NCRL: Neighborhood-Based Collaborative Residual Learning for Adaptive QoS Prediction

Guobing Zou, Shaogang Wu, Shengxiang Hu, Chenhong Cao, Yanglan Gan,
Bofeng Zhang, and Yixin Chen

Abstract—How to accurately predict vacant QoS has become a fundamental issue for service-oriented downstream tasks. However, most QoS prediction approaches based on model learning fail to discriminatively capture the latent feature representations of a user and a service, since they either leverage the shallow neural network such as MLP or take advantage of insufficient location information. Moreover, collaborative relationships of similar neighborhood have not been fully taken into account together with prediction model learning. To address these issues, we propose a novel framework for adaptive QoS prediction named Neighborhood-based Collaborative Residual Learning (NCRL). Location-aware two-tower deep residual network is designed to achieve neural QoS prediction by extracting latent features of users and services, which are fed to generate similar neighborhood for collaborative prediction based on historical QoS invocations. They are integrally combined to perform adaptive QoS prediction. Extensive experiments are conducted based on a large-scale real-world QoS dataset called WS-DREAM with almost 2,000,000 historical QoS invocations. The results indicate that NCRL can remarkably outperform state-of-the-art competing baselines.

Index Terms—Web service, QoS prediction, deep residual learning, collaborative filtering, adaptive QoS prediction

1 Introduction

With the rapid development of Internet technology and 5G cellular network, Web services are ubiquitous in today's real world application scenarios and play a crucial role in the era of big data, Internet of Things (IoT), cloud computing, and edge computing [1]. In the past few years, more and more Web services have been published on the largest online RESTful service repository, ProgrammableWeb,[1] which has registered more than 24,000 Web services as of November 9, 2021. As the essential building components for service discovery, selection, composition and recommendation, Web services prominently accelerate machine-to-machine interactions and promote the advancement of service-oriented software systems.

With the overwhelming explosion of Web services, many of them share the same or similar functionality, making it difficult to select the most suitable services for service requesters. The key factor to distinguish those functionally equivalently services is their provision quality [2]. Thus, Quality of Service (QoS) has been widely applied as a discriminant to represent non-functional service characteristics and recommend desired services with high quality in real-world application scenarios. However, due to the enormous number of users and services, it is impractical and time-consuming for service requesters to invoke all Web services and service providers to monitor QoS information for each service invocation. To satisfy the demands of downstream tasks, it is crucial to accurately perform QoS prediction, which has become a challenging research issue because of the remarkable sparsity of historical user-service invocations.

Collaborative Filtering (CF) as the most important technique has been used to predict the unknown QoS, which has received many attentions and there are a lot of correlative investigations. CF-based QoS prediction can be classified into two categories, including memory-based and model-based approaches [3]. Memory-based approaches perform similarity computation, such as Pearson Correlation Coefficient (PCC) [4] and Ratio-Based Similarity (RBS) [5], to generate a set of similar users or services. They are collaboratively aggregated to predict the unknown QoS in conjunction with the historical QoS invocations [6], [7], [8], [9], [10]. However, these kinds of approaches are highly dependent on common invocations to calculate similar neighborhood and thus are vulnerable to the sparsity of user-service invocations, which may incur low accuracy of QoS prediction. To alleviate the sparsity problem of memory-based approaches, different model learning techniques have been leveraged to represent the features of users and services for unknown QoS prediction, such as clustering

[1] https://www.programmableweb.com/

- *Guobing Zou, Shaogang Wu, Shengxiang Hu, and Chenhong Cao are with the School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China. E-mail: {gbzou, sgwu, shengxianghu, caoch}@shu.edu.cn.*
- *Yanglan Gan is with the School of Computer Science and Technology, Donghua University, Shanghai 201620, China. E-mail: ylgan@dhu.edu.cn.*
- *Bofeng Zhang is with the School of Computer and Information Engineering, Shanghai Polytechnic University, Shanghai 201209, China. E-mail: bfzhang@sspu.edu.cn.*
- *Yixin Chen is with the Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO 63130 USA. E-mail: chen@cse.wustl.edu.*

Manuscript received 1 June 2022; revised 18 September 2022; accepted 4 October 2022. Date of publication 10 October 2022; date of current version 12 June 2023.
This work was supported in part by the National Natural Science Foundation of China under Grants 62272290 and 62172088, and in part by the Shanghai Natural Science Foundation under Grant 21ZR1400400.
(Corresponding authors: Yanglan Gan and Bofeng Zhang.)
Digital Object Identifier no. 10.1109/TSC.2022.3213129

algorithms, matrix factorization and machine learning [11], [12], [13], [14], [15]. These kinds of approaches can learn a QoS prediction model by parameter optimization to mine linear or nonlinear invocation relationships among users and services. Additionally, QoS prediction accuracy can be further improved by introducing users' or services' contextual information, such as geographical location information and service invocation time, into learning paradigms [16], [17], [18]. Especially, more recent investigations focus on deep learning techniques that are applied to extract latent feature representations for unknown QoS prediction [19], [20], [21].

