A Posterior-Neighborhood-Regularized Latent Factor Model for Highly Accurate Web Service QoS Prediction

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

Abstract—Neighborhood regularization is highly important for a latent factor (LF)-based Quality-of-Service (QoS)-predictor since similar users usually experience similar QoS when invoking similar services. Current neighborhood-regularized LF models rely prior information on neighborhood obtained from common raw QoS data or geographical information. The former suffers from low prediction accuracy due to the difficulty of constructing the neighborhood based on incomplete QoS data, while the latter requires additional geographical information that is usually difficult to collect considering information security, identity privacy, and commercial interests in real-world scenarios. To address the above issues, this work proposes a posterior-neighborhood-regularized LF (PLF) model for QoS prediction. The main idea is to decompose the LF analysis process into three phases: a) primal LF extraction, where the LFs are extracted to represent involved users/services based on known QoS data, b) posterior-neighborhood construction, where the neighborhood of each user/service is achieved based on similarities between their primal LF vectors, and c) posterior-neighborhood-regularized LF analysis, where the objective function is regularized by both the posterior-neighborhood of users/services and L_2-norm of desired LFs. Experimental results from large scale QoS datasets demonstrate that PLF outperforms state-of-the-art models in terms of both accuracy and efficiency.

Index Terms—Web service, quality-of-service, latent factor analysis, posterior-neighborhood, regularization, cloud computing, big data

1 Introduction

Web Services are the fundamental components for cloud computing-based software applications [1]. They are designed for easy exchange of data among software applications over the World Wide Web [2], [3], [4]. In the era of big data and cloud computing, many service providers deploy web services to serve their customers, which explosively increases the number of online web services [2], [3], [5]. Under such circumstances, it is a challenge for potential users to select appropriate web services, especially when many available candidate web services have highly similar or even equivalent functionality [6], [7], [8].

Quality-of-Service (QoS), which measures the nonfunctional characteristics of web services, e.g., response time and throughput, plays an important role in service discovery and selection [2], [3], [8], [9]. If the QoS data of candidate web services are available, web services that fulfill potential users' QoS requirements can be selected and recommended. Warming-up tests, which directly invoke web services, are frequently performed to retrieve QoS data [10], [11]. However, the number of candidate services is usually very large in real-world applications. Besides, most web service invocations are charged. Hence, it is very time-consuming, expensive and thus impractical to inspect all the candidate web services for retrieving QoS data [6], [12], [13].

Alternatively, QoS prediction is another widely used way to acquire QoS data without such deficiency [10], [12], [13], [14], [15], [16]. A QoS predictor aims at accurately predicting unknown QoS data based on historical web service invocations. Collaborative filtering (CF), as a very successful technique for implementing recommender systems in e-commerce [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], has been employed to implement the QoS predictors in recent years [10], [11], [12], [13], [14], [15], [29], [30], [31], [32], [33], [34]. A CF-based QoS predictor makes predictions based on a user-service QoS matrix [10], [11], [12], [13], [14], [15], [29], [30], [31], [32], [33], [35], [36], where each column denotes a web service, each row denotes a user,

- *D. 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 also with the University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: wudi@cigit.ac.cn.*
- *X. Luo 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 also with the Hengrui (Chongqing) Artificial Intelligence Research Center and Department of Big Data Analyses Techniques, Cloudwalk, Chongqing 401331, China. E-mail: luoxin21@cigit.ac.cn.*
- *M. Shang, and G. Wang are 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, wanggy@ieee.org.*
- *Q. He is with the School of Software and Electrical Engineering, Swinburne University of Technology, Melbourne 3122, Australia. E-mail: qhe@swin.edu.au.*
- *Y. He is with the University of Louisiana at Lafayette, Lafayette, Louisiana 70503 USA. E-mail: yi.he1@louisiana.edu.*

Manuscript received 16 Dec. 2018; revised 11 Nov. 2019; accepted 18 Dec. 2019. Date of publication 24 Dec. 2019; date of current version 8 Apr. 2022. (Corresponding author: Xin Luo.)
Digital Object Identifier no. 10.1109/TSC.2019.2961895

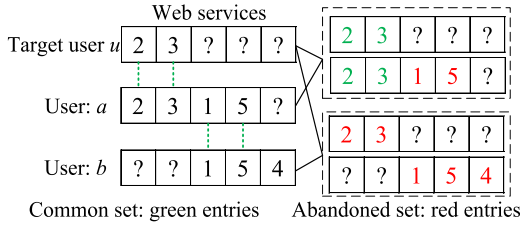


Fig. 1. Dilemma in building reliable neighborhood based on common sets defined on raw QoS data.

and each entry denotes the QoS record of a user's innovation on a service. This user-service matrix is very sparse because a user usually experiences only a very small subset of all the available services in real-world scenarios. Thus, the major problem of CF-based QoS prediction is how to make accurate predictions based on a very sparse user-service QoS matrix.

Among various CF-based QoS predictors [12], [13], [37], latent factor (LF)-based QoS predictors are the most popular due to their high scalability and prediction accuracy [11], [14], [15], [16], [29], [30], [31], [33], [38]. Neighborhood regularization plays a key role in an LF-based QoS predictor because similar users usually experience similar QoS when invoking similar services [11], [14], [15], [33]. Current neighborhood-regularized LF-based QoS predictors mostly employ the common set defined on the raw partial QoS matrix or additional geographical information to identify users'/services' neighbors. They rely on *prior neighborhood* that is identified before the LF analysis process and suffer from the following major limitations:

- It is difficult to accurately identify users'/services' neighbors based merely on the common set defined on the extremely sparse raw user-matrix matrix. The major reason is that the common sets of users/services are often too small to model user/service similarities precisely. For example, Fig. 1 shows that many known data (red entries) are not taken into account in the search of the common set among users. As a result, a lot of useful information is not properly utilized, which lowers the accuracy of the identified user's/service's neighbors. As shown in Fig. 1, although target user u, user a, and user b are similar, user b is also considered as a dissimilar user to target user u because they have no common QoS records.
- It is often difficult to retrieve the required geographical information due to the issues of identity privacy, information security, and commercial interest. Moreover, geographical similarities can be influenced by unexpected factors such as information facilities, routing policies, network throughput, and time of invocation.

To overcome the above limitations, this paper proposes a *posterior-neighborhood*-regularized latent factor (PLF) model for QoS prediction, which neighborhood accurately for regularization solely and fully based on the user-service matrix. The main idea of PLF is to decompose the LF analysis process into three phases: a) primal LF extraction, where the LFs are extracted based on an objective function without any regularization for precisely fitting the given data; b) posterior-neighborhood construction, where the involved users'/services' neighborhood is identified based on the similarity between their primal LF vectors; and c) posterior-neighborhood-regularized LF analysis, where the objective function is regularized by both the posterior-neighborhood and l_2-norm of desired LFs. The main contributions of this paper include:

- A posterior-neighborhood-regularized latent factor (PLF) model with excellent abilities in achieving highly accurate QoS prediction;
- A suite of theoretical analyses and algorithms design of PLF;
- Extensive experiments on a widely-used real-world QoS dataset regarding the QoS prediction comparisons between PLF and related state-of-the-art models.

To the best of our knowledge, PLF significantly advances QoS prediction in accuracy for web services. Besides, PLF differs from state-of-the-art models in two major ways. First, PLF is capable of constructing neighborhood accurately based on the full information in the user-service matrix, which makes PLF achieve highly accurate QoS predictions. Second, PLF relies solely on the user-service matrix and does not require any additional information, which makes PLF easy to use and generally applicable to various recommendation scenarios.

The rest of this paper is organized as follows. Section 2 states the preliminaries. Section 3 presents the theories and algorithms that facilitate the PLF model. Section 4 provides and discusses the experimental results. Section 5 analyzes the related work. Finally, Section 6 gives conclusions and future works.

2 Preliminaries

2.1 Notations

Table 1 summarizes the annotations used in this paper.

2.2 Latent Factor Analysis-Based QoS Prediction

An LF-based QoS predictor takes a user-service matrix R as the input [15], [17], [18]. Since a user u from U usually invokes a very small subset of the services in S, R is very sparse with $|R_K| << |R_U|$. According to prior researches [10], [11], [15], [17], [18], [38], [39], an LF-based QoS predictor is defined as:

Definition. *Given U, S, R, R_K and* f*, an LF-based QoS predictor extracts an LF matrix* P *to represent* U *and another LF matrix* Q *to represent* S *based on R_K to achieve* R*'s rank-*f *approximation $\hat{R}$.* P *and* Q *are extracted by minimizing an objective function defined on R_K while fulfilling the condition of $f << \min(|U|, |S|)$. $\hat{R}$ can be obtained by $\hat{R} = PQ$.*

To accurately represent R_K with P and Q, an appropriate objective function measuring the difference between R and $\hat{R}$ is highly important [10], [11], [15], [17], [18], [38]. A Euclidean distance-based objective is a common choice [15], [17], [18]:

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

Note that (1) is ill-posed [17], [38]. To address this issue, l_2-norm regularization is employed to enhance the generality of a resultant LF model:

TABLE 1
Symbol Annotations

Symbol	Explanation
U	Target user set.
S	Target web service set.
R	Target user-service matrix with $\|U\|$ rows and $\|S\|$ *columns*.
$r_{u,s}$	R's element at uth row and sth column denoting the QoS record experienced by user $u \in U$ experience on service $s \in S$.
R_K	Known entry set of R.
R_U	Unknown entry set of R.
f	The number of dimensions in the LF space.
$P^{\|U\|\times f}$	LF matrix for U.
$P\prime^{\|U\|\times f}$	LF matrix for U.
p_u	uth row-vector of P.
p'_u	uth row-vector of P'.
$\bar{p}_u$	Mean of LFs in p_u.
$Q^{f\times\|S\|}$	LF matrix for S.
$Q\prime^{f\times\|S\|}$	LF matrix for S.
q_s	sth column-vector of Q.
$q\prime_s$	sth column-vector of Q'.
$\bar{q}_s$	Mean of LFs in q_s.
$\hat{R}$	R's rank-f approximation built on R_K with $f << \min(\|U\|, \|S\|)$.
$\hat{r}_{u,s}$	$\hat{R}$'s element at uth row and sth column denoting prediction for $r_{u,s}$.
λ	Regualrization coefficient for l_2-norm-based regularization.
$\varepsilon(P, Q)$	Objective function with respect to P and Q.
$\varepsilon_{u,s}$	Instant loss on $r_{u,s}$.
N_{mtr}	Max-training-round count for an LF model.
η	Learning rate.
$PCC_u(u,k)$	Pearson correlation coefficient (PCC) [16] between users u and k.
$PCC_s(s,j)$	PCC between services s and j.
K_1	The number of the K_1-nearest-neighbors for a user.
$N_{K1}(u)$	K_1-nearest-neighbor set for user u.
$T_u(u)$	Regularization neighbor set for user u.
K_2	The number of the K_2-nearest-neighbors for a service.
$N_{K2}(s)$	K_2-nearest-neighbor set for service s.
$T_s(s)$	Regularization neighbor set for service s.
$Su_{u,k}$	Normalized similarity weight between users u and k.
$Ss_{s,j}$	Normalized similarity weight between services s and j.
α_1	Regularization coefficient for posterior-neighborhood-based regularization on users.
α_2	Regularization coefficient for posterior-neighborhood-based regularization on services.
Γ	Testing set.
G	A synthetic dataset
$g_{u,s}$	G's element at uth row and sth column.
$\|\cdot\|$	The cardinality of an enclosed set.
$\|\cdot\|_{abs}$	The absolute value of an enclosed number.

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

With an optimization algorithm, such as stochastic gradient descent (SGD), P and Q can be extracted from (2).

3 Posterior-Neighborhood-Regularized Latent Factor Model

As analyzed in prior studies [11], [14], [15], [33], the prediction accuracy of an LF-based QoS predictor can be further improved by regularizing an objective function like (2) with users'/services' neighborhood information. However, current models rely on the prior neighborhood that is built before the LF analysis to achieve the regularization. Such regularization may be not always useful to improve the prediction accuracy because the prior neighborhood directly identified on the raw partial QoS matrix are not reliable. To address this issue, we propose the PLF model, which adopts a posterior approach to identify users'/services' neighborhood.

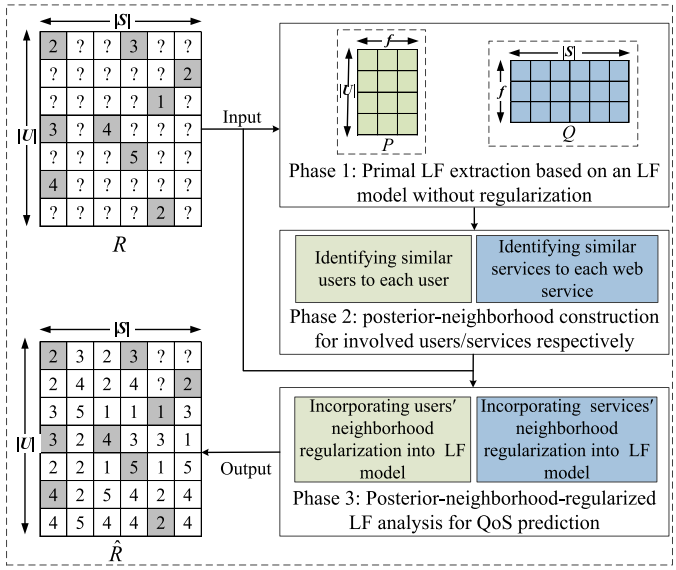


Fig. 2. Flowchart of the proposed PLF model.

Fig. 2 shows the flowchart of a PLF model that consists of three phases. Phase 1 is the primal LF extraction. By training an LF model without any regularization, P and Q are extracted from the full known entries in R for users and services respectively. Phase 2 is the posterior neighbor identification for target users/services. Specifically, each target user's similar users are identified from P and each target service's similar services are identified from Q. Phase 3 is the posterior-neighborhood-regularized LF analysis for predicting the missing QoS data in R, where the objective function is regularized by both the posterior-neighborhood and l_2-norm of the desired LFs. In this section, we discuss and analyze the three phases in detail.

3.1 Primal LF Extraction

In this phase, the aim is to precisely extract primal LF matrices P and Q for users and services respectively, which means that the LF model should be trained without regularization to accurately fit R_K. To do so, we apply SGD to (1). By considering the instant loss on a single instance $r_{u,s}$, we have

$$\varepsilon_{u,s} = \frac{1}{2}\left(r_{u,s}-\sum_{n=1}^{f}p_{u,n}q_{s,n}\right)^2. \tag{3}$$

With SGD, the LFs involved in (3) are trained by moving them along the opposite of the stochastic gradient of (3) with respect to each single LF, i.e., we make

$$On\ r_{u,s}, for n = 1 \sim f: \begin{cases} p_{u,n} \leftarrow p_{u,n} + \eta q_{s,n}\left(r_{u,s}-\sum_{n=1}^{f}p_{u,n}q_{s,n}\right), \\ q_{s,n} \leftarrow q_{s,n} + \eta p_{u,n}\left(r_{u,s}-\sum_{n=1}^{f}p_{u,n}q_{s,n}\right). \end{cases} \tag{4}$$

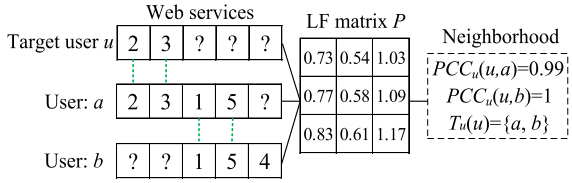


Fig. 3. An example of posterior-neighborhood construction.

After all the $r_{u,s}$ in R_K are used to train with (4), we can extract the primal LFs, which can be utilized to identify the posterior-neighbors of the target users/services in Phase 2.

3.2 Posterior-Neighborhood Construction

With the primal LF matrices P and Q extracted in Phase 1, we can obtain the feature vectors for each user/service. Those feature vectors are dense. In addition, by accurately fitting R_K, P and Q well represent the information hidden in R. Thus, we can precisely identify target users'/services' posterior-neighbors based on P and Q. For example, as shown in Fig. 3, through the primal LF extraction, we obtain the dense LF matrix P which includes the feature vectors for target user u, user a, and user b. Base on P, we can identify users a and b as target user u's similar users. This addresses the limitation shown in Fig. 1 where user b cannot be identified as target user u's similar user because they have no common set.

As indicated by prior researches [11], [14], [15], [40], PCC is a reliable similarity metric. Hence, we choose it in this research as the similarity metric for posterior-neighbor identification. For two users u and k, we compute their similarity $PCC_u(u,\ k)$ as follow

$$\underset{u\in U,k\in U,u\neq k}{PCC_u(u,k)} = \frac{\sum_{n=1}^{f}\left(p_{u,n}-\bar{p}_u\right)\left(p_{k,n}-\bar{p}_k\right)}{\sqrt{\sum_{n=1}^{f}\left(p_{u,n}-\bar{p}_u\right)^2}\sqrt{\sum_{n=1}^{f}\left(p_{k,n}-\bar{p}_k\right)^2}}, \tag{5}$$

where $\bar{p}_u$ and $\bar{p}_k$ are respectively given by

$$\bar{p}_u = \frac{1}{f}\sum_{n=1}^{f} p_{u,n},\ \bar{p}_k = \frac{1}{f}\sum_{n=1}^{f} p_{k,n}. \tag{6}$$

Note that $PCC_u(u,k)$ lies in the interval of [-1,1], where a large value denotes a high similarity between the corresponding pair of users. Based on user similarity given by (5), the K_1-nearest-neighbor set for each user can be identified. Then, we can obtain the regularization neighbor set for each user by

$$T_u(u) = \left\{k \middle| k \in N_{K_1}(u), PCC_u(u,k) > 0, k \neq u\right\}. \tag{7}$$

Note that the neighborhood relationship described by (7) is asymmetric.

