. 909–920, Mar. 2018.
[48] Z. Liu, E. Blasch, and V. John, "Statistical comparison of image fusion algorithms: Recommendations," *Inf. Fusion*, vol. 36, pp. 251–260, 2017.
[49] B. Rosner, R. J. Glynn, and M. L. T. Lee, "The wilcoxon signed rank test for paired comparisons of clustered data," *Biometrics*, vol. 62, no. 1, pp. 185–192, 2006.
[50] Z.-H. Zhou, and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *Proc. 26th Int. Joint Conf. Artif. Intell., IJCAI*, 2017.
[51] K. Sangsanit, W. Kurutach, and S. Phoomvuthisarn, "REST web service composition: A survey of automation and techniques." in *Proc. 2018 Int. Conf. Inf. Netw.*, 2018.
[52] O. Kondratyeva, N. Kushik, A. Cavalli, and N. Yevtushenko, "Evaluating quality of web services: A short survey." in *Proc. 2013 IEEE Int. Conf. Web Services*, 2013.
[53] S. H. Ghafouri, S. M. Hashemi, and P. C. K. Hung, "A survey on web service QoS prediction methods," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2020.2980793.
[54] Z. Zheng, L. Xiaoli, M. Tang, F. Xie, and M. R. Lyu, "Web service QoS prediction via collaborative filtering: A survey," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2020.2995571.
[55] H. Shin, S. Kim, J. Shin, and X. Xiao, "Privacy enhanced matrix factorization for recommendation with local differential privacy," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1770–1782, Sep. 2018.
[56] Y. He, C. Wang, and C. Jiang, "Correlated matrix factorization for recommendation with implicit feedback," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 3, pp. 451–464, Mar. 2019.
[57] X. Ren, M. Song, E. Haihong, and J. Song, "Context-aware probabilistic matrix factorization modeling for point-of-interest recommendation," *Neurocomputing*, vol. 241, pp. 38–55, 2017.
[58] H. Wu, Z. Zhang, K. Yue, B. Zhang, J. He, and L. Sun, "Dual-regularized matrix factorization with deep neural networks for recommender systems," *Knowl.-Based Syst.*, vol. 145, pp. 46–58, 2018.
[59] C. Wang *et al.*, "Confidence-aware matrix factorization for recommender systems," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.
[60] Z. u. Rehman, O. K. Hussain, F. K. Hussain, E. Chang, and T. S. Dillon, "User-side QoS forecasting and management of cloud services," *World Wide Web*, vol. 18, no. 6, pp. 1677–1716, 2015.
[61] M. S. Hossain, "QoS in web service-based collaborative multimedia environment," in *Proc. 16th Int. Conf. Adv. Communication Technol.*, 2014.
[62] S. S. Kumar, and S. M. Anouncia, "Qos-based concurrent user-service grouping for web service recommendation," *Autom. Control Comput. Sci.*, vol. 52, no. 3, pp. 220–230, 2018.
[63] Y. Yin, S. Aihua, G. Min, X. Yueshen, and W. Shuoping, "QoS prediction for web service recommendation with network location-aware neighbor selection," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 26, no. 04, pp. 611–632, 2016.
[64] W. Zhang, H. Sun, X. Liu, and X. Guo, "Temporal qos-aware web service recommendation via non-negative tensor factorization," in *Proc. 23rd ACM Int. Conf. World Wide Web*, ACM, pp. 585–596, 2014.
[65] X. Wang, J. Zhu, Z. Zheng, W. Song, Y. Shen, and M. R. Lyu, "A spatial-temporal QoS prediction approach for time-aware web service recommendation," *ACM Trans. Web*, vol. 10, no. 1, pp. 1–25, 2016.
[66] C. Yu, and L. Huang, "Time-aware collaborative filtering for qos-based service recommendation," in *Proc. 2014 IEEE Int. Conf. Web Services Serv.*, IEEE, 2014.
[67] W. Chen, W. Qiu, X. Wang, Z. Zheng, and X. Yang, "Time-aware and sparsity-tolerant QoS prediction based on collaborative filtering," in *Proc. 2016 IEEE Int. Conf. Web Services Serv.*, 2016.
[68] L. Qi, R. Wang, C. Hu, S. Li, Q. He, and X. Xu, "Time-aware distributed service recommendation with privacy-preservation," *Inf. Sci.*, vol. 480, pp. 354–364, 2019.
[69] S. Yang, C. M. Wang, and Y. Y. Fanjiang, "A survey of time-aware dynamic QoS forecasting research: Its future challenges and research directions," in *Proc. 2018 Int. Conf. Services Comput.*, 2018.