Although these conventional approaches can partially facilitate QoS prediction, they still cannot reach the satisfactory prediction performance for service-oriented application scenarios. The primary reason is that it has become a fundamental but challenging issue on how to accurately learn latent features of users and services, as it has distinct sparsity of user-service invocations. More specifically, most QoS prediction approaches based on model learning fail to discriminatively capture the latent feature representations of a user and a service, since they either leverage shallow neural network such as MLP or take advantage of insufficient location information. Furthermore, collaborative relationships of similar neighborhood has not been fully integrated together with model learning for QoS prediction. Thus, current model-based approaches are incapable of effectively extract the characteristics of user-service invocations from both contextual information and historical QoS records, which significantly reduces the performance of QoS prediction.

To address the above two issues, we propose a novel framework for adaptive QoS prediction named Neighborhood-based Collaborative Residual Learning (NCRL), including three mutually correlative procedures. It first leverages our designed location-aware two-tower deep residual network to separately extract users' and services' latent features and perform neural QoS prediction. Then, the extracted latent feature vectors are used to calculate a user's or service's similar neighborhood to perform collaborative prediction based on historical QoS invocations. Finally, predicting an unknown QoS for a target user who desires to invoke a target service can be achieved by adaptively integrating the neural predicted QoS and collaborative predicted QoS. To demonstrate the effectiveness of our proposed NCRL, extensive experiments are conducted on a public and large-scale real-world dataset called WS-DREAM[8], involving 5,825 real-world Web services from 74 regions and 339 service users from 31 regions. By comparing NCRL with several state-of-the-art baselines, the results validate its effectiveness on multiple evaluation metrics for QoS prediction.

The main contributions are summarized as follows:

- We propose a novel framework NCRL for effective QoS prediction, which integrates context-aware deep neural network and collaborative relationships of similar neighborhood by historical QoS invocations, leading to better QoS prediction accuracy.
- We propose an adaptive QoS prediction approach, where multiple location information are taken as heuristics to our designed two-tower deep residual network for more effective neural QoS prediction by precisely mining the latent features of users and services. It is beneficial to collaborative QoS prediction based on historical QoS invocations, where similar neighbors are calculated by the extracted latent feature representations. They are combined together with weight factors to adaptively predict an unknown QoS.
- Extensive experiments have been conducted on a large number of real-world QoS dataset to evaluate the performance. The results demonstrate that NCRL can remarkably outperform state-of-the-art baseline approaches on prediction accuracy, while achieving superior prediction efficiency with lower computation cost.

The remainder of this paper is organized as follows. Section 2 formulates QoS prediction problem. Section 3 illustrates the overall framework of NCRL. Section 4 presents the approach in detail. Section 5 shows and analyzes the experimental results. Section 6 reviews the related work. Finally, Section 7 concludes the paper and discusses the future work.

2 Problem Formulation

Definition 1 (Service User). *Service users mainly refer to those who have invoked one or more Web services. Let* $U = \{u_1, u_2, \ldots, u_m\}$ *be a set of users. For each* $u \in U$*, it can be described as a five-tuple* $u =< ID, RG, AS, Lat, Lon >$ *. ID is the identifier of* u *and the rest can be collectively represented as location information.*

Here, a service user's location information mainly includes Region (RG), Autonomous System (AS), Latitude (Lat.) and Longitude (Lon.), respectively.

Definition 2 (Web Service). *For QoS prediction problem, we mainly focus on the non-functional features of a Web service. Let* $S = \{s_1, s_2, \ldots, s_n\}$ *be a set of Web services. For each* $s \in S$*, it can be described as a five-tuple* $s =< ID, RG, AS, Lat, Lon >$ *. ID is the identifier of* s *and the rest can be collectively represented as location information.*

Definition 3 (User-Service Invocation Record). *Given a user set* U *and a service set* S*, a user-service invocation record is defined as a three-tuple* $r =< u, s, r_{u,s} >$ *, where* $u \in U$ *is a service user,* $s \in S$ *is a Web service, and* $r_{u,s}$ *is QoS value when* u *invokes* s*.*

A user-service invocation set R can be obtained by collecting all of the invocation records among users and services. If a user u_i has invoked a service s_j, we have $< u_i, s_j, r_{u_i,s_j} > \in R$, otherwise $< u_i, s_j, r_{u_i,s_j} > \notin R$.

