Towards Efficient and Privacy-Preserving Service QoS Prediction with Federated Learning

Yilei Zhang[1,2(✉)], Xiao Zhang[1], and Xinyuan Li[1]

[1] Anhui Normal University, Wuhu, China
stonezyl@gmail.com, {zx95,bread}@mail.ahnu.edu.cn
[2] The Chinese University of Hong Kong, Hong Kong, China

Abstract. Cloud computing provides many service resources that enable large-scale cloud applications composed of services to be widely adopted in many crucial domains. Quality of Service (QoS) is often used as an indicator in service selection and composition to guarantee the quality of cloud applications. To facilitate QoS-based selection and composition, previous studies have employed collaborative filtering techniques to predict unknown QoS values as a supplement to limited user-perceived QoS data. However, Collaborative modeling approaches encounter privacy issues in the practice of QoS prediction. Users may be reluctant to collaborate through sharing data. As a result, addressing privacy threats has become a key effort towards making QoS prediction methods practical. In this paper, we leverage federated learning techniques and propose a privacy-preserving QoS prediction approach to address this challenge. We further propose several efficiency improvement techniques to significantly reduce system overhead so that the prediction model can provide results quickly and timely. We conduct experiments on a large-scale real-world QoS dataset to evaluate our approach, and the experimental results show that it can make fast and accurate predictions.

Keywords: QoS prediction · Cloud services · Privacy preservation · Federated learning · Communication efficiency

1 Introduction

Cloud computing provides many service resources that enable large-scale cloud applications, which provide services to millions of customers around the world on a 24/7 basis. With the rapid growth of service market, a large number of services are publicly accessible in cloud to provide various atomic functions, which makes Service-Oriented Architecture (SOA) has become the primary paradigm for constructing cloud applications. Desired services are discovered and composed by users (i.e. cloud applications) in a loosely-coupled way at both design time and runtime. Ensuring the performance of cloud applications especially those used in key areas such as government, health care, and finance, is a big challenge as

H. Gao et al. (Eds.): CollaborateCom 2020, LNICST 350, pp. 37–57, 2021.
https://doi.org/10.1007/978-3-030-67540-0_3

it is difficult to get details of the internal implementation of services provided by third parties.

In order to ensure the performance of applications, it is necessary to select high-quality services. Generally, effective service selection employs QoS as a primary indicator to distinguish a set of candidate services with equivalent or similar functions. QoS is widely used to describe the non-functional characteristics of services, such as response time, throughput, Availability, etc. The QoS values of services hosted by different providers may vary depending on the server side environment (e.g., workload, bandwidth, etc.). In addition, due to user-side factors (e.g., geographical location, device capabilities, etc.) and the network environment (e.g., bandwidth, congestion, etc.) between users and services, the observed QoS for each user is different from the others, even on the same service. Therefore, effective service selection is highly dependent on the user-perceived QoS values of services. However, it is impossible to get QoS values for all candidate services because each user invokes a very limited number of services and can only observe the QoS values for the corresponding service. It is also impractical to actively evaluate these QoS values due to the high cost of invoking a large number of services.

In recent literature, collaborative filtering is commonly employed by QoS prediction approaches [25,30,33] to solve the QoS prediction problem. A global QoS dataset is formed by collecting user-contributed local service usage records. By building a prediction model on the dataset, unknown QoS values can be estimated rather than measured by actual invocations. Unfortunately, the practical use of these approaches is severely hampered by privacy threat issues. In order to receive the predicted QoS values from the prediction model, users need to provide their local usage data, which may disclose their sensitive data and privacy. It is difficult for users to prevent others from inferring personal information or selling their data. Consequently, some users are concerned that their valuable data may be leaked and are not enthusiastic in participating in collaborative modeling. As a result, the limited historical training data leads to the low prediction accuracy of the model.

In order to allow more users to safely participate in the collaborative model construction, there is an urgent need to implement privacy-preserving mechanisms in the QoS prediction approach.

Different from the conventional collaborative filtering approaches which use the original data to build a prediction model, several privacy-preserving approaches leveraging encryption [12], obfuscation [35], anonymization [19], and other data conversion techniques for QoS prediction. While these methods can provide a certain degree of privacy protection, they still face the following limitations:

- **Overhead:** All users need to transfer local usage data to the central server. Large-scale online applications (i.e., users) typically run around the clock and support millions of customers, generating an incredible amount of usage data. Transferring large amounts of data over the network imposes a huge overhead on users and the central server.

- **Accuracy:** The use of obfuscated data techniques can lead to the loss of some useful information on the original QoS data, which will affect the accuracy of the model. It is difficult to determine the level of obfuscation to balance the trade-off between privacy preservation and prediction accuracy.
- **Efficiency:** Due to the large amount of data needed to train the predictive model, it puts forward a extremely high demand on the processing capacity of the server. It is a severe challenge for the central server to provide prediction results to all users in a timely manner at runtime. In addition, data obfuscation, data encryption and data decryption lead to extra computation time, which further delays the delivery of prediction results.

This paper proposes a highly efficient QoS prediction approach using federated learning techniques [26] to protect user privacy. Users and the server work together in a distributed and collaborative manner to train a global QoS prediction model. In this paradigm, users do not have to send a large amount of local data, but only a small number of parameters of the local model to the central server, thus achieving the goal of protecting user privacy. In order to improve the efficiency of the federated prediction model, we propose several techniques, including reducing the computational overhead, reducing the size of the transmission message, and reducing the number of communication rounds. Compared with the existing privacy-preserving QoS prediction approaches, our approach has several significant advantages: 1) Compared with QoS data, the data volume of model parameters is very small, which greatly reduces the transmission overhead. 2) The model is trained over real QoS data rather than obfuscated data, which ensures the accuracy of the model while protecting the privacy of users. 3) Since a lot of training and prediction work is distributed on the user side in the decentralized prediction framework, the workload of central server is cut down and the computing capacity on the server side is no longer a bottleneck. The proposed efficiency improvement techniques further reduce the processing time on the user side. We evaluate our approach on a real-world QoS dataset which has been widely employed for evaluating QoS prediction approaches in the literature. The experimental results show that our approach can still achieve high accuracy while preserving the privacy of users, and is more efficient than the existing approaches.

In summary, this paper makes the following contributions:

- We propose a novel decentralized framework for privacy-preserving QoS prediction by employing federated learning techniques.
- We further propose several effective techniques to significantly improve the efficiency of federated QoS prediction approach.
- We conduct extensive experiments on a large-scale real-world QoS dataset to evaluate the effectiveness and efficiency of our privacy-preserving prediction approach.