Di Wu (Member, IEEE) received the Ph.D. degree in computer science from the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences (CAS), Chongqing, China, in 2019. He is currently an associate professor with the Chongqing Institute of Green and Intelligent Technology, CAS, Chongqing, China. He was a visiting scholar from April 2018 to April 2019 at the University of Louisiana at Lafayette, USA. His research interests include data mining and machine learning.

Xin Luo (Senior Member, IEEE) received the B.S. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2005, and the Ph.D. degree in computer science from Beihang University, Beijing, China, in 2011. In 2016, he joined the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China, as a professor of computer science and engineering. He is currently also a distinguished professor of computer science with the Dongguan University of Technology, Dongguan, China. His current research interests include big data analysis and intelligent control. He has published over more than 100 papers (including over more than 50 IEEE Transactions papers) in the above areas. He was a recipient of the Hong Kong Scholar Program jointly by the Society of Hong Kong Scholars and China postdoctoral Science Foundation, in 2014, the Pioneer Hundred Talents Program of the Chinese Academy of Sciences, in 2016, and the Advanced Support of the Pioneer Hundred Talents Program of Chinese Academy of Sciences, in 2018. He is currently serving as an associate editor for the *IEEE/CAA Journal of Automatica Sinica*, *IEEE Access* and *Neurocomputing*. He has received the Outstanding associate editor Award of *IEEE Access*, in 2018.

Mingsheng Shang received his the Ph.D. degree in computer science from the University of Electronic Science and Technology of China (UESTC). He joined the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China, in 2015, and is currently a professor of computer science and engineering. His research interests include data mining, complex networks, and cloud computing.

Yi He received his the B.E. degree in transportation engineering from the Harbin Institute of Technology, China, and his the M.S. degree in civil engineering from the University of Louisiana at Lafayette, USA, in 2013 and 2017, respectively. He is currently working toward the Ph.D. degree in computer science from the University of Louisiana at Lafayette, USA. His research interests include optimization, machine learning, and data mining.

Guoyin Wang (Senior Member, IEEE) received the B.E. degree in computer software, the and M.S. degree in computer software, and the Ph.D. degree in computer organization and architecture from Xi'an Jiaotong University, Xi'an, China, in 1992, 1994, and 1996, respectively. He worked at the University of North Texas, USA, and the University of Regina, Canada, as a visiting scholar during 1998–1999. Since 1996, he has been working at the Chongqing University of Posts and Telecommunications, where he is currently a professor and Ph.D. supervisor, director of the Chongqing Key Laboratory of Computational Intelligence, and the dean of the College of Computer Science and Technology. His research interests include data mining, machine learning, rough set, granular computing, cognitive computing, etc. He is the president of International Rough Set Society (IRSS), a vice-president of the Chinese Association for Artificial Intelligence (CAAI), a fellow of the China Computer Federation (CCF), and a senior member of IEEE.

Xindong Wu (Fellow, IEEE) received the Ph.D. degree in artificial intelligence from The the University of Edinburgh, Edinburgh, U.K., in 1993. He is currently the president of the Mininglamp Academy of Sciences, Mininglamp Technology, Beijing, China. He is also a professor with the the Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Hefei, China. His research interests include data mining and knowledge engineering. He is a fellow of the Association for American Association for the Advancement of Science (AAAS). He is also the editor-in-chief of *Knowledge and Information Systems* and *Advanced Information and Knowledge Processing* (Springer book series).