Definition 4 (QoS Prediction Problem). *Given a set of users* U*, a set of services* S *and all observed QoS invocation records* R*, a QoS prediction problem can be defined as* $\Omega =< U, S, R, u, s >$ *, where* $u \in U$ *is a target user,* $s \in S$ *is a target service, and* $< u, s, r_{u,s} > \notin R$*.*

The solution to a QoS prediction problem Ω is $< u, s, \hat{r}_{u,s} >$. It indicates the predicted QoS value when a target user invokes a target service, by exploiting the provided information of invocation records among users and services. Based on a set of prediction results, desired services with high QoS can be recommended to service users.

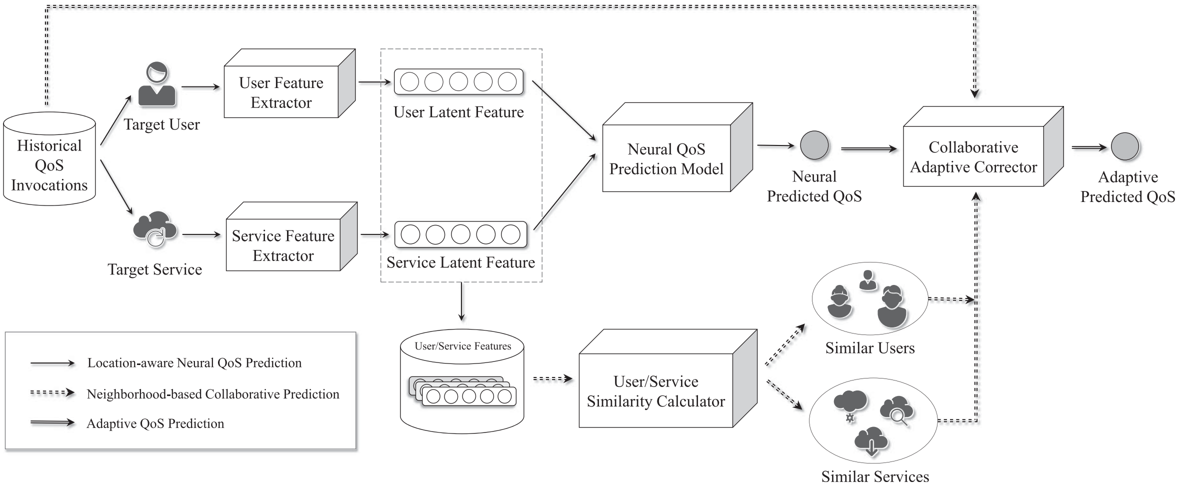

Historical QoS Invocations
Target User
Target Service
User Feature Extractor
Service Feature Extractor
User Latent Feature
Service Latent Feature
Neural QoS Prediction Model
Neural Predicted QoS
Collaborative Adaptive Corrector
Adaptive Predicted QoS
User/Service Features
User/Service Similarity Calculator
Similar Users
Similar Services
Location-aware Neural QoS Prediction
Neighborhood-based Collaborative Prediction
Adaptive QoS Prediction

Fig. 1. Overall framework of neighborhood-based collaborative residual learning for adaptive QoS prediction.

In a natural approach, QoS objectives might differ and often conflict. We may consider choosing the most suitable one among the candidate services with reference to different QoS criteria such as response time, throughput and reliability. In practical scenarios, it needs to evaluate the importance of various QoS objectives with different metrics and scales, and makes a trade-off among diverse QoS objectives to select a Web service that comprehensively satisfies users' concerned QoS requirements instead of achieving the best quality on one of the QoS criteria.

3 The Framework of NCRL

Fig. 1 is the overall framework of NCRL for adaptive QoS prediction. The goal of NCRL is to automatically predict an unknown QoS value, when a target user aims at invoking a target Web service. It consists of three crucial stages, including location-aware neural QoS prediction, neighborhood-based collaborative prediction, and adaptive QoS prediction. The processes of the three stages are marked with different arrow types and described as below.

- In the stage of location-aware neural QoS prediction, the identifiers and location information of users and services are transformed into dense vector representations. A two-tower deep residual network is designed to learn latent and discriminative feature representations of users and services, which are further used to perform neural QoS prediction and also fed into neighborhood-based collaborative prediction for finding similar neighborhood of users and services.
- In the stage of neighborhood-based collaborative prediction, those vacant QoS are estimated by using the historical QoS invocations of similar neighborhood. To find a set of similar users or services for a target user or service, similarity calculation is performed based on the extracted latent feature vectors in the previous stage.
- In the stage of adaptive QoS prediction, neural predicted QoS and collaborative predicted QoS are integrated by weight coefficient to adaptively predict the final unknown QoS.