The rest of the paper is organized as follows. We describe the detailed privacy-preserving QoS prediction approach in Sect. 2. We conduct experiments and discuss the evaluation results in Sect. 3. We present some related works in Sect. 4. Finally, we conclude this paper in Sect. 5.

2 Efficient Privacy-Preserving QoS Prediction

In this section, we propose a novel privacy-preserving federated QoS prediction approach. We first introduce the QoS prediction problem of cloud services in Sect. 2.1. Then we describe the conventional matrix factorization and discuss its limitations on privacy-preserving issues in Sect. 2.2. The details of proposed privacy-preserving approach are described in Sect. 2.3. Finally, three techniques are proposed to enable efficient privacy-preserving QoS prediction in Sect. 2.4.

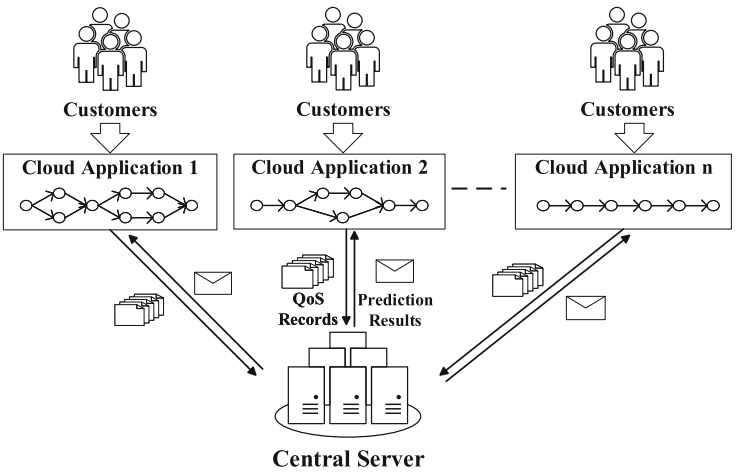


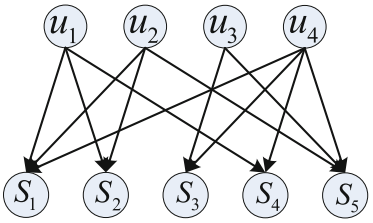
Fig. 1. QoS prediction problem

2.1 Problem Description

Figure 1 illustrates the QoS prediction problem we study in this paper. Many applications are deployed in the cloud to serve their customers on a 24/7 basis. In the workflow of an application, multiple services need to be invoked at runtime. By integrating response results in the workflow, applications can implement specific logic. Note that in this paper, we use users to refer to applications. Generally, a user can invoke a set of services, and a service can be invoked by different users. Since there exists many functionally equivalent or similar services, users can select services with better QoS performance. Each user records the QoS values observed for each service invocation, representing the personalized service performance experienced by that user at that time. By collecting data from all users with collaborative sharing mechanism [32], a global QoS matrix can be formed. In the matrix, each row represents a user, each column represents a service, and each entry represents the QoS value of the corresponding service observed by a user. An empty entry indicates that the corresponding user did not invoke the service before. Typically, a user invokes only a small number

of services, while a large number of services are not invoked. Therefore, most of the entries in the QoS matrix are unknown. This paper investigates how to predict unknown QoS values more efficiently and accurately through data sharing mechanisms while protecting user privacy.

Formally, suppose there are m users and n services. $Q \in \mathbb{R}^{m \times n}$ represents the QoS matrix, with each entry q_{ij} representing the QoS value of service s_j observed by user u_i. $I_{ij} = 1$ indicates an observed entry, and $I_{ij} = 0$ indicates an empty entry. Given all the known entries $\{q_{ij}|I_{ij} = 1\}$, the empty entries $\{q_{ij}|I_{ij} = 0\}$ should be predicted representing unknown QoS values.



(a) Service Invocations

	S_1	S_2	S_3	S_4	S_5
u_1	0. 98	0. 23	?	0. 22	?
u_2	0. 13	0. 26	?	?	0. 43
u_3	?	?	0. 38	?	0. 12
u_4	0. 69	?	0. 22	0. 22	0. 34

(b) Observed QoS

	S_1	S_2	S_3	S_4	S_5
u_1	0. 98	0. 23	0. 24	0. 22	0. 53
u_2	0. 13	0. 26	0. 27	0. 25	0. 43
u_3	0. 85	0. 37	0. 38	0. 36	0. 12
u_4	0. 69	0. 23	0. 22	0. 22	0. 34

(c) Predicted QoS

Fig. 2. Matrix factorization

2.2 Conventional Matrix Factorization

In collaborative filtering [15], Matrix factorization [17] is widely used to build recommender systems. In cloud computing, some existing approaches [31] use it to collect data and build prediction models for QoS prediction. Matrix factorization derives user feature matrix and service feature matrix from the original QoS matrix. These two matrices are low-rank, high-quality, which represent latent factors of users and services, respectively. The premise behind a low-rank factor model is that the entries of Q have a large correlation in practice, therefore a small number of latent factors can represent the features more effectively. The degree to which user latent vector match the corresponding service latent vector determines what QoS value the user can observe. Figure 2 illustrates a toy example of matrix factorization in QoS prediction problem. As shown in Fig. 2(a), the

invocation from a user to a service is represented by a solid line. The user records the QoS value it observed in the QoS matrix in the corresponding entry of this invocation as shown in Fig. 2(b). By decomposing the matrix, users and services are mapped to a low-dimensional joint latent factor space, so that the existing entries in the QoS matrix can be recaptured as the inner product of the latent vectors in the space. Finally, the unknown entries in the matrix can be predicted in a similar way using latent vectors as shown in Fig. 2(c).