Similarly, given two services s and j, we compute their similarity $PCC_s(s,\ j)$ with

$$PCC_s(s,j)_{s\in S,j\in S,s\neq j} = \frac{\sum_{n=1}^{f}\left(q_{s,n}-\bar{q}_s\right)\left(q_{j,n}-\bar{q}_j\right)}{\sqrt{\sum_{n=1}^{f}\left(q_{s,n}-\bar{q}_s\right)^2}\sqrt{\sum_{n=1}^{f}\left(q_{j,n}-\bar{q}_j\right)^2}},$$
$$\bar{q}_s = \frac{1}{f}\sum_{n=1}^{f} q_{s,n},\ \bar{q}_j = \frac{1}{f}\sum_{n=1}^{f} q_{j,n}. \tag{8}$$

With the service similarity given by (8), we can obtain the regularization neighbor set for each service by

$$T_s(s) = \left\{j \middle| j \in N_{K_2}(s), PCC_s(s,j) > 0, j \neq s\right\}. \tag{9}$$

Given that the known data of a target QoS matrix are reliable (i.e., they are true user-service invocation records), an LF analysis-based model can assure that its resultant LFs well represent the highly valuable information hidden in them. From this point of view, we see the reason why we choose to perform LF analysis on a sparse QoS matrix as a pre-step before the neighborhood detection. Naturally, we can achieve neighborhood based on a sparse QoS matrix directly by computing PCC among its row/column entities, i.e., the involved users/services. However, the quality of the resultant neighborhood can be greatly impaired by the target matrix's sparsity. In comparison, a PLF model first extracts compact and dense LF matrices from the sparse QoS matrix, where each involved user/service is described by a dense LF vector. Moreover, such an LF vector is acquired on premise of well representing observed interactions in the target system. Thus, the neighborhood built on the LF matrices is expected to be far more reliable on one built on the original sparse QoS matrix.

3.3 Posterior-Neighborhood-Regularized LF Analysis

In this phase, we adopt the posterior-neighbors identified in phase 2 to help to extract two new LF matrices $P\prime$ and $Q\prime$. Considering similar users tend to experience similar QoS on similar services [11], [14], [15], we design the following loss function:

$$\begin{aligned}&\underset{P',Q'}{\arg\min}\,\varepsilon(P',Q')\\ &=\sum_{u=1}^{|U|}\left(\sum_{k\in T_u(u)} Su_{u,k}\sum_{n=1}^{f}\left(p'_{u,n}-p'_{k,n}\right)^2\right)\\ &+\sum_{s=1}^{|S|}\left(\sum_{j\in T_s(s)} Ss_{s,j}\sum_{n=1}^{f}\left(q'_{s,n}-q'_{j,n}\right)^2\right),\end{aligned} \tag{10}$$

where $Su_{u,k}$ and $Ss_{s,j}$ model the linear weight of user u's kth neighbor and service s's jth neighbor respectively based on the corresponding posterior similarities. To neutralize the impact of magnitudes, $Su_{u,k}$ and $Ss_{s,j}$ are given by

$$Su_{u,k} = \frac{PCC_u(u,k)}{\sum_{k'\in T_u(u)} PCC_u(u,k')},\ Ss_{s,j} = \frac{PCC_s(s,j)}{\sum_{j'\in T_s(s)} PCC_s(s,j')}. \tag{11}$$

From (10) and (11), we can see that a user's/service's most similar neighbor has the highest impact on the generalized loss. Moreover, with (10) we attempt to minimize the difference between LFs from the most similar neighbors, thereby keeping them similar as reflected in the posterior-neighborhood.

By integrating (10) into (2), we obtain the posterior-neighborhood-regularized objective function

$$\begin{aligned}\varepsilon(P',Q') = &\frac{1}{2}\sum_{(u,s)\in R_K}\left(r_{u,s}-\sum_{n=1}^{f}p'_{u,n}q'_{s,n}\right)^2\\ &+\frac{\lambda}{2}\sum_{(u,s)\in R_k}\left(\sum_{n=1}^{f}\left(p'_{u,n}\right)^2+\sum_{n=1}^{f}\left(q'_{s,n}\right)^2\right)\\ &+\frac{\alpha_1}{2}\sum_{u=1}^{|U|}\left(\sum_{k\in T_u(u)}Su_{u,k}\sum_{n=1}^{f}\left(p'_{u,n}-p'_{k,n}\right)^2\right)\\ &+\frac{\alpha_2}{2}\sum_{s=1}^{|S|}\left(\sum_{j\in T_s(s)}Ss_{s,j}\sum_{n=1}^{f}\left(q'_{s,n}-q'_{j,n}\right)^2\right),\end{aligned} \tag{12}$$

where the instant loss on a single instance $r_{u,s}$ is

$$\begin{aligned}\varepsilon_{u,s} = &\frac{1}{2}\left(r_{u,s}-\sum_{n=1}^{f}p'_{u,n}q'_{s,n}\right)^2+\frac{\lambda}{2}\left(\sum_{n=1}^{f}\left(p'_{u,n}\right)^2+\sum_{n=1}^{f}\left(q'_{s,n}\right)^2\right)\\ &+\frac{\alpha_1}{2}\sum_{k\in T_u(u)}Su_{u,k}\sum_{n=1}^{f}\left(p'_{u,n}-p'_{k,n}\right)^2\\ &+\frac{\alpha_2}{2}\sum_{j\in T_s(s)}Ss_{s,j}\sum_{n=1}^{f}\left(q'_{s,n}-q'_{j,n}\right)^2.\end{aligned} \tag{13}$$

To train the involved LFs, we apply SGD to (13)

$$On\ r_{u,s}, forn = 1\sim f: \begin{cases} p'_{u,n} \leftarrow p'_{u,n} - \eta\frac{\partial\varepsilon_{u,s}}{\partial p'_{u,n}} \\ q'_{s,n} \leftarrow q'_{s,n} - \eta\frac{\partial\varepsilon_{u,s}}{\partial q'_{s,n}}\end{cases}. \tag{14}$$

Note that after phases 1 and 2, $Su_{u,k}$ and $Ss_{s,j}$ are already known, they should be treated as constant in (14). Hence, we obtain the following training rules for a PLF model

$$On\ r_{u,s}, forn = 1\sim f: \begin{cases} p'_{u,n} \leftarrow p'_{u,n} + \eta q'_{s,n}\left(r_{u,s}-p'_{u,n}q'_{s,n}\right) \\ -\eta\lambda p'_{u,n} - \eta\alpha_1\sum_{k\in T_u(u)}Su_{u,k}\left(p'_{u,n}-p'_{k,n}\right), \\ q'_{s,n} \leftarrow q'_{s,n} + \eta p'_{u,n}\left(r_{u,s}-p'_{u,n}q'_{s,n}\right) \\ -\eta\lambda q'_{s,n} - \eta\alpha_2\sum_{j\in T_s(s)}Ss_{s,j}\left(q'_{s,n}-q'_{j,n}\right).\end{cases} \tag{15}$$

3.4 Algorithm Design and Analysis

In our design, a PLF model relies on two algorithms, i.e., Posterior-Neighborhood Construction (PNC), and Posterior-neighborhood-regularized Latent Factor Analysis (PLFA). Algorithm PNC constructs the posterior-neighborhood that is adopted by Algorithm PLFA. Algorithm PLFA calls Algorithm PNC to minimize (12) for predicting the missing QoS data. Their pseudocode and the time cost are given in Algorithm PNC and Algorithm PLFA, respectively. Next, we analyze their computational cost.

The computational complexity of Algorithm PNA is

$$\begin{aligned}C = &\Theta(1)+\Theta(1)+N_{mtr}\times(|R_K|\times f\times 2\times\Theta(1)+\Theta(1))+\Theta(1)\\ &+|U|\times\Theta(f)+|U|\times(|U|-1)/2\times\Theta(f)+|U|\times K_1\\ &+|S|\times\Theta(f)+|S|\times(|S|-1)/2\times\Theta(f)+|S|\times K_2+\Theta(1)\\ &\approx\Theta\Big(N_{mtr}\times|R_K|\times f+\left(|U|^2+|S|^2\right)\times f\Big).\end{aligned} \tag{16}$$

The computational complexity of Algorithm PLFA is

Algorithm PNC

Input: R
Output: $PCC_u(u,k)$, $T_u(u)$, $PCC_s(s,j)$, $T_s(s)$ — Cost

1	Initializing f, λ, η, N_{mtr}	$\Theta(1)$
2	**while** $t \le N_{mtr}$ && 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** $n = 1$ to f	$\times\ f$
5	computing $p_{u,n}$ according to formula (4)	$\Theta(1)$
6	computing $p_{s,n}$ according to formula (4)	$\Theta(1)$
7	**end for**	–
8	**end for**	–
9	$t = t+1$	$\Theta(1)$
10	**end while**	–
11	**for** $u = 1$ to $\lvert U\rvert$	$\times\ \lvert U\rvert$
12	computing $\bar{p}_u$ according to formula (6)	$\Theta(f)$
13	**end for**	–
14	**for** $u = 1$ to $\lvert U\rvert$	$\times\ \lvert U\rvert$
15	**for** $k = \mathrm{u}+1$ to $\lvert U\rvert$	$\times\ (\lvert U\rvert-1)/2$
16	computing $PCC_u(u,k)$ according to formula (5)	$\Theta(f)$
17	**end for**	–
18	**end for**	–
19	**for** $u = 1$ to $\lvert U\rvert$	$\times\ \lvert U\rvert$
20	computing $T_u(u)$ according to formula (7)	K_1
21	**end for**	–
22	**for** $s = 1$ to $\lvert S\rvert$	$\times\ \lvert S\rvert$
23	computing $\bar{q}_s$ according to formula (8)	$\Theta(f)$
24	**end for**	–
25	**for** $s = 1$ to $\lvert S\rvert$	$\times\ \lvert S\rvert$
26	**for** $j = \mathrm{s}+1$ to $\lvert S\rvert$	$\times\ (\lvert S\rvert-1)/2$
27	computing $PCC_s(s,j)$ according to formula (8)	$\Theta(f)$
28	**end for**	–
29	**end for**	–
30	**for** $s = 1$ to $\lvert S\rvert$	$\times\ \lvert S\rvert$
31	computing $T_s(s)$ according to formula (9)	K_2
32	**end for**	–
33	**return:** $PCC_u(u,k)$, $T_u(u)$, $PCC_s(s,j)$, $T_s(s)$	$\Theta(1)$

$$\begin{aligned}C = &\Theta(1)+|U|\times(\Theta(K_1)+|U|\times\Theta(1))\\ &+|S|\times(\Theta(K_2)\\ &+|S|\times\Theta(1))+N_{mtr}\times\Theta(|U|\times f\times K_1\\ &+|S|\times f\times K_2+|R_K|\times f\times 2)+\Theta(1)\\ &\approx\Theta(|U|^2+|S|^2+N_{mtr}\times(|R_K|+|U|\times K_1\\ &+|S|\times K_2)\times f).\end{aligned} \tag{17}$$