4 Approach

4.1 Location-Aware Neural QoS Prediction

Fig. 2 illustrates the architecture of location-aware two-tower deep residual network, which consists of two independent sub-networks for users and services, respectively. When performing the neural QoS prediction, it has four layers including location input layer, location embedding layer, latent feature extraction layer and neural QoS prediction layer. Table 1 presents all the notations.

4.1.1 Latent Feature Extraction of Users and Services

Location Input Layer. The function of the layer is mainly to generate an initial feature representations for a user and a service. As shown in Fig. 2, each tower has its own location input layer, which receives the identifiers and location information of users or services and integrates them separately. Each user or service has its own unique ID, which is represented by a non-negative integer. In addition, the longitude and latitude of a user or a service is represented as real numbers. To generate a user's or a service's initial feature vector, we make basic conversions on RG and AS. Specifically, since the number of regions and autonomous systems is limited, and simultaneously each user or service corresponds to only one region and one autonomous system, we perform a mapping from all regions and autonomous systems to a corresponding non-negative integer set.

User-service invocations involve complex contextual environments, such as network status, user device performance, and service runtime workload. To a large extent, network status of users and services can particularly affect non-functional performance of Web services, such as response time and throughput. Correlative investigations [20], [22], [23] have demonstrated that location information is of great importance and extremely determines the network status by geographical distance between users and services, which leads to the differentiation of QoS invocations. In such case, users may receive a better QoS experience when they invoke those services that are geographically closer to them. Therefore, we leverage location information of both users and services as heuristics to auxiliarily promote the effectiveness of QoS prediction performance.

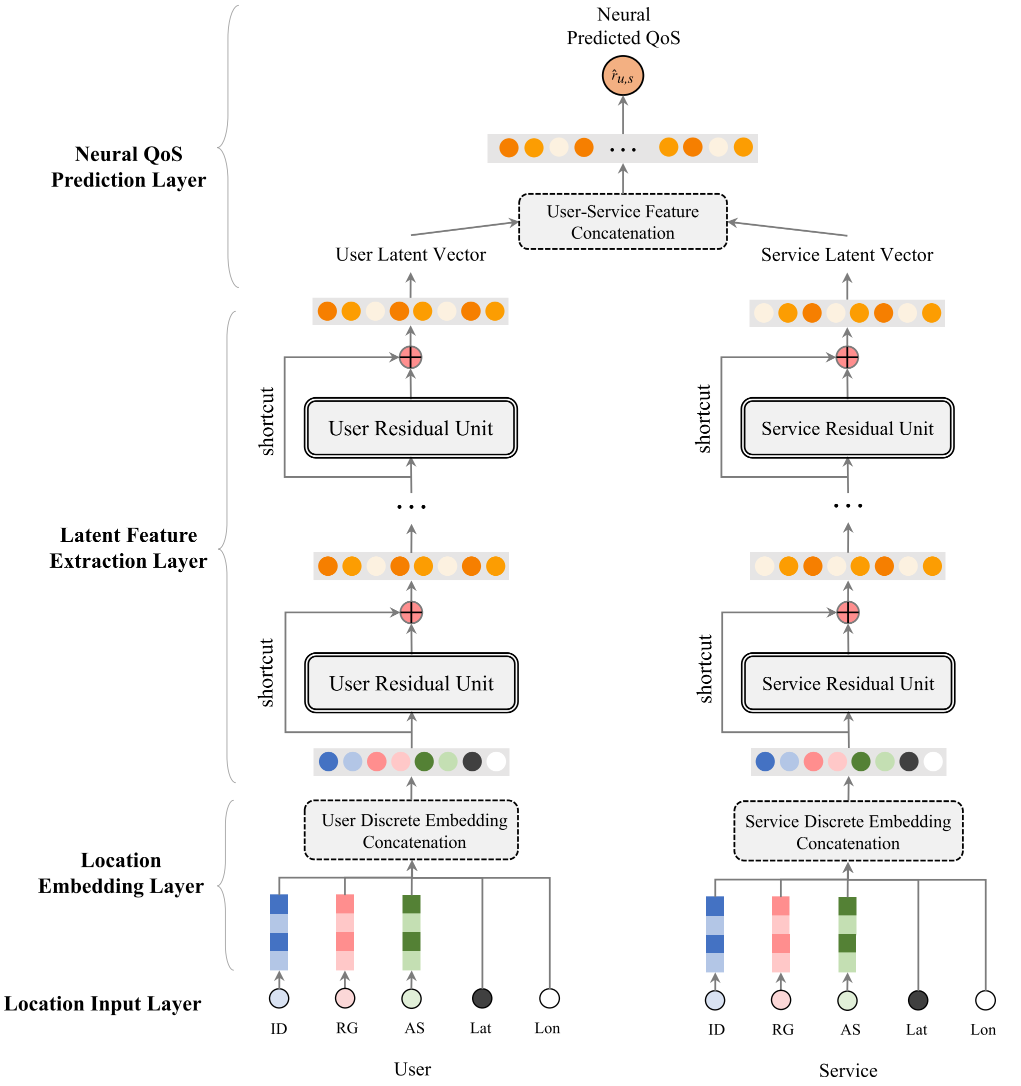


Fig. 2. Location-aware neural QoS prediction by two-tower deep residual network.