We use $U \in \mathbb{R}^{l \times m}$ to denote the user latent matrix and each row represents the latent vector of a particular user. $S \in \mathbb{R}^{l \times n}$ denotes the service latent matrix and each row represents the latent vector of a particular service. l is the rank of the QoS matrix. Matrix factorization attempts to find the optimal U and S, so that Q can be well fitted by the inner product of U and S (i.e., $Q \approx U^T S$). It can be formalized to minimize the following loss function:

$$\mathcal{L} = \frac{1}{2} \sum_i \sum_j I_{ij} (q_{ij} - U_i^T S_j)^2 + \frac{\lambda}{2} (||U||^2 + ||S||^2), \tag{1}$$

where $I_{ij} = 1$ means q_{ij} is observed, and $I_{ij} = 0$ means q_{ij} is unknown. $|| \cdot ||$ is the Euclidean norm. The first term calculates the sum of squared errors between observed QoS values and the predicted ones by $U^T S$. We add two regularization terms in the last positions to avoid overfitting. λ controls the extent of regularization. The optimal U and S is calculated by an iterative process of gradient decent until convergence:

$$U_i \leftarrow U_i - \eta \frac{\partial \mathcal{L}}{\partial U_i}, \tag{2}$$

$$S_j \leftarrow S_j - \eta \frac{\partial \mathcal{L}}{\partial S_j}, \tag{3}$$

where η is the learning rate that controls the step size at each iteration. By calculate the inner product of user latent vectors and service latent vectors, the unknown QoS values can be predicted as follows:

$$p_{ij} = U_i^T S_j. \tag{4}$$

2.3 Privacy-Preserving Matrix Factorization

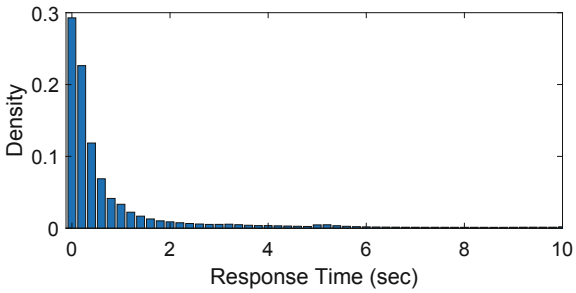
Despite the high prediction accuracy achieved by conventional matrix factorization models, it is difficult to apply them directly in practice. Users are concerned that their privacy will be compromised when providing data to third parties. Consequently, collaborative filtering approaches are difficult to work effectively. In this paper, we propose a privacy-preserving QoS prediction approach to address this challenge.

Data Transformation

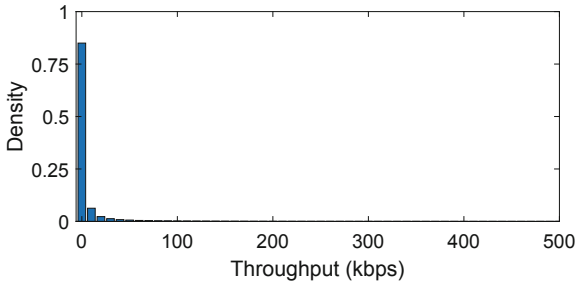
Figure 3 illustrates the distributions of response time and throughput values in the real-world QoS dataset. Obviously, these two distributions have high skewness and large variance which is conflict with the probabilistic hypothesis of matrix factorization [17]. Moreover, different QoS attributes also have different value ranges (e.g., 0–7000 kbps for throughput, 0–100% for availability). This will affect the prediction accuracy of matrix factorization based approaches. To handle this issue, we conduct data transformation on the raw QoS data by employing Box-Cox transformation [18] before building prediction model. Box-Cox is a useful data transformation technique widely used to stabilize variance in data analysis. A Box-Cox transformation is a way to transform non-normal dependent variables into a normal shape as follows:

$$b(x) = \begin{cases} \dfrac{x^{\alpha} - 1}{\alpha}, & \text{if } \alpha \neq 0, \\ log(x), & \text{if } \alpha = 0, \end{cases} \tag{5}$$

where α controls the extent of the transformation. Let q_{max} and q_{min} be the upper and lower bounds of the QoS value respectively, which can be specified by the user. Since $b(x)$ is a monotonic non-decreasing function, $b(q_{max})$ and $b(q_{min})$ are the upper and lower bounds after data transformation. Then the QoS values can be mapped to the range $[0, 1]$ by the following transformation:



(a) Response Time



(b) Throughput

Fig. 3. QoS value distributions

$$\hat{q}_{ij} = \frac{b(q_{ij}) - b(q_{min})}{b(q_{max}) - b(q_{min})}. \tag{6}$$

Accordingly, the prediction result p_{ij} can be mapped to the range $[0, 1]$ by the logistic function, as simply done in [17]:

$$\hat{p}_{ij} = \frac{1}{1 + e^{-p_{ij}}}. \tag{7}$$

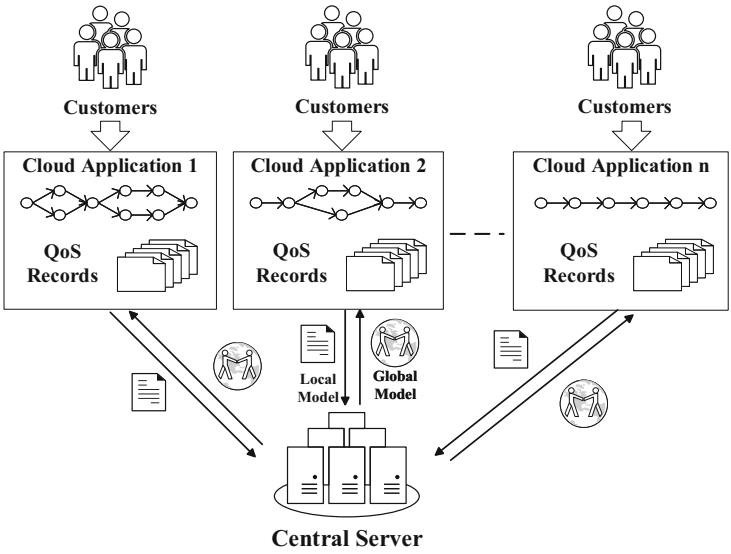


Fig. 4. Federated QoS prediction framework

Federated QoS Prediction

Figure 4 illustrates the federated QoS prediction framework. Users record the observed QoS data locally while the application is running, and this data does not need to be transmitted to the central server. Part of the matrix factorization process takes place locally at the user site and is integrated again at the central server through federated learning technologies. More specifically, user latent factors are learned on the user side and the service latent matrix is learned on the server side. The prediction model is continuously trained and updated in multi-rounds (i.e., time slices). In each round, the user receives the latest global service latent matrix from the central server. Each user then updates the local QoS matrix with the latest observed QoS values. The local matrix factorization process is then performed based on the global service latent matrix to obtain the local user latent vector and the updates to the global service matrix latent matrix. After that, each user simply sends the updated global service latent

matrix to the central server and keeps the raw QoS data and user latent vector in local site. Then, the central server collects the updated from all users and combines them to determine the updates that should be made to the global service latent matrix in this round, and provides the updated global service latent matrix to the users in the next round. Each user can quickly predict unknown QoS values using local user latent vector and global service latent matrix.