In real-world scenarios, there are only a limited number of users or services that are similar to the target user or service in their QoS experiences on co-invoked services. As a result, K_1 and K_2 are much smaller than $|U|$ and $|S|$ for a PLF model. Thus, the overall computational complexity of Algorithm PLFA is

$$C\approx\Theta\Big(|U|^2+|S|^2+N_{mtr}\times|R_K|\times f\Big). \tag{18}$$

Finally, the total computational complexity of building a PLF model is

$$C \approx \Theta\Big(N_{mtr} \times |R_K| \times f + \Big(|U|^2 + |S|^2\Big) \times f\Big) + \Theta\Big(|U|^2 + |S|^2 + N_{mtr} \times |R_K| \times f\Big) \approx \Theta\Big((|U|^2 + |S|^2) \times f\Big) + \Theta(N_{mtr} \times |R_K| \times f). \tag{19}$$

In Section 4, we will theoretically and experimentally demonstrate that the computational complexity of Algorithm PLFA is comparable to the state-of-the-art LF-based QoS predictors with neighborhood-regularization.

Algorithm PLFA

Input: R, $\mathrm{PCC_u(u,k)}$, $\mathrm{T_u(u)}$, $\mathrm{PCC_s(s,j)}$, $\mathrm{T_s(s)}$
Output: $\hat{R}$ — **Cost**

1	Calling Algorithm PNC	$\Theta(1)$
2	Initializing f, λ, η, $\mathrm{N_{mtr}}$	$\Theta(1)$
3	**for** $\mathrm{u} = 1$ to $\mathrm{|U|}$	$\times \mathrm{|U|}$
4	**computing** $\sum_{\mathrm{k'} \in \mathrm{T_u(u)}} \mathrm{PCC_u(u,k')}$	$\Theta(\mathrm{K_1})$
5	**for** $\mathrm{k} = 1$ to $\mathrm{|U|}$	$\times \mathrm{|U|}$
6	computing $\mathrm{Su_{u,k}}$ according to formula (11)	$\Theta(1)$
7	**end for**	–
8	**end for**	–
9	**for** $\mathrm{s} = 1$ to $\mathrm{|S|}$	$\times \mathrm{|S|}$
10	**computing** $\sum_{\mathrm{j'} \in \mathrm{T_s(s)}} \mathrm{PCC_s(s,j')}$	$\Theta(\mathrm{K_2})$
11	**for** $\mathrm{s} = 1$ to $\mathrm{|S|}$	$\times \mathrm{|S|}$
12	computing $\mathrm{Ss_{s,j}}$ according to formula (11)	$\Theta(1)$
13	**end for**	–
14	**end for**	–
15	**while** $\mathrm{t} \leq \mathrm{N_{mtr}}$ && not converge	$\times \mathrm{N_{mtr}}$
16	**for** $\mathrm{u} = 1$ to $\mathrm{|U|}$	$\times \mathrm{|U|}$
17	**for** $\mathrm{n} = 1$ to f	$\times$ f
18	computing $\eta\alpha_1 \sum_{\mathrm{k} \in \mathrm{T_u(u)}} \mathrm{Su_{u,k}(p_{u,n} - p_{\mathit{k},n})}$	$\Theta(\mathrm{K_1})$
19	**end for**	–
20	**end for**	–
21	**for** $\mathrm{s} = 1$ to $\mathrm{|S|}$	$\times \mathrm{|S|}$
22	**for** $\mathrm{n} = 1$ to f	$\times$ f
23	computing $\eta\alpha_2 \sum_{\mathrm{j} \in \mathrm{T_s(s)}} \mathrm{Su_{s,j}(q_{s,n} - q_{j,n})}$.	$\Theta(\mathrm{K_1})$
24	**end for**	–
25	**end for**	–
26	**for** each known entry $\mathrm{r_{u,s}}$ in R // $\mathrm{r_{u,s}} \in \mathrm{R_K}$	$\times \mathrm{|R_K|}$
27	**for** $\mathrm{n} = 1$ to f	$\times$ f
28	computing $\mathrm{pd'_{u,n}}$ according to formula (15)	$\Theta(1)$
29	computing $\mathrm{p'_{s,n}}$ according to formula (15)	$\Theta(1)$
30	**end for**	–
31	**end for**	–
32	$\mathrm{t} = \mathrm{t} + 1$	$\Theta(1)$
33	**end while**	–
34	**return** P, Q	$\Theta(1)$

4 Experiments and Results

4.1 Datasets

To evaluate the PLF model, we conduct extensive experiments on a benchmark dataset named *Response Time*. It contains real-world web service QoS data and has been commonly used in prior research on QoS prediction [2], [6], [10], [11], [14], [15], [29], [30], [31], [33]. It records the response times of 5,825 web services experienced by 339 users' 1,873,838 invocations. Different test cases are designed to validate the performance of PLF. Table 2 summarizes the properties of all the test cases, where the column "Density" indicates the density of the training data. Each test case is repeated 10 times and the results are averaged.

TABLE 2
Properties of All the Designed Test Cases

Dataset	No.	Density	Training data	Testing data
Response Time	D1	5%	93,692	1,780,146
	D2	10%	187,384	1,686,454
	D3	15%	281,076	1,592,762
	D4	20%	374,768	1,499,070

4.2 Evaluation Protocol

QoS prediction aims to predict the unknown QoS data based on the known ones. Hence, this work mainly focuses on the prediction accuracy, i.e., the closeness between the prediction results and the actual values. In our experiments, we employ the mean absolute error (MAE) and the root mean squared error (RMSE), which are also widely used for researching QoS prediction [11], [14], [15], [29], [30], [31], [33], to evaluate the prediction accuracy of PLF:

$$MAE = \left(\sum_{(w,j)\in\Gamma} \left|r_{w,j} - \hat{r}_{w,j}\right|_{abs}\right) \Big/ |\Gamma|, \tag{20}$$

$$RMSE = \sqrt{\left(\sum_{(w,j)\in\Gamma} \left(r_{w,j} - \hat{r}_{w,j}\right)^2\right) \Big/ |\Gamma|}, \tag{21}$$

where a lower MAE or RMSE value denotes a higher prediction accuracy. All the experiments are run with 3.7 GHz i7 central processing unit (CPU) and 64 GB random access memory.

4.3 Impacts of K_1 and K_2

First, we analyze the impacts of K_1 and K_2 on the prediction accuracy of PLF. In this set of experiments, the parameters are set as $\alpha_1 = 0.2$, $\alpha_2 = 0.2$, $f = 20$, $\lambda = 0.01$, and $\eta = 0.01$, uniformly. Figs. 4 and 5 shows the experimental results on MAE and RMSE respectively when K_1 increases from 0 to 50 and K_2 from 0 to 100. From these figures, we have the following observations.

When K_1 and K_2 are both set as zero, PLF has the highest MAE and RMSE in almost all the test cases because it is equivalent to an LF model. This observation confirms that the posterior-neighborhood-based regularization is useful in improving the prediction accuracy for an LF model.

When K_1 or K_2 is set as zero, PLF has a relatively high MAE and RMSE in all the test cases except for the MAE in one situation where K_2 is set as zero on D1. The reason for this exception is that the density of D1 is low (5 percent) and the number of web services (5,825) is much larger than that of the users (339). In such a situation, more posterior-neighborhood information is extracted from the users than the services. As a result, the users' posterior-neighborhood-based regularization is more useful than the services' in improving the prediction accuracy of PLF.