TABLE 1
Notations

Notation	Description
U	a set of users
S	a set of Web services
I_u	initial feature vector of a user u
I_s	initial feature vector of a Web service s
x_u	embedded feature vector of a user u
x_s	embedded feature vector of a Web service s
$\mathbf{W}_j$	embedding weight matrix of the j^{th} discrete feature
$\mathbf{b}_j$	bias term of the j^{th} discrete feature
f_a	ReLU activation function of location embedding layer
g_a	GELU activation function of latent feature extraction layer
x'_u	latent feature of a user u
x'_s	latent feature of a service s
$X_{u,s}$	invocation feature of a user u and a service s
$\hat{r}_{u,s}$	neural predicted QoS
$\hat{r}'_{u,s}$	collaborative predicted QoS
$\hat{R}_{u,s}$	adaptive predicted QoS

Specifically, the initial features of a user and service are represented as two multi-dimensional vectors as follows:

$$I_u = (u_{ID}, u_{RG}, u_{AS}, u_{Lat}, u_{Lon}) \quad (1)$$

$$I_s = (s_{ID}, s_{RG}, s_{AS}, s_{Lat}, s_{Lon}) \quad (2)$$

where I_u and I_s are the initial feature vectors. Suppose that there is a user302 from China, located in the AS of China Education and Research Network Center, whose latitude and longitude are (35.0, 105.0). Then we can obtain the user's initial feature vector, which is represented as:

$$I_{u_{302}} = (302,\ 29,\ 123,\ 0.6983,\ 0.7898)$$

where region and autonomous system are mapped into 29 and 123. For latitude and longitude, we use the transformation tools from the scikit-learn[2] to scale them into the range of $[0, 1]$.

Location Embedding Layer. Given a user's or a service's initial feature vector I_u or I_s, we apply embedding on each discrete feature of the initial feature vector. That is, only those

[2] https://scikit-learn.org/

discrete features, such as ID, region and AS, are represented as one-hot encoding and mapped into dense feature vectors with location embedding layer, respectively. It can be regarded as a fully-connected network, which embeds one-hot encoding vectors of each discrete feature into high-dimensional but dense representation space. The formula [24] is expressed as follows:

$$E_j = f_a(\mathbf{W}_j X_j + \mathbf{b}_j) \tag{3}$$

where j indexes a certain discrete feature, $X_j \in \mathbb{R}^{n_j}$ represents one-hot encoding vector of the feature, $\mathbf{W}_j \in \mathbb{R}^{m_j \times n_j}$ represents the embedding weight matrix, $\mathbf{b}_j \in \mathbb{R}^{m_j}$ represents the bias term, f_a represents the ReLU activation function of location embedding layer, and $E_j \in \mathbb{R}^{m_j}$ is the embedded feature.

Taking the above embedding function in Eq. (3), we transform the initial feature vectors of a user's and service's ID, region and AS into their corresponding embedded feature vectors: $E_{u_{ID}}, E_{u_{RG}}, E_{u_{AS}}, E_{s_{ID}}, E_{s_{RG}}, E_{s_{AS}}$. Note that we set $m_j < n_j$ when taking discrete feature embedding, since it is used to reduce the dimensionality of the transformed one-hot encoding representations from the initial feature vectors. On this basis, these independently embedded features are combined together with the latitude and longitude features to obtain a user's and service's embedded feature vectors x_u and x_s , respectively. The concatenation is expressed as follows:

$$x_u = \Phi(E_{u_{ID}}, E_{u_{RG}}, E_{u_{AS}}, u_{Lat}, u_{Lon}) = \begin{bmatrix} E_{u_{ID}} \\ E_{u_{RG}} \\ E_{u_{AS}} \\ u_{Lat} \\ u_{Lon} \end{bmatrix} \tag{4}$$

$$x_s = \Phi(E_{s_{ID}}, E_{s_{RG}}, E_{s_{AS}}, s_{Lat}, s_{Lon}) = \begin{bmatrix} E_{s_{ID}} \\ E_{s_{RG}} \\ E_{s_{AS}} \\ s_{Lat} \\ s_{Lon} \end{bmatrix} \tag{5}$$

where Φ represents the concatenation operation, x_u and x_s denote a user's and service's embedded feature vectors, respectively. These embedded feature vectors are fed into the latent feature extraction layer.