More specifically, at the beginning of each round, the local QoS matrix $Q_i \in \mathbb{R}^{1\times m}$ is updated with newly observed QoS data. Each user u_i get a local copy $S^i \in \mathbb{R}^{l\times n}$ of the global service latent matrix S from the central server. The local user latent vector $U_i \in \mathbb{R}^{l\times 1}$ are initialized with small random numbers. The local loss function is as follows:

$$\mathcal{L}_i = \frac{1}{2}\sum_j I_{ij}(\hat{q}_{ij} - \hat{p}_{ij})^2 + \frac{\lambda}{2}(||U_i||^2 + ||S^i||^2), \tag{8}$$

and the global loss function is:

$$\mathcal{L} = \sum_i \mathcal{L}_i. \tag{9}$$

Instead of minimizing $\mathcal{L}$, the federated model relaxes to minimize the local loss function $\mathcal{L}_i$ at each user site. U_i and S^i are updated on local QoS matrix Q_i by the iterative process until converge:

$$U_i \leftarrow U_i - \eta((\hat{q}_{ij} - \hat{p}_{ij})\hat{p}'_{ij}S^i_j + \lambda U_i), \tag{10}$$

$$S^i_j \leftarrow S^i_j - \eta((\hat{q}_{ij} - \hat{p}_{ij})\hat{p}'_{ij}U_i + \lambda S^i_j), \tag{11}$$

where $\hat{p}'_{ij} \leftarrow \dfrac{e^{p_{ij}}}{(e^{p_{ij}}+1)^2}$ is the derivative of $\hat{p}_{ij}$ and η is the learning rate. The user keeps U_i locally and send updated S^i to the central server. The central server then takes the average of updated S^i from all users and updates the global service latent matrix as follows:

$$S = \frac{1}{m}\sum_i S^i. \tag{12}$$

For each unobserved q_{ij}, the corresponding user u_i can conduct a quick local prediction as follows:

$$p_{ij} = U_i^T S^i_j. \tag{13}$$

Algorithm 1 shows the detailed process of federated QoS prediction. In each round, the local QoS data is stored and processed on the corresponding user side without exposing to others, thus protect the user from privacy threat.

Differential Privacy. In order to prevent the central server from inferring user privacy from gradients, we adopt differential privacy technology [7] to encrypt and decrypt gradient parameters transmitted between the user and the server.

Algorithm 1: Federated QoS Prediction

Input: QoS matrix Q, the current model: U, S, and the parameters η, λ.
Output: The updated model: U, S.

repeat
  foreach *user* u_i **do**
    $S^i \leftarrow S$; `/* Local copy of global S */`
    Update local QoS matrix Q_i with new QoS values;
    Initialize U_i with small random numbers;
    repeat `/* Update local U_i and S^i */`
      $p_{ij} \leftarrow U_i^T S_j^i$;
      $\hat{p}_{ij} \leftarrow \frac{1}{1+e^{-p_{ij}}}$;
      $\hat{p}'_{ij} \leftarrow \frac{e^{p_{ij}}}{(e^{p_{ij}}+1)^2}$;
      $U_i \leftarrow U_i - \eta((\hat{q}_{ij} - \hat{p}_{ij})\hat{p}'_{ij} S_j^i + \lambda U_i)$;
      $S_j^i \leftarrow S_j^i - \eta((\hat{q}_{ij} - \hat{p}_{ij})\hat{p}'_{ij} U_i + \lambda S_j^i)$;
    until *converge*;
  end
  $S \leftarrow \frac{1}{m}\sum_i S^i$; `/* Update global S */`
until *forever*;

For simplicity, we denote the gradient by g hereafter. A secret key comprising of two big prime numbers is generated and distributed to each user. Let x and y be the two prime numbers, $N = xy$ be the public parameter, and ϵ be the privacy budget. For each user u_i, the randomly sampled noise $Lap(\frac{\Delta f}{\epsilon})$ from Laplace distribution is add to the local gradients as follows:

$$g_{i,x} \equiv (g_i + Lap(\frac{\Delta f}{\epsilon})) \mod x, \tag{14}$$

$$g_{i,y} \equiv (g_i + Lap(\frac{\Delta f}{\epsilon})) \mod y, \tag{15}$$

where Δf denotes the sensitivity, and ϵ controls privacy budget. A smaller ϵ indicates a higher privacy level. The encrypted gradient is then calculated by:

$$E_i = y^{-1} y g_{i,x}^x + x^{-1} x g_{i,y}^y \mod N, \tag{16}$$

where y^{-1} and x^{-1} are the inverses of y and x, respectively. After receiving E_i from each user u_i, the central server performs global aggregation in the following way:

$$E = \sum_i E_i. \tag{17}$$

At the beginning of each round, each user u_i receives a local copy of the latest encrypted global gradients E and decrypts it by:

$$g \equiv E \mod x, \tag{18}$$

$$g \equiv E \mod y, \tag{19}$$

such that $g \approx \sum_i g_i$ is an unbiased estimate of the global gradient.

2.4 Efficient Federated Learning

While federated QoS prediction approach can protect user privacy, further improvements are needed to accommodate the diversity capacities of user devices. As cloud applications usually run on customers' devices, such as desktops, laptops, tablets, smart phones, and in-car systems, computing capacity and network bandwidth may become bottlenecks. In order to solve this problem, we propose several techniques to improve the model efficiency from three aspects: reducing computational overhead, reducing the amount of data transmission, and reducing the number of communications.

Reducing the Overhead of User-Side Computation in Each Round As shown in Algorithm 1, each user needs to learn the potential user factor u_i and update the potential service matrix S during several rounds of iteration. The iterative process takes time from random initialization to convergence. Since the user latent vectors do not change much between two consecutive rounds, we initialized each round with the u_i from the previous round instead of random values to reduce learning time. Therefore, we only need to calculate the incremental updates in each round. For this purpose, we employ stochastic gradient descent (SGD) [16], a wildly-used online learning technique, to train the model. For each newly observed QoS value q_{ij} from user u_i in the current round, we have the pointwise loss function:

$$\Delta\mathcal{L}_{ij} = \frac{1}{2}(\hat{q}_{ij} - \hat{p}_{ij})^2 + \frac{\lambda}{2}(||U_i||^2 + ||S_j^i||^2), \tag{20}$$

such that $\mathcal{L}_i = \sum_j \Delta\mathcal{L}_{ij}$. Instead of minimizing $\mathcal{L}_i$, SGD relaxes to minimize the pointwise loss function $\Delta\mathcal{L}_{ij}$ over all newly observed QoS values. The corresponding updating equations are as follows:

$$U_i \leftarrow U_i - \eta((\hat{q}_{ij} - \hat{p}_{ij})\hat{p}'_{ij}S_j^i + \lambda U_i), \tag{21}$$

$$S_j^i \leftarrow S_j^i - \eta((\hat{q}_{ij} - \hat{p}_{ij})\hat{p}'_{ij}U_i + \lambda S_j^i), \tag{22}$$

where $\hat{p}'_{ij} = \dfrac{e^{p_{ij}}}{(e^{p_{ij}}+1)^2}$ is the derivative of $\hat{p}_{ij}$ and η is the learning rate. The user then keeps U_i locally and only need to transfer the gradient descent $\frac{\partial\mathcal{L}_i}{\partial S^i}$ instead of S^i to the central server. The central server collects contributions from all users and takes advantage of the learning results to make a simple average to get a global gradient descent:

$$\frac{\partial\mathcal{L}}{\partial S_j} = \frac{1}{m}\sum_i \frac{\partial\mathcal{L}_i}{\partial S_j^i}. \tag{23}$$

Then the global service latent matrix can be updated as follows:

$$S_j \leftarrow S_j - \eta \frac{\partial \mathcal{L}}{\partial S_j}. \tag{24}$$

Reducing the Size of Transmitted Messages in Each Round
In practice, federated QoS prediction involves a massive number of users with various devices (e.g., desktops, laptops, smart phones, etc.), and communication on the Internet is many orders of magnitude slower than local computing [10]. The asymmetry of Internet speed (i.e., the uplink is usually much slower than the downlink) makes communication possible to become a bottleneck in the construction of a federated prediction model. It is necessary to reduce the communication cost, especially the uplink communication cost. We adopt three optimization measures to reduce the amount of data transmitted.

- Firstly, according to Eq. (22), only when user u_i observes an updated QoS value q_{ij} in the current round, will the corresponding service latent factor S_j^i be updated. Therefore, we can submit a subset of $\frac{\partial \mathcal{L}_i}{\partial S^i}$ (i.e., the set of updated $\frac{\partial \mathcal{L}_i}{\partial S_j^i}$) instead of the entire $\frac{\partial \mathcal{L}_i}{\partial S^i}$ to the central server without affecting accuracy. Accordingly, the central server aggregates the local updates by:

$$\frac{\partial \mathcal{L}}{\partial S_j} = \frac{1}{|N_j|} \sum_{u_i \in N_j} \frac{\partial \mathcal{L}_i}{\partial S_j^i}, \tag{25}$$

  where N_j is the set of users update S_j^i in the current round.
- Secondly, instead of sending $\frac{\partial \mathcal{L}_i}{\partial S_j^i}$, each user only communicates $\frac{\partial \tilde{\mathcal{L}}_i}{\partial S_j^i}$ which is formed from a random subset of the values of $\frac{\partial \mathcal{L}_i}{\partial S_j^i}$. the central server aggregates the local updates to form a global update $\frac{\partial \tilde{\mathcal{L}}}{\partial S_j}$, which is an unbiased estimator of the true update:

$$\frac{\partial \mathcal{L}}{\partial S_j} = \mathbb{E}[\frac{\partial \tilde{\mathcal{L}}}{\partial S_j}]. \tag{26}$$

  The mask can be randomized independently for each user in each round by a synchronized seed stored on both the user side and the server side. The percentage of omitted values is defined by the parameter r.
- Thirdly, we compress the transmitted data by quantizing the updates. Let $d \in \mathbb{R}^{1\times l}$ denotes the vector of $\frac{\partial \tilde{\mathcal{L}}_i}{\partial S_j^i}$, $d_{max} = max(d_k)$ and $d_{min} = min(d_k)$. The quantized d is calculated as follows:

$$\bar{d}_k = \begin{cases} d', & \text{with probability } \dfrac{d - d_{min}}{d_{max} - d_{min}} \\ d'', & \text{with probability } \dfrac{d_{max} - d}{d_{max} - d_{min}} \end{cases}, \tag{27}$$

where $d' = \lceil \frac{d - d_{min}}{d_{max} - d_{min}} 2^b \rceil$ and $d'' = \lfloor \frac{d - d_{min}}{d_{max} - d_{min}} 2^b \rfloor$. It is easy to show that $\bar{d}$ is an unbiased estimator of d. Parameter b represents b-bit quantization, which controls the trade-off between accuracy and communication overhead. The error incurred by this compression was analyzed in [22].

Reducing the Number of Communication Rounds
Local service latent factors usually do not change much in one round. In order to reduce the number of communications for each user, a threshold δ can be set to control how often updates are transmitted to the central server. If the maximum value of the updates is greater than δ, the updates are transmitted. Otherwise, the updates are not transmitted and saved at the local site. Undelivered updates will be accumulated in the next round. In this way, large updates have influence on the model in time. The delay of small updates has little effect on the accuracy of the model, and small updates eventually produce an effect on the model in the form of cumulative updates. δ controls the tradeoff between efficiency and accuracy.

3 Experiments

In this section, we conduct several experiments on a real-world QoS dataset of Web services to evaluate our online prediction approach.

3.1 Dataset Description

We conduct experiments on a commonly used real-world dataset [31] to evaluate the QoS prediction approaches. This dataset contains two important QoS attributes, response time (RT) and throughput (TP), which characterize the non-functional quality of services. RT refers to the time it takes for a service to respond to a user request. TP refers to the data transfer rate during an invocation. Without losing generality, our approach can be easily extended to other QoS attributes.

Table 1. Statistics of QoS dataset

QoS	#Users	#Services	#Slices	#Records	Scale	Median
RT (*sec*)	142	4,532	64	27,393,508	0–20	0.377
TP (*kbps*)	142	4,532	64	25,643,690	0–1000	1.851

The QoS values are collected by invoking 4,532 services from 142 geographically distributed users on 64 consecutive time slices, each for 15 min. The users and the real-world services are distributed in 22 countries and 57 countries, respectively. The statistics of the dataset are summarized in Table 1. The range

of response time is 0–20 s, and the range of throughput is 0–1000 kbps. The medians of response time and throughput are 0.377 s and 1.851 kbps, respectively. Note that there are some missing values in the dataset. Figure 3 shows the data distributions of response time and throughput, which are highly skewed. Figure 5 shows the data distributions closer to normal shape after data transformation by tuning α to -0.007 for RT and -0.005 for TP.