As K_1 and K_2 increase, the MAE and RMSE of PLF decrease at first and then increase in general. For example,

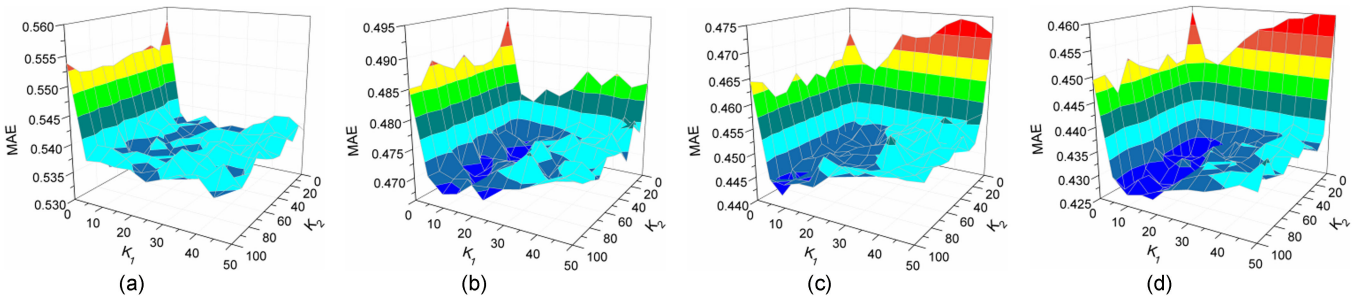


Fig. 4. MAE of PLF when K_1 increases from 0 to 50 and K_2 increases from 0 to 100: (a) D1, (b) D2, (c) D3, (d) D4.

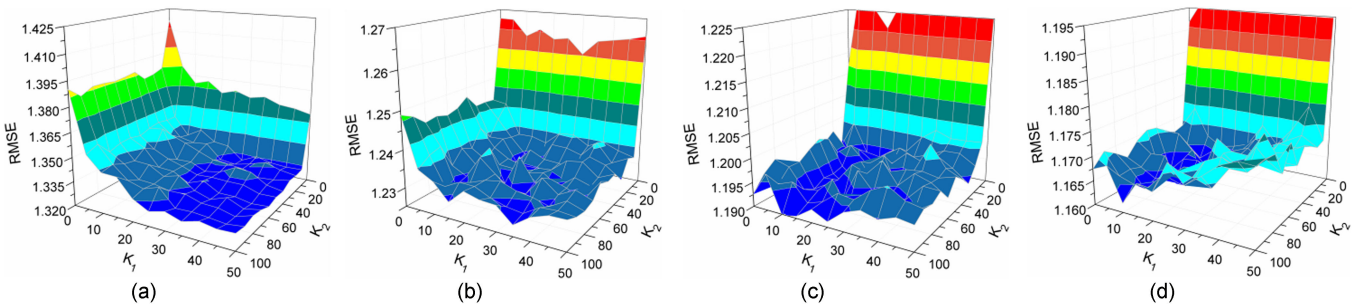


Fig. 5. RMSE of PLF when K_1 increases from 0 to 50 and K_2 increases from 0 to 100: (a) D1, (b) D2, (c) D3, (d) D4.

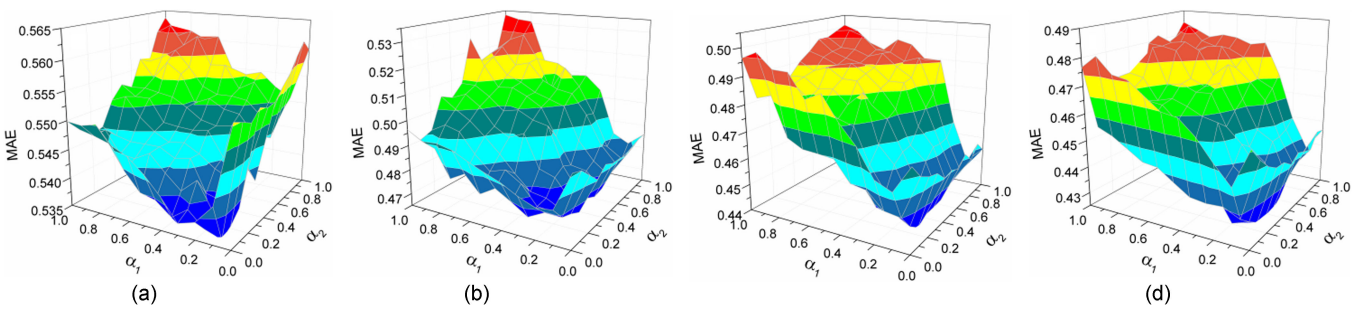


Fig. 6. MAE of PLF when α_1 and α_2 increase from 0 to 1: (a) D1, (b) D2, (c) D3, (d) D4.

on D4, since more similar users/services are included in PLF as K_1 and K_2 increase, the MAE decreases from 0.4589 to 0.4238 at first. However, when K_1 and K_2 become too large, some dissimilar users/services are also introduced into PLF and jeopardize its prediction accuracy.

As the density of the training data increases, PLF tends to achieve the lowest MAE and RMSE on the smaller K_1 and K_2. For example, on D1 (5 percent), PLF achieves the lowest RMSE when K_1 is in the range of 40—50 and K_2 is in the range of 50—100. On D4 (20 percent), the optimal ranges of K_1 and K_2 for PLF decrease to 10—20 and 40—60 respectively. The major reason is that more training data contains more posterior-neighborhood information, which is beneficial for PLF in finding highly similar users/services. In other words, some dissimilar users/services, which were included in the prediction model at the beginning, will be discovered and discarded finally as the density of the training data increases. According to this observation, we set $K_1 = 15$ and $K_2 = 50$ as the default setting in the next experiments.

In conclusion, these observations verify that the posterior-neighborhood-based regularization is significantly effective in improving the prediction accuracy of PLF.

4.4 Impacts of α_1 and α_2

This set of experiments focuses on the impacts of α_1 and α_2 when their values increase from 0 to 1, as shown in Figs. 6 and 7. The other parameters are set as $K_1 = 15$, $K_2 = 50$, $f = 20$, $\lambda = 0.01$, and $\eta = 0.01$, uniformly. As demonstrated, the MAE and RMSE of PLF decrease as α_1 and α_2 increase at the beginning. After reaching a certain threshold, as α_1 and α_2 continue to increase, the MAE and RMSE of PLF increase. The reason is that when α_1 and α_2 are too large, PLF will heavily rely on the posterior-neighborhood-based regularization, which leads to underfitting. In real-world applications, the datasets generated from different domains differ from each other vastly in user count, web service count, instance count, and data density. The optimal α_1 and α_2 depend heavily on such data characteristics. Hence, α_1 and α_2 are domain-specific, they should be experimentally inspected and set for PLF to achieve high prediction accuracy.

4.5 Impact of *f*

This set of experiments analyzes the impact of f with $K_1 = 15$, $K_2 = 50$, $\alpha_1 = 0.1$, $\alpha_2 = 0.4$, $\lambda = 0.01$, and $\eta = 0.01$. Fig. 8 shows the experimental results with f increasing from 10, 20, 40, 80, 160 to 320. A higher-dimensional LF space

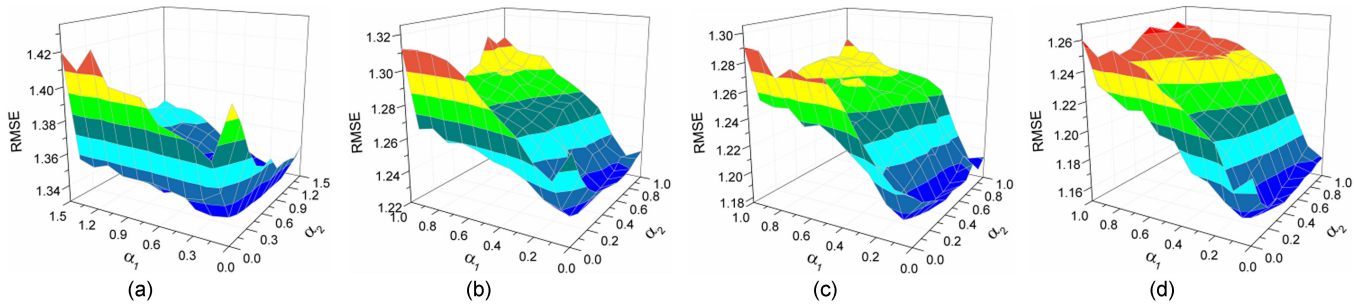


Fig. 7. RMSE of PLF when α_1 and α_2 increase from 0 to 1: (a) D1, (b) D2, (c) D3, (d) D4.

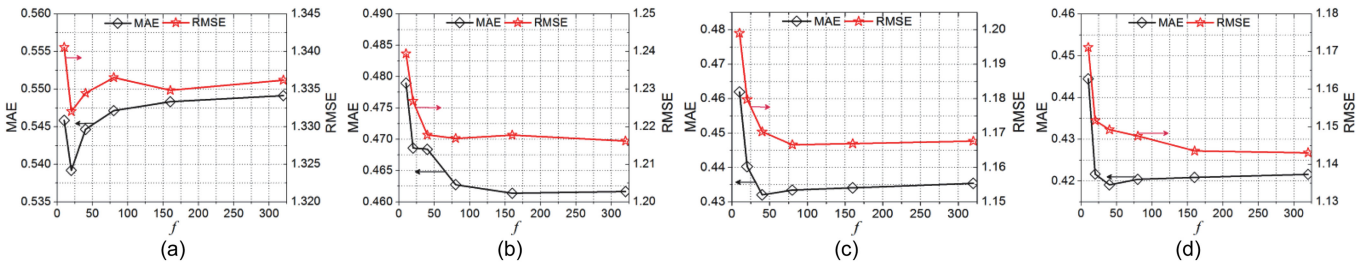


Fig. 8. MAE and RMSE of PLF with different *f*. (a) D1, (b) D2, (c) D3, (d) D4.