Latent Feature Extraction Layer. In this layer, we take the embedded feature vectors as inputs and extract latent features of users and services by Residual Net [25] that solves the problem of neural network performance degradation caused by the increase of network layers. Generally, the Residual Net consists of a large number of convolutional layers. In the model of Deep Crossing [24], an improved Residual Unit has been used instead of convolutional kernels, which extends and boosts the capability for many application scenarios. Inspired by Deep Crossing, we apply the improved Residual Units to further learn latent and discriminative feature representations of users and services.

The latent feature extraction layer is constructed from a set of Residual Units of users and services, respectively. As illustrated in Fig. 3, a Residual Unit consists of two nonlinear layers and an identity shortcut. The input feature vector of a Residual Unit is added back after passing through two nonlinear layers. Formally, the feature propagation and aggregation of a user or service Residual Unit is as follows:

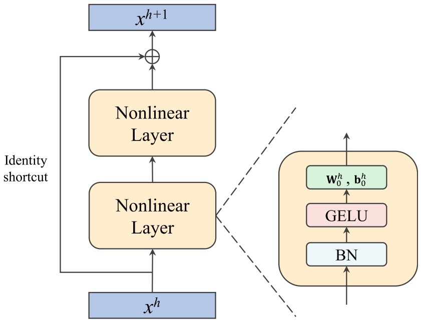


Fig. 3. Residual Unit of users and services in latent feature extraction layer.

$$Y^h = \mathbf{W}_0^h g_a(\mathrm{BN}(x^h)) + \mathbf{b}_0^h \tag{6}$$

$$Z^h = \mathbf{W}_1^h g_a(\mathrm{BN}(Y^h)) + \mathbf{b}_1^h \tag{7}$$

$$x^{h+1} = Z^h + x^h \tag{8}$$

where x^h represents the input of the h^{th} Residual Unit, BN represents the Batch Normalization [26], g_a represents the GELU activation function, $\mathbf{W}_{\{0,1\}}^h$ and $\mathbf{b}_{\{0,1\}}^h$ are the parameters of the two layers in the h^{th} Residual Unit, x^{h+1} is the output of the h^{th} Residual Unit, and it is also the input of the $h+1^{th}$ Residual Unit.

The used activation function is Gaussian Error Linear Unit (GELU) [27], which introduces the idea of random regularization into activation. The GELU nonlinearity weights inputs by their value, rather than gates inputs by their sign as in ReLUs [27]. The formula [27] is expressed as follows:

$$\begin{aligned} \mathrm{GELU}(x) &= xP(X \le x) = x\Phi(x) \\ &= x \cdot \frac{1}{2}[1 + \mathrm{erf}(x/\sqrt{2})] \end{aligned} \tag{9}$$

where erf represents gauss error function.

Note that similar model architectures with deep layers, such as MLP and CNN, can also learn a user's and service's latent feature representations for QoS prediction. However, it is observed that as the depth of these deep neural networks increases, the performance of unknown QoS prediction gets saturated and then degrades rapidly. In other words, the increasing network layers cause the instability of these neural networks, but Residual Units applied in our model can perform some kind of regularization that leads to network stability [28] beneficial to latent feature extraction of users and services for better predicting vacant QoS.

Furthermore, Unlike the Residual Unit in [24], we leverage a "full pre-activation" structure as in [28]. In contrast to the conventional wisdom of "post-activation", the activation functions (GELU and BN) are moved to the front of the weight layers. Such adjustment can significantly improve the performance of the transformation and propagation of a user's or service's embedding feature. Meanwhile, moving BN to the front of GELU in each tower of user and service feature extraction has the same advantageous effect as regularization, which reduces over-fitting issue of our model, enhancing the performance of QoS prediction.

Consequently, given a user's embedded vector x_u and a service's embedded vector x_s, the latent features can be extracted and expressed as follows:

$$x'_u = \mathrm{RL}_u^H(x_u) \tag{10}$$

$$x'_s = \mathrm{RL}_s^H(x_s) \tag{11}$$

where RL_u^H and RL_s^H represent the functions of residual layers with H Residual Units in user-tower network and service-tower network, respectively. x'_u and x'_s are the extracted latent features of a user and a service.

In NCRL, user-tower network and service-tower network are designed similarly with the same multi-layer architecture for extracting latent features of users and services, although they can be trained independently by specific network hyperparameters optimization in parallel.