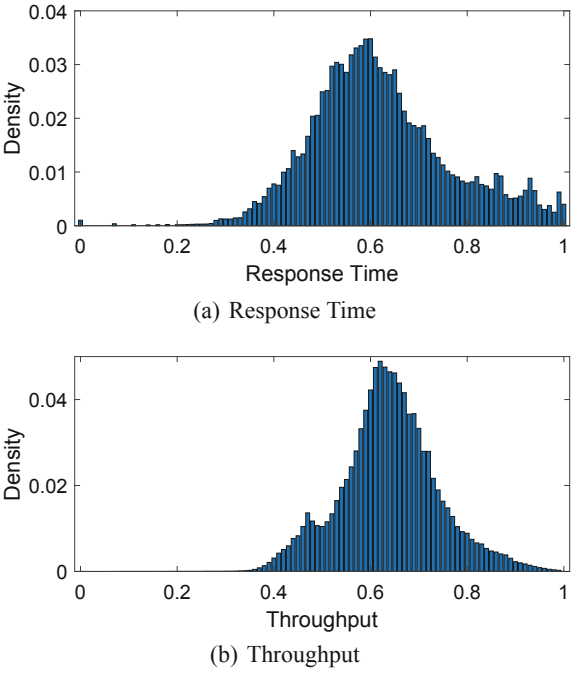


Fig. 5. Transformed data distributions

3.2 Evaluation Metric

We use MAE (Mean Absolute Error) to evaluate the accuracy of QoS prediction approaches. MAE is defined by:

$$MAE = \frac{1}{N} \sum_{I_{ij}=0} |q_{ij} - p_{ij}|, \tag{28}$$

where q_{ij} represents the real QoS value of an empty entry, p_{ij} is the predicted one by the model, and N represents the total number of predicted entries in the experiments. MAE is a widely used metric for evaluating the accuracy of a prediction model. Generally, the lower the MAE value, the better the performance of the model and the higher the prediction accuracy.

3.3 Prediction Accuracy

We compare the proposed privacy-preserving QoS prediction approach against other counterpart approaches in terms of accuracy. Note that some of these prediction approaches have no mechanism to protect privacy. The representative approaches in the literature are as follows:

- UIPCC [32]: This approach use historical QoS data of similar users and similar services to predict the QoS for current users. It is a collaborative filtering approach without any privacy-preserving mechanism.
- PMF [33]: This is a collaborative filtering approach employing matrix factorization techniques for QoS prediction, with no privacy-preserving mechanism.
- P-PMF [35]: This approach is similar to PMF, with the difference that it integrates a privacy-preserving mechanism through the use of data obfuscation techniques.
- **FMF:** This is the federated QoS prediction approach proposed in Sect. 2.3 with a privacy-preserving mechanism.
- **EFMF:** This approach further extends the FMF approach with efficiency improvement techniques proposed in Sect. 2.4.

We randomly designate some entries as unobserved ones and remove them from the QoS matrix. We vary the percentage of removed entries from 10% to 30% with a step of 5%. $l = 10$, $r = 20\%$, $b = 8$, and δ is tuned to optimal value in experiments for EFMF. We also optimize the parameters for counterpart approaches in experiments accordingly. We carry out the experiments 20 times with randomized initialization for each density and take the average accuracy as the final result. Table 2 shows the prediction accuracy. The results show that FMF and EFMF have higher accuracy than UIPCC and P-PMF. The predictive accuracy of FMF and EFMF is similar to that of PMF, whereas PMF does not consider privacy issues. Compared to approaches based on similar users and services, matrix factorization can retrieve more useful information in sparse data. Therefore, FMF and EFMF have a higher accuracy than UIPCC. P-PMF approach uses confused data, which will bring noise to the model. FMF and EFMFF use raw QoS data and are therefore more accurate than P-PMF. Generally, all approaches are more accurate as the data becomes more dense, since more training data means more information. The experimental results shows that the proposed privacy-preserving QoS prediction approach is effect.

Table 2. Prediction accuracy (w.r.t. MAE)

QoS	Approach	Data density				
		10%	15%	20%	25%	30%
RT	UIPCC	0.593	0.527	0.489	0.461	0.445
	PMF	0.506	0.485	0.468	0.451	0.439
	P-PMF	0.557	0.524	0.501	0.482	0.469
	FMF	0.517	0.492	0.475	0.458	0.446
	EFMF	0.522	0.500	0.481	0.462	0.454
TP	UIPCC	23.457	21.253	19.964	18.582	17.738
	PMF	16.580	15.307	14.681	14.178	13.957
	P-PMF	21.267	18.742	17.502	16.930	16.635
	FMF	17.270	16.028	15.145	14.354	14.067
	EFMF	17.714	16.351	15.296	14.397	14.105

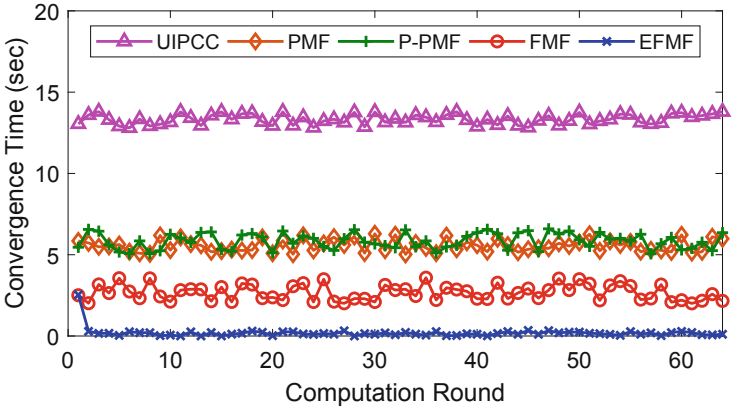
3.4 Prediction Efficiency

We compare the convergence time of different approaches to evaluate the efficiency. Figure 6 shows that FMF and EFMF need less time to than other approaches to make prediction. Since the federated QoS prediction model is trained in a distributed manner, it includes both local models and a global model. Users process small amounts of data locally to update the local model. The central server, on the other hand, can update the global model by simply handling the updates collected for users. In contrast, other approaches require the central server to process large amounts of data and train the model over long periods of time. EFMF is more efficient than FMF due to the several techniques we used in Sect. 2.4. In addition, the convergence time EFMF in the first round is relatively long, and it is very short in the subsequent rounds. This is because EFMF only needs to incrementally update the model by using online learning techniques for new observations. These results demonstrate the efficiency of our approaches.