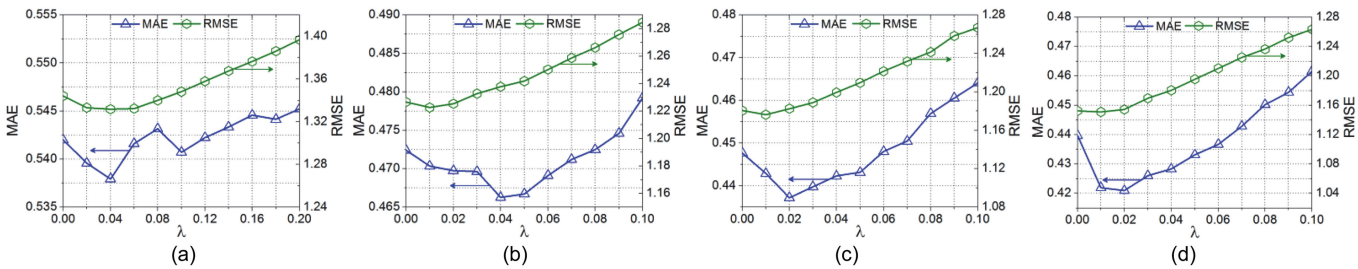


Fig. 9. MAE and RMSE of PLF as λ increases: (a) D1, (b) D2, (c) D3, (d) D4.

usually provides PLF with better representation learning ability [6], [17]. Fig. 8 shows that the MAE and RMSE of PLF decrease as f increases in general in all the test cases. Moreover, it shows that the MAE and RMSE of PLF tend to decrease slightly or even increase after f exceeds a certain threshold. The reason is that when an appropriate value is assigned to f, PLF obtains the highest representation learning ability. An extra increase in f cannot bring significant improvement in the prediction accuracy. Instead, it increases the probability of overfitting which lowers prediction accuracy. Besides, the computational cost of PLF also increases linearly with f. Hence, we set $f = 20$ to balance the computational cost and the prediction accuracy in all the experiments.

4.6 Impact of λ

In this set of experiments, we analyze the impact when λ increases. The other parameters are set as $K_1 = 15$, $K_2 = 50$, $\alpha_1 = 0.1$, $\alpha_2 = 0.4$, $f = 20$, and $\eta = 0.01$, uniformly. The results are shown in Fig. 9. Since the l_2-norm regularization can prevent PLF from overfitting, we find that the MAE and RMSE of PLF decrease at first as λ increases in all the test cases. However, after λ exceeds a certain point, the MAE and RMSE of PLF increase. This indicates that PLF is largely impacted by the l_2-norm regularization. From the experimental results, we conclude that λ is crucial for PLF to achieve highly accurate prediction results. Since λ is also dependent on the data characteristics [38], its optimal value needs to be experimentally inspected.

4.7 Comparisons Between PLF and State-of-the-Art Models

Finally, we theoretically and experimentally compare PLF with seven related state-of-the-art models in their prediction accuracy and computational efficiency. These compared models are three LF-based models (BMF, NIMF, and RSNMF), three LF-based models with additional geographical information (NAMF, GeoMF, and LMF-PP), and one deep neural network (DNN) based model (AutoRec). Note that NIMF, NAMF, GeoMF, and LMF-PP are all prior-neighborhood-based, while PLF is posterior-neighborhood-based. These compared models have different characteristics as given in Table 3. Besides, their computational complexity is concluded in Table 4, where there are two major parts, one is for constructing the neighborhood regularization and the other one is for minimizing the objective function. Since AutoRec's computational complexity is obviously much higher than its peers due to its DNN structure [43], we do not summarize its computational complexity in Table 4.

In prediction accuracy, the dimension of LF is set as $f = 20$ for all the models except for AutoRec—because it is

TABLE 3
Descriptions of All the Compared Models

Models	Descriptions
BMF	The basic matrix factorization (MF) model [38] which consists of two situations, i.e., with and without linear biases. We pick the best one for comparisons on each test case.
NIMF	The neighborhood-integrated MF model [16] extends BMF by employing the information of similar users.
RSNMF	The regularized single element dependent non-negative MF model [6] is designed for QoS prediction by incorporating regularization into a non-negative MF.
NAMF	The network-aware MF model [15] employs the additional network distances information to construct the neighborhood.
GeoMF	The improved MF model [14] employs the additional geographical relationships to construct the neighborhood.
LMF-PP	The location-based MF model [11] employs the additional location information to construct the neighborhood.
AutoRec	The deep neural networks based model [41] employs an autoencoder [42] framework for CF and consists of I-AutoRec and U-AutoRec. The best one is chosen to compare.
PLF	The neighborhood of latent-factor based MF model proposed in this paper.

TABLE 4
The Computational Complexities of All the Models

Model	Complexity of constructing neighborhood regularization	Complexity of minimizing objective function
BMF [38]	/	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
NIMF [16]	$\Theta(\lvert U \rvert^2 \times \lvert S \rvert)$	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f \times K_1^2)$
RSNMF [6]	/	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
NAMF [15]	$\Theta(\lvert U \rvert^2)$	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
GeoMF [14]	$\Theta(\lvert U \rvert^2 \times \lvert S \rvert + \lvert S \rvert^2 \times \lvert U \rvert)$	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f^2 \times (K_1 + K_2))$
LMF-PP [11]	$\Theta(\lvert U \rvert^2 \times \lvert S \rvert + \lvert S \rvert^2 \times \lvert U \rvert)$	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$
PLF	$\Theta((\lvert U \rvert^2 + \lvert S \rvert^2) \times f)$	$\Theta(N_{mtr} \times \lvert R_K \rvert \times f)$

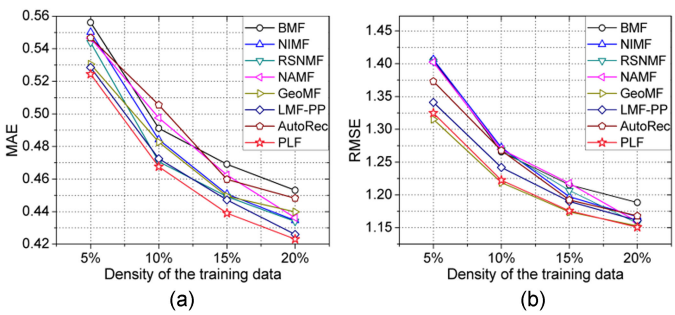


Fig. 10. The compared results on all the test cases: (a) MAE, (b) RMSE.

a deep neural networks based model—to facilitate fair comparisons. Besides, all other parameters involved in the compared models are all set as instructed in their corresponding papers. For PLF, the other parameters are set as $K_1 = 15$, $K_2 = 50$, $\alpha_1 = 0.1$, $\alpha_2 = 0.4$, and $\eta = 0.01$. Fig. 10 records the compared results. It shows that PLF achieves the lowest MAE in all the test cases, which are 4.8—6.62 percent, 2.62—4.69 percent, 0.66—3.57 percent, 3—6.03 percent, 1.15—3.8 percent, 0.78—1.83 percent, and 4.08—7.5 percent lower than that achieved by BMF, NIMF, RSNMF, NAMF, GeoMF, LMF-PP, AutoRec respectively. In RMSE, PLF and GeoMF perform better than the other models in all the test cases. Furthermore, the RMSE achieved by PLF is slightly worse than GeoMF on D1 and D2.

TABLE 5
Statistical Results of Prediction Accuracy by Conducting Wilcoxon Signed-Ranks Test With A Significance Level of 0.05

Comparison	$R+$	$R-$	p-value
PLF vs. BMF	36	0	**0.0039**
PLF vs. NIMF	36	0	**0.0039**
PLF vs. RSNMF	36	0	**0.0039**
PLF vs. NAMF	36	0	**0.0039**
PLF vs. GeoMF	27	9	0.1250
PLF vs. LMF-PP	36	0	**0.0039**
PLF vs. AutoRec	36	0	**0.0039**

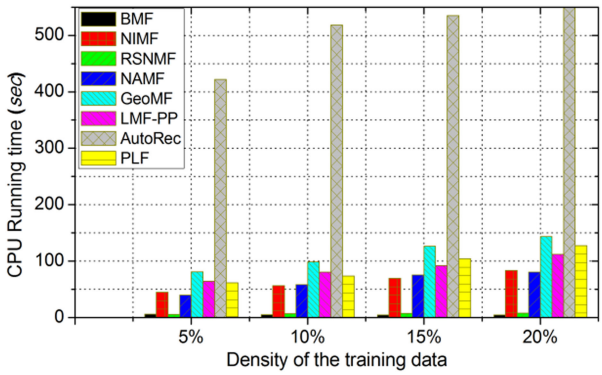


Fig. 11. The compared results of CPU running time on all the test cases.

To validate whether PLF achieves significantly higher prediction accuracy than the other models, the Wilcoxon signed-ranks test [44], which is a nonparametric pairwise comparison procedure, is applied to perform the statistical analysis. Table 5 records the statistical results, where three columns respectively show the achieved rankings $R+$ and $R-$ values and its associated p-value. A larger $R+$ value indicates that PLF has better prediction accuracy. The accepted hypotheses that PLF outperforms the other models with a significance level of 0.05 are highlighted in bold. Table 5 clearly validates that PLF has significantly higher prediction accuracy than the other models except for GeoMF. However, we can see that PLF also achieves much higher $R+$ rankings than GeoMF, which indicates that PLF has slightly higher prediction accuracy than GeoMF. More importantly, PLF does so without the need for the additional geographical information that is required by GeoMF.