4.1.2 Neural QoS Prediction and Model Training

Given latent features x'_u and x'_s, a nonlinear transformation is applied for neural QoS prediction. First, the latent features of a user and service are concatenated into an invocation feature, which is denoted as $X_{u,s}$ and expressed by:

$$X_{u,s} = \Phi(x'_u, x'_s) = \begin{bmatrix} x'_u \\ x'_s \end{bmatrix} \tag{12}$$

where Φ represents the concatenation operation. Then, a neural QoS prediction layer is applied by linear transformation, which is expressed as:

$$\hat{r}_{u,s} = f_i(\mathbf{W}^O X_{u,s} + \mathbf{b}^O) \tag{13}$$

where $\mathbf{W}^O$ and $\mathbf{b}^O$ are the parameters to be learned by our model, f_i represents the identity function, and $\hat{r}_{u,s}$ is the result of neural QoS prediction when a target user u invokes a target service s.

To effectively perform neural QoS prediction, our model is trained by the loss function, which measures mean absolute error for overall prediction performance, rather than sensitive to those outliers[29]. The loss function of model parameter optimization J is expressed as:

$$J = \frac{1}{N}\sum_{k=1}^{N}\left|\mathcal{F}(x_k, \Theta) - R_{x_k}\right| + \lambda_\Theta \mathbb{L}_{reg}(\Theta) \tag{14}$$

where N is the batch size, Θ denotes all the parameters to be learned, $\mathcal{F}$ denotes the function of our two-tower deep neural network which maps the input x_k to the predicted QoS value, and R_{x_k} denotes the real QoS value. Additionally, $\lambda_\Theta \mathbb{L}_{reg}(\Theta)$ is the regularization item, which prevents our model from overfitting.

The goal of the training process is to minimize J. We use Adam optimizer [30] to update all the parameters that need to be optimized in NCRL for latent feature extraction of users and services as expressed in (15):

$$\Theta - \eta \frac{\partial J}{\partial \Theta} \rightarrow \Theta \tag{15}$$

where η is the learning rate. Once the model converges, latent feature vectors of users and services are generated and concatenated as invocation feature for neural QoS prediction.

4.2 Neighborhood-Based Collaborative Prediction

Although our designed two-tower deep residual network can capture user-service invocations by their location information, the historical invocations are not fully utilized for QoS prediction. In particular, some implicit similarity relationships among users and services are not intuitively and directly mined by deep neural network. To further improve the QoS prediction accuracy, we perform neighborhood-based collaborative prediction by similarity calculation.

Based on the extracted latent features, we generate similar neighborhood of users and services, respectively. Given two latent features of a target user x'_u and a candidate similar user x'_{u_c}, we use (16) to measure the similarity of two users:

$$Sim(x'_u, x'_{u_c}) = \frac{x'_u x'^{\top}_{u_c}}{\|x'_u\| \|x'_{u_c}\|} \tag{16}$$

where x'_u and x'_{u_c} are latent feature vectors, and $Sim(x'_u, x'_{u_c})$ is the cosine similarity.

Analogously, given two latent features of a target service x'_s and a candidate similar service x'_{s_c}, the similarity of two services is calculated as:

$$Sim(x'_s, x'_{s_c}) = \frac{x'_s x'^{\top}_{s_c}}{\|x'_s\| \|x'_{s_c}\|} \tag{17}$$

where x'_s and x'_{s_c} are latent feature vectors, and $Sim(x'_s, x'_{s_c})$ is the cosine similarity.

Given a target user u, it corresponds to a set of services S_u that have been invoked by u. By applying service similarity calculation in (17) with a threshold θ, we generate a subset of service neighborhood S'_u that have been invoked by u and simultaneously share highly close similarity with a target service s. The generation of service neighborhood S'_u is formally expressed as follows:

$$S'_u = \left\{ s' \in S_u \mid Sim(x'_s, x'_{s'}) \geq \theta \right\} \tag{18}$$

Likewise, given a target service s, we can obtain a set of users U_s that have invoked s. By applying user similarity calculation in (16) with a threshold θ, we can obtain a set of user neighborhood U'_s, where each user $u' \in U_s$ has high similarity with the target user u. It can be expressed by:

$$U'_s = \left\{ u' \in U_s \mid Sim(x'_u, x'_{u'}) \geq \theta \right\} \tag{19}$$

After filtering out all of those dissimilar users and services from U_s and S_u, the generated service neighborhood S'_u of a target user and user neighborhood U'_s of a target service are used for collaborative prediction. Specifically, taking the above similarity between two users or two services as weight coefficients, we combine them together to predict unknown QoS based on historical QoS invocations. Formally, given a target user u and a target service s, collaborative predicted QoS $\hat{r}'_{u,s}$ is calculated as follows:

$$\hat{r}'_{u,s} = \frac{\sum_{u' \in U'_s} Sim(x'_u, x'_{u'}) * r_{u',s} + \sum_{s' \in S'_u} Sim(x'_s, x'_{s'}) * r_{u,s'}}{\sum_{u' \in U'_s} Sim(x'_u, x'_{u'}) + \sum_{s' \in S'_u} Sim(x'_s, x'_{s'})} \tag{20}$$

where U'_s and S'_u are user neighborhood of a target service s and service neighborhood of a target user u, respectively. $r_{u',s}$ and $r_{u,s'}$ represent the historical invocation QoS, when a user neighborhood u' has invoked the target service s, or the target user u has invoked a service neighborhood s'.