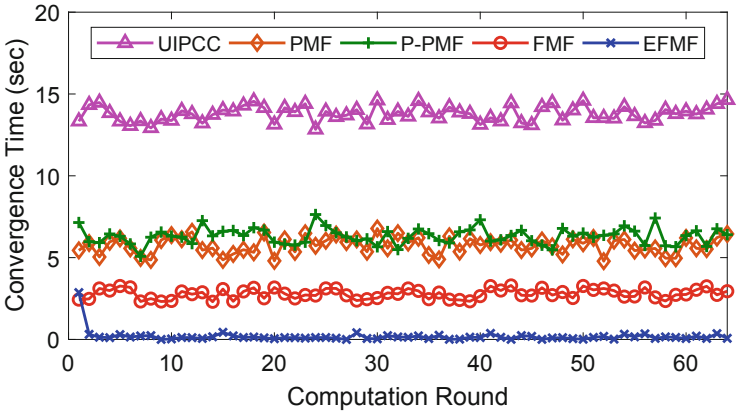
3.5 Scalability Analysis

We further analyze the scalability of our approach. We compare the median convergence time under different data volumes by varying the number of users and services. Due to the limited size of the real-world dataset, we generate large-scale synthetic datasets by repeating the data, which is reasonable for measuring convergence time. Figure 7 shows that the convergence time (log scale) of all approaches increases with the number of users and services. The increase of convergence time of FMF and EFMF is very gentle, while the convergence time of other approaches increases greatly. This is because the computing tasks are distributed to users in FMF and EFMF. In addition, EFMF employ online prediction techniques and make incremental updates on new data, while other

approaches requires an iterative process in each time slice. These results demonstrate that EFMF can be easily extend to process large-scale data from more users and services.



(a) Response time



(b) Throughput

Fig. 6. Prediction efficiency

4 Related Work

4.1 QoS Prediction

QoS refers to a series of non-functional characteristics of services (e.g., response time, throughput, availability, etc.) [5]. It has been widely employed to facilitate QoS-driven approaches in cloud and service computing [27]. QoS-aware service recommendation and selection help users select high-quality services from a set

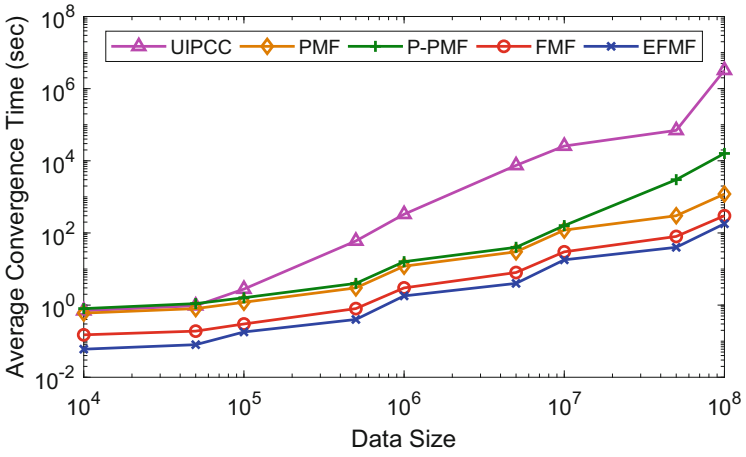


Fig. 7. Scalability result

of functional equivalent or similar service candidates. QoS-aware composition [11,24] aims to orchestrating a set of available services to provide complicated functions through a defined work flow. QoS-aware service adaption [6] help users replace current working services with corresponding service candidates at runtime to tolerate unacceptable QoS degradation or failures [36].

Accurate QoS value is important for QoS-aware techniques. Online testing [13,34] is proposed to obtain QoS values by active measurement. However, this approach incurs additional overhead and the observed QoS data is very sparse. In order to solve this problem, QoS prediction approaches [14,23,29–31] are proposed to use historical observed QoS data and make prediction for unobserved QoS data.

Collaborative filtering techniques [9,21] are widely used in social computing to predict unknown ratings on items (e.g., books, movies, etc.). They are introduced recently make QoS prediction for services. Matrix factorization is a representative model in collaborative filtering. The model derives user latent factors and service factors to represent the user-specific features and service-specific features, respectively. The unobserved QoS values are predicted by how the user-specific features apply to the corresponding service-specific features. In the literature, some of the state-of-the-art QoS prediction approaches such as PMF [17], CloudPred [30], WSPred [31], AMF [36], P-PMF [35], use matrix factorization techniques to provide accurate QoS prediction results. In contrast, this paper tries to protect user privacy while making QoS prediction.

4.2 Privacy Preservation

In the era of big data, privacy threat issues are particularly concerned by many studies. In cloud computing, many third parties provide services to enable service composition. Personal sensitive data usually flows across different parties.

Untrusted parties may expose these data and pose a threat to user privacy. To address this challenge, there appears some privacy-preserving approaches in service computing, such as privacy-preserving service selection [20,28], service recommendation [1,35] and service composition [2,4], etc. For example, Zhu et al. [35] propose a data obfuscation approach before sharing data across parties to prevent data leakage. Barhamgi et al. [2] analyze the privacy issue in composition of data services. Different from these studies, we focus on privacy-preserving issues for QoS prediction.

4.3 Federated Learning

Federated learning techniques have been used by major service providers [3] for train models over remote devices. In federated learning, a star network, where a central server is connected to a network of devices, is the main communication topology [26]. The decentralized training has been demonstrated to be effective when operating on networks with low bandwidth or high latency [8]. Under federated learning paradigm, users keep their data at local site without transferring it to the central server. In this paper, we conduct matrix factorization in a distributed manner by leveraging federated learning. Further more, we reduce computing cost and improve communication efficiency to make federated matrix factorization feasible.

5 Conclusion

In order to build and maintain high-quality cloud applications, privacy protection is the key to improve the accuracy of QoS prediction. In this paper, we propose a novel efficient and privacy-preserving QoS prediction approach for cloud services. We employ federated learning techniques to train the QoS prediction model in a decentralized manner. Users can keep their data at local site securely without fear of data leakage. We further make some improvements to significantly improve the efficiency of the QoS prediction model. The experimental results on a large-scale real-world QoS dataset demonstrate that our approach is effective and efficient, while effectively preventing privacy threats.

Acknowledgment. The work described in this paper was supported by the National Natural Science Foundation of China (61802003), and the Anhui Innovation Program for Overseas Students.