Considering the computational efficiency of involved models, we summarize their CPU running time in all testing cases in Fig. 11. From Fig. 11 and Table 4, we have several observations:

a) BMF and RSNMF have no computational cost in constructing neighborhood regularization because they do not consider neighborhood regularization, making them consume less time than their peers do. However, they are generally outperformed by their peers in prediction accuracy for missing data.
b) PLF's computational efficiency is higher than that of GeoMF and LMF-PP because of $f << \min\{|U|, |S|\}$.
c) For minimizing the objective function, NIMF and GeoMF are slower than the other models. In other

TABLE 6
Properties of All the Designed Test Cases on the Synthetic Large Scale Dataset

Dataset	No.	Density	Training data	Testing data
Synthetic large scale dataset	D_S1	5%	4,684,595	89,007,305
	D_S2	10%	9,369,190	84,322,710
	D_S3	15%	14,053,785	79,638,115
	D_S4	20%	18,738,380	74,953,520

TABLE 7
Compared Results on Prediction Accuracy

Test cases	RMSE			MAE		
	BMF	RSNMF	PLF	BMF	RSNMF	PLF
D_S1	1.0715	1.0589	**1.0085**	0.4053	0.3849	**0.3758**
D_S2	1.0181	0.9986	**0.9804**	0.3959	0.3611	**0.3584**
D_S3	1.0018	0.9666	**0.9498**	0.3964	0.3524	**0.3511**
D_S4	0.9948	0.9497	**0.9458**	0.3977	0.3494	**0.3488**

words, the PLF, BMF, RSNMF, NAMF and LMF-PP models have the lowest computational complexity.

d) AutoRec has the most CPU running time among all the models due to its DNN-based structure.

e) PLF's computational efficiency is comparable to that of NIMF, NAMF, GeoMF, and LMF-PP.

Based on the experimental results and the analyses in this section, we conclude that PLF has better performance than the seven state-of-the-art models because a) it has the best prediction accuracy, and b) when comparing with NAMF, GeoMF, and LMF-PP, it has comparable computational complexity and does not require additional geographical information.

4.8 Performance on Large Scale Dataset

As analyzed in Section 4.7, PLF consumes more CPU running time than the ordinary LF-based QoS predictors without neighborhood regularization (BMF and RSNMF). To evaluate PLF's performance on a large scale dataset, we generate a synthetic large scale dataset by introducing Gaussian noise to the dataset *Response Time*. The specific method of generating synthetic dataset is: a) for each known entry $r_{u,s}$ in $Response\,Time\ R^{|339|\times|5825|}$, generating the corresponding synthetic entry $g_{u,s}$ by the formula $g_{u,s} = r_{u,s} + r_{u,s} \times 0.05 \times rand()$, where $rand()$ indicates a random real number that obeys the standard normal distribution, then we have a synthetic dataset $G^{|339|\times|5825|}$, b) repeating the step a for 50 times to obtain 50 different datasets $G^{|339|\times|5825|}$, and c) merging the 50 different datasets by rows to form a large scale dataset $G^{|16950|\times|5825|}$. After that, the original *Response Time* is expanded from 339 users to 16950 users, and the total number of invocations is increased from 1,873,838 to 93,691,900. Then, different test cases as designed in Table 2 are also used to validate PLF's performance on the synthetic large scale dataset, as shown in Table 6.

Then, we compare PLF with BMF and RSNMF (without neighborhood regularization) on prediction accuracy and computational efficiency. The parameters are set the same as in Section 4.7. Tables 7 and 8 record the compared results.

TABLE 8
Compared Results of CPU Running Time (sec)

Test cases	BMF	RSNMF	PLF*
D_S1	**2348.91**	11461.05	88.21+9979.38
D_S2	**2301.39**	2690.64	76.88+9507.93
D_S3	**1205.52**	2946.77	80.17+6855.11
D_S4	**524.73**	3341.05	75.16+6328.33

**The first part before + is for constructing neighborhood regularization and the second part after + is for minimizing objective function.*

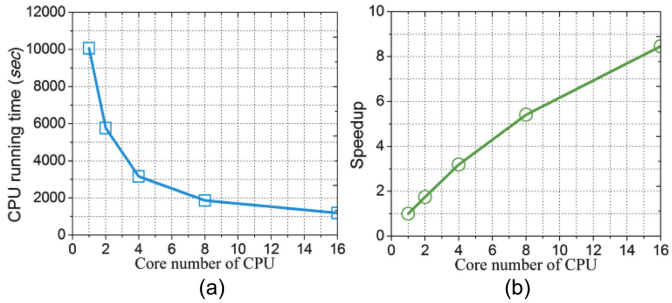


Fig. 12. PLF's performance on DS1 as the core number of CPU increases: (a) CPU running time, (b) Speedup.

From them, we observe that: a) PLF has better prediction accuracy while consumes more CPU running time than BMF and RSNMF, b) PLF consumes much less CPU running time for constructing neighborhood regularization than for minimizing the objective function, and c) all the models consume less CPU running time as density increases, which means that more training data can speed up their convergence. These results validate that on the large scale dataset, PLF performs well on prediction accuracy while its computational efficiency is not outstanding.

As analyzed in [45], [46], an LF-based model's computational efficiency can be greatly improved by implementing parallelization. Especially, for an LF-based model, its different users'/services' LFs can be simultaneously trained by the alternating stochastic gradient descent (ASGD) without impacting prediction accuracy [46]. On this basis, we develop PLF to a parallel version that can parallel minimizing its objective function (because PLF consumes most time to minimize the objective function), please refer to [46] for details. Then, we test PLF's computational efficiency on D_S1 as the core number of CPU increases, the results are recorded in Fig. 12. From it, we can see that PLF's computational efficiency has been improved as the core number of CPU increases. Specially, we find that the CPU running time with 8 cores (1860.91) is less than BMF (2348.91 seconds), which means that PLF has higher computational efficiency than BMF when it employs more than 8 cores to develop a parallel version.

Hence, based on the above analyses, we conclude that on the large scale dataset, PLF not only performs well on prediction accuracy but also has high computational efficiency by implementing parallelization.

5 Related Work

The proposed PLF is an LF-based model. An LF model originates from matrix factorization [6], [17], [47] and has been

widely used to develop the QoS predictor due to high scalability and prediction accuracy [16], [29], [30], [31], [33], [38]. An LF-based QoS predictor is built on a low-rank approximation to a user-service matrix based only on the known entries in the matrix. It maps both users/services into the same low-dimensional LF space, trains desired LFs on the known entries in the user-service matrix, and then predicts missing entries (i.e., QoS data) in the matrix heavily based on resultant LFs [6], [17], [30], [38].

Since similar users usually experience similar QoS when invoking similar services, neighborhood information can be used to improve an LF-based QoS predictor's prediction accuracy. Zheng *et al*. first propose a neighborhood-integrated LF-based QoS predictor based only on the user-service matrix [16]. Next, they develop their model by integrating the network map of users [15]. After that, Chen *et al*. propose an LF-based QoS predictor that incorporates the knowledge of geographical neighborhoods [14], Ryu *et al*. propose an LF-based QoS predictor by using the location information of users/service [11], and Feng and Huang propose a neighborhood enhanced LF-based QoS predictor by systematically considering geographical information, sample set diversity computation and platform context [33]. These proposed QoS predictors have the common point that the neighborhood information, which is incorporated into an LF model as the regularization term, is defined on sparse QoS data and/or geographical information.

In this study, PLF is significantly different from existing QoS predictors. First, existing predictors without neighborhood regularization [6], [17], [30], [38] commonly suffer low prediction accuracy because they ignore relationships among involved users/services. In comparison, PLF models such relationships via constructing the neighborhood among them accurately based on the dense LF matrices well representing the known data of a sparse QoS matrix. Second, existing predictors with neighborhood regularization [11], [14], [15], [33] require additional geographical information to construct users'/services' neighborhood, while PLF relies solely on the user-service matrix and does not require any additional information. It is expected to achieve even higher prediction accuracy for missing QoS data by combining heterogeneous neighborhoods originated from different data source.

Generally, the performance of a web service changes over time because of variations of network conditions and service status. Hence, it is necessary to make the QoS predictors time-aware. To do so, Zhang *et al*. propose a time-aware personalized QoS predictor by employing a tensor factorization model [48]. Zhang *et al*. extend the model proposed in [48] by considering the non-negative constraint [49]. Next, some other time-aware QoS predictors are proposed, including spatial-temporal model [50], time-aware collaborative filtering model [51], time-aware and sparsity-tolerant model [52], and time-aware and privacy-preserving model [53]. Besides, Syu *et al* survey the recent progress in time-aware QoS predictors [54]. For PLF, it can also be extended to a time-aware QoS predictor by considering the time information.

6 Conclusions

In this paper, we propose a posterior-neighborhood-regularized latent factor (PLF) model for achieving highly accurate Quality-of-Service (QoS) prediction for web services. PLF goes through three phases: primal LF extraction, posterior neighbor identification, and posterior-neighborhood-regularized LF analysis. Extensive experiments are conducted on a real-world benchmark dataset to evaluate PLF against the state-of-the-art models. The experimental results validate that: a) users'/services' posterior-neighborhood information is significantly effective for PLF to improve its prediction accuracy, b) PLF has significantly better performance than the state-of-the-art models in both prediction accuracy and prediction efficiency, and c) PLF relies only on the user-service matrix and does not require any additional information. Besides, PLF has also been demonstrated that it performs well in terms of both prediction accuracy and computational efficiency on the large scale dataset.