References

1. Badsha, S., et al.: Privacy preserving location-aware personalized web service recommendations. IEEE Trans. Serv. Comput. (2018)
2. Barhamgi, M., Perera, C., Yu, C.M., Benslimane, D., Camacho, D., Bonnet, C.: Privacy in data service composition. IEEE Trans. Serv. Comput. (2019)
3. Bonawitz, K., et al.: Towards federated learning at scale: system design. arXiv preprint arXiv:1902.01046 (2019)

4. Carminati, B., Ferrari, E., Tran, N.H.: A privacy-preserving framework for constrained choreographed service composition. In: 2015 IEEE International Conference on Web Services, pp. 297–304. IEEE (2015)
5. Chen, T., Bahsoon, R.: Self-adaptive and online QoS modeling for cloud-based software services. IEEE Trans. Softw. Eng. **43**(5), 453–475 (2016)
6. Chen, X., Wang, H., Ma, Y., Zheng, X., Guo, L.: Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model. Future Gener. Comput. Syst. **105**, 287–296 (2020)
7. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_14
8. He, L., Bian, A., Jaggi, M.: COLA: decentralized linear learning. In: Advances in Neural Information Processing Systems, pp. 4536–4546 (2018)
9. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web, pp. 173–182 (2017)
10. Huang, J., et al.: An in-depth study of LTE: effect of network protocol and application behavior on performance. ACM SIGCOMM Comput. Commun. Rev. **43**(4), 363–374 (2013)
11. Li, J., Fan, G., Zhu, M., Yan, Y.: Pre-joined semantic indexing graph for QoS-aware service composition. In: 2019 IEEE International Conference on Web Services (ICWS), pp. 116–120. IEEE (2019)
12. Liu, A., et al.: Differential private collaborative web services QoS prediction. World Wide Web **22**(6), 2697–2720 (2019)
13. Liu, X., Sheu, R.K., Lo, W.T., Yuan, S.M.: Automatic cloud service testing and bottleneck detection system with scaling recommendation. Concurr. Comput.: Pract. Exp. **32**(1), e5161 (2020)
14. Luo, X., Wu, H., Yuan, H., Zhou, M.: Temporal pattern-aware QoS prediction via biased non-negative latent factorization of tensors. IEEE Trans. Cybern. **50**, 1798–1809 (2019)
15. Ma, H., Yang, H., Lyu, M.R., King, I.: SoRec: social recommendation using probabilistic matrix factorization. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 931–940 (2008)
16. Mandt, S., Hoffman, M.D., Blei, D.M.: Stochastic gradient descent as approximate Bayesian inference. J. Mach. Learn. Res. **18**(1), 4873–4907 (2017)
17. Mnih, A., Salakhutdinov, R.R.: Probabilistic matrix factorization. In: Advances in Neural Information Processing Systems, pp. 1257–1264 (2008)
18. Osborne, J.: Improving your data transformations: applying the box-cox transformation. Pract. Assess. Res. Eval. **15**(1), 12 (2010)
19. Qi, L., Xiang, H., Dou, W., Yang, C., Qin, Y., Zhang, X.: Privacy-preserving distributed service recommendation based on locality-sensitive hashing. In: 2017 IEEE International Conference on Web Services (ICWS), pp. 49–56. IEEE (2017)
20. Squicciarini, A., Carminati, B., Karumanchi, S.: A privacy-preserving approach for web service selection and provisioning. In: 2011 IEEE International Conference on Web Services, pp. 33–40. IEEE (2011)
21. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. In: Advances in Artificial Intelligence 2009 (2009)
22. Suresh, A.T., Yu, F.X., Kumar, S., McMahan, H.B.: Distributed mean estimation with limited communication. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 3329–3337 (2017)

23. Wang, S., Zhao, Y., Huang, L., Xu, J., Hsu, C.H.: QoS prediction for service recommendations in mobile edge computing. J. Parallel Distrib. Comput. **127**, 134–144 (2019)
24. White, G., Palade, A., Clarke, S.: QoS prediction for reliable service composition in IoT. In: Braubach, L., et al. (eds.) ICSOC 2017. LNCS, vol. 10797, pp. 149–160. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91764-1_12
25. Xu, Y., Yin, J., Deng, S., Xiong, N.N., Huang, J.: Context-aware QoS prediction for web service recommendation and selection. Expert Syst. Appl. **53**, 75–86 (2016)
26. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. ACM Trans. Intell. Syst. Technol. (TIST) **10**(2), 1–19 (2019)
27. Zhang, Y., Lyu, M.R.: QoS Prediction in Cloud and Service Computing: Approaches and Applications. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-5278-1
28. Zhang, Y., Zhang, P., Luo, Y., Luo, J.: Efficient and privacy-preserving federated QoS prediction for cloud services. In: IEEE Conference on Web Services (ICWS) (2020)
29. Zhang, Y., Zhang, X., Zhang, P., Luo, J.: Credible and online QoS prediction for services in unreliable cloud environment. In: IEEE Conference on Services Computing (SCC) (2020)
30. Zhang, Y., Zheng, Z., Lyu, M.R.: Exploring latent features for memory-based QoS prediction in cloud computing. In: 2011 IEEE 30th International Symposium on Reliable Distributed Systems, pp. 1–10. IEEE (2011)
31. Zhang, Y., Zheng, Z., Lyu, M.R.: WSPred: a time-aware Personalized QoS Prediction Framework for Web services. In: 2011 IEEE 22nd International Symposium on Software Reliability Engineering, pp. 210–219. IEEE (2011)
32. Zheng, Z., Ma, H., Lyu, M.R., King, I.: WSRec: a collaborative filtering based web service recommender system. In: IEEE International Conference on Web Services, pp. 437–444. IEEE (2009)
33. Zheng, Z., Ma, H., Lyu, M.R., King, I.: Collaborative web service QoS prediction via neighborhood integrated matrix factorization. IEEE Trans. Serv. Comput. **6**(3), 289–299 (2012)
34. Zhong, H., Zhang, L., Khurshid, S.: TestSage: regression test selection for large-scale web service testing. In: 2019 12th IEEE Conference on Software Testing, Validation and Verification (ICST), pp. 430–440. IEEE (2019)
35. Zhu, J., He, P., Zheng, Z., Lyu, M.R.: A privacy-preserving QoS prediction framework for web service recommendation. In: 2015 IEEE International Conference on Web Services, pp. 241–248. IEEE (2015)
36. Zhu, J., He, P., Zheng, Z., Lyu, M.R.: Online QoS prediction for runtime service adaptation via adaptive matrix factorization. IEEE Trans. Parallel Distrib. Syst. **28**(10), 2911–2924 (2017)