Although PLF has shown promising prospects, several issues remain unveiled: 1) how to make the parameters of PLF self-adaptive based on particle swarm optimization [55] so that they do not need to be inspected experimentally, 2) how to extend PLF to iteratively and dynamically identify users'/services' neighborhood based on a deep-random-forest structure [43], 3) as discussed in Section 3, the first two steps of a PLF model focus on identifying neighborhood based on an QoS matrix. Given that a QoS matrix can be extremely sparse, it is necessary to investigate other efficient neighborhood detectors like a random walk-based one [56], [57] for a PLF model, and 4) as discussed in prior research [48], [49], [50], [51], [52], [53], [54], modeling and acquisition of temporal dynamics in QoS data is a vital issue in services computing society. It appears highly important to further develop the temporal-aware extension of a PLF model. We plan to address them in our future work.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 61702475, 61772493, 61902370, 91646114, and 61602434, in part by the Natural Science Foundation of Chongqing (China) under Grants cstc2019jcyj-msxmX0578 and cstc2019jcyjjqX0013, and in part by the Pioneer Hundred Talents Program of Chinese Academy of Sciences. Di Wu and Qiang He are co-first authors of this article..

References

[1] B. Bai, Y. Fan, W. Tan, and J. Zhang, "DLTSR: A deep learning framework for recommendation of long-tail Web services," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2017.2681666.

[2] Z. Zheng, Y. Zhang, and M. R. Lyu, "Distributed qos evaluation for real-world web services," in *Proc. 2010 IEEE Int. Conf. Web Services*, 2010, pp. 83–90.

[3] 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 Services*, 2009, pp. 437–444.

[4] L. Purohit and S. Kumar, "A classification based Web service selection approach," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2018.2805352.

[5] R. Lu, X. Jin, S. Zhang, M. Qiu, and X. Wu, "A study on big knowledge and its engineering issues," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 9, pp. 1630–1644, Sep. 2019.

[6] 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 Netw. Learn. Syst.*, vol. 27, no. 3, pp. 524–537, Mar. 2016.

[7] 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, pp. 881–890.

[8] 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.

[9] 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.

[10] K. Lee, J. Park, and J. Baik, "Location-based Web service QoS prediction via preference propagation for improving cold start problem," in *Proc. IEEE Int. Conf. Web Services*, 2015, pp. 177–184.

[11] 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.

[12] X. Chen, X. Liu, Z. Huang, and H. Sun, "Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation," in *Proc. IEEE Int. Conf. Web Services*, 2010, pp. 9–16.

[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, Apr.-Jun. 2011.

[14] 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.

[15] 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.

[16] 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, Jul.-Sep. 2013.

[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," *Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 579–592, 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. Surveys*, 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. IEEE Int. Conf. Data Mining*, 2017, pp. 811–816.

[20] H. Zhao, Q. Yao, J. T. Kwok, and D. L. Lee, "Collaborative filtering with social local models," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 645–654.

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

[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.*, 2014, pp. 1512–1520.

[23] Y. Cai, H. Leung, Q. Li, J. Tang, and J. Li, "TyCo: Towards 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," *Recommender Systems Handbook*. Berlin, Germany: Springer, 2015, pp. 77–118.

[25] M. Wang, X. Zheng, Y. Yang, and K. Zhang, "Collaborative filtering with social exposure: A modular approach to social recommendation," in *Proc. 33rd Nat. Conf. Artif. Intell.*, 2018, pp. 1–8.

[26] X. Luo, Z. Liu, S. Li, M. Shang, and Z. Wang, "A fast non-negative latent factor model based on generalized momentum method," *IEEE Trans. Syst., Man., Cybern., Syst.*, to be published, doi: 10.1109/TSMC.2018.2875452.

[27] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. World Wide Web Conf.*, to be published, doi: 10.1109/TSMC.2018.2872842.

[28] L. Wu, P. Sun, R. Hong, Y. Ge, and M. Wang, "Collaborative neural social recommendation," *IEEE Trans. Syst., Man, Cybern., Syst.*, pp. 1–13, 2018.

[29] 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.

[30] 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.

[31] 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. IEEE Int. Conf. Web Services*, 2015, pp. 161–168.

[32] A. Liu *et al.*, "Differential private collaborative Web services QoS prediction," *World Wide Web*, pp. 1–24, 2018, doi: 10.1007/s11280-018-0544-7.

[33] Y. Feng and B. Huang, "Cloud manufacturing service QoS prediction based on neighbourhood enhanced matrix factorization," *J. Intell. Manuf.*, pp. 1–12, 2018, doi: 10.1007/s10845-018-1409-8.

[34] 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.

[35] M. Wang, X. Zheng, Y. Yang, and K. Zhang, "Collaborative filtering with social exposure: A modular approach to social recommendation," in *Proc. 33rd Nat. Conf. Artif. Intell.*, 2017, pp. 1–8.

[36] 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.

[37] 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.

[38] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Comput.*, vol. 42, no. 8, pp. 30–37, 2009.

[39] D. Wu, X. Luo, M. Shang, Y. He, G. Wang, and M. Zhou, "A deep latent factor model for high-dimensional and sparse matrices in recommender systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, doi: 10.1109/TSMC.2019.2931393.

[40] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Qos-aware web service recommendation by collaborative filtering," *IEEE Trans. Services Comp.*, vol. 4, no. 2, pp. 140–152, Apr.-Jun. 2015.

[41] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec:Autoencoders meet collaborative filtering," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 111–112.

[42] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[43] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, *arXiv:1702.08835*.

[44] 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. Informat.*, vol. 14, no. 3, pp. 909–920, Mar. 2018.

[45] 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.

[46] 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.

[47] L. Yao, X. Wang, Q. Z. Sheng, B. Benatallah, and C. Huang, "Mashup recommendation by regularizing matrix factorization with API Co-invocations," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2018.2803171.

[48] Y. Zhang, Z. Zheng, and M. R. Lyu, "WSPred: A time-aware personalized QoS prediction framework for Web services," in *Proc. IEEE 22nd Int. Symp. Softw. Rel. Eng.*, 2011, pp. 210–219.

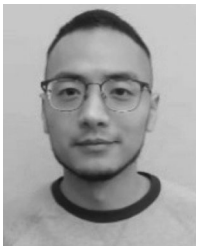
[49] W. Zhang, H. Sun, X. Liu, and X. Guo, "Temporal QoS-aware web service recommendation via non-negative tensor factorization," in *Proc. ACM 23rd Int. Conf. World Wide Web*, 2014, pp. 585–596.

[50] 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.

[51] C. Yu and L. Huang, "Time-aware collaborative filtering for QoS-based service recommendation," in *Proc. IEEE Int. Conf. Web Services*, 2014, pp. 265–272.

[52] W. Chen, W. Qiu, X. Wang, Z. Zheng, and X. Yang, "Time-aware and sparsity-tolerant QoS prediction based on collaborative filtering," in *Proc. IEEE Int. Conf. Web Services*, 2016, pp. 637–640.

[53] 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.
[54] 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. Int. Conf. Services Comput.*, 2018, pp. 36–50.
[55] B. Chopard and M. Tomassini, "Particle swarm optimization," *An Introduction to Metaheuristics for Optimization*. Berlin, Germany: Springer, 2018, pp. 97–102.
[56] M. Tang, X. Dai, B. Cao, and J. Liu, "Wswalker: A random walk method for QoS-aware Web service recommendation," in *Proc. IEEE Int. Conf. Web Services*, 2015, pp. 591–598
[57] M. Jamali and M. Ester, "Trustwalker: A random walk model for combining trust-based and item-based recommendation," in *Proc. 15th Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 397–406.



Di Wu (SM'18–M'19) received the BS degree in applied physics from the Nanjing University of Science and Technology, Jiangsu, China, in 2009, the MS degree in optical engineering from the Chongqing University, Chongqing, China, in 2012, and the PhD degree in computer science from Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China, in 2019. 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. He is a member of the IEEE.

Qiang He received the first PhD degree from the Swinburne University of Technology, Australia, in 2009, and the second PhD degree in computer science and engineering from the Huazhong University of Science and Technology, China, in 2010. He is currently a senior lecturer with Swinburne. His research interests include service computing, software engineering, cloud computing and edge computing. He is a senior member of the IEEE. For more information, please visit https://sites.google.com/site/heqiang/.

Xin Luo (M'14–SM'17) received the BS degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2005, and the PhD 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 more than 100 papers (including more than 30 *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 Post-Doctoral Science Foundation, in 2014, the Pioneer Hundred Talents Program of 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 reward of *IEEE ACCESS*, in 2018. He has also served as the Program Committee Member for more than 20 international conferences. He is a senior member of the IEEE.

Mingsheng Shang received the PhD 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 and their applications.

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

Guoyin Wang (M'98–SM'03) received the BE degree in computer software, in 1992, the MS degree in computer software, in 1994, and the PhD degree in computer organization and architecture, in 1996, from Xi'an Jiaotong University, Xi'an, China. 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 Postsand Telecommunications, where he is currently a professor and PhD supervisor, the 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 the IEEE.