4.3 Adaptive QoS Prediction

Adaptive QoS prediction takes full advantage of neural QoS prediction and neighborhood-based collaborative prediction. Neural QoS prediction is based on two-tower deep residual network that can effectively extract latent features of users and services by location information, which is applied for calculating similar neighborhood to collaboratively predict QoS by historical QoS invocations.

Based on the results of above neural QoS prediction and neighborhood-based collaborative prediction, the finally adaptive predicted QoS is calculated as follows:

$$\widehat{R}_{u,s} = \alpha \hat{r}_{u,s} + \beta \hat{r}'_{u,s} \tag{21}$$

where $\hat{r}_{u,s}$ is neural predicted QoS, $\hat{r}'_{u,s}$ is neighborhood-based collaborative predicted QoS, and $\widehat{R}_{u,s}$ is finally adaptive predicted QoS. Here, α and β are the weight coefficients with $\alpha, \beta \in [0, 1]$ and $\alpha + \beta = 1$.

5 Experiments

5.1 Experimental Setup and QoS Dataset

All the experiments are carried out on our workstation equipped with two NVIDIA GTX 1080Ti GPUs, an Intel(R) Xeon(R) Gold 6130 @2.60 GHz CPU and 192GB RAM. The components of NCRL in the experiments are implemented by Python 3.7.1 with Pytorch 1.4.0.

To validate the performance of NCRL, we conduct extensive experiments on a publicly available large-scale real-world QoS dataset called WS-DREAM [8], which has been widely used for QoS prediction verification. It consists of two kinds of QoS criteria, including response time (RT) and throughput (TP), which totally has 1,974,675 historical QoS invocation records collected from 339 users and 5,825 Web services. In addition, multi-source location information of these users and services are provided in RT and TP, such as region, latitude and longitude. More detailed statistics of the QoS dataset is shown in Table 2.

The QoS dataset of RT or TP can be represented as a user-service QoS matrix, where a row represents a set of QoS values that a user invokes all of the services, and a column represents a set of QoS values that a service is invoked by all of the users. Considering the sparsity of user-service invocations in real application scenarios, QoS dataset is set as four different low densities for model training on RT and TP, including 2.5%, 5%, 7.5% and 10%, respectively. For the comparisons of QoS prediction accuracy, remaining QoS samples under each density are used as testing data in the experiments.

5.2 Evaluation Metrics

In the experiments, we compare the performance of QoS prediction among NCRL and competing baselines by two evaluation metrics, including Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). They have been widely used to measure the accuracy of QoS prediction.

Let $r_{u,s}$ and $\hat{r}_{u,s}$ denote the original and predicted QoS of a service s invoked by a user u, respectively. MAE and RMSE are applied to quantify the deviation between predicted QoS and the original one, which are defined as:

TABLE 2
Statistics of QoS Dataset

Item Name	Value
Users	339
Services	5825
Service Invocations	1,974,675
Users' Regions	31
Users' AS	137
Services' Regions	74
Services' AS	992
Services' Providers	2699

$$MAE = \frac{\sum_{u,s} |r_{u,s} - \hat{r}_{u,s}|}{N} \tag{22}$$

$$RMSE = \sqrt{\frac{\sum_{u,s} (r_{u,s} - \hat{r}_{u,s})^2}{N}} \tag{23}$$

where N is the number of the predicted QoS values. Here, it is observed that smaller deviations on MAE and RMSE indicate better performance of vacant QoS prediction.

MAE evaluates the overall accuracy of QoS prediction by calculating the averaged absolute deviations to the ground truth QoS values, while RMSE is more sensitive to individual outliers.

5.3 Competing Methods

To evaluate the performance of NCRL, we compare it with nine competing baselines, including three memory-based conventional approaches [8], [9], two model-based matrix factorization approaches [13], [31], and four model-based deep neural network approaches [19], [20], [21], [32]. They are described as below.

- *UPCC* [9]: It is a user-based QoS prediction method that finds a set of similar users as the neighborhood of a target user with PCC, and combines the average QoS of the target user with the deviation migration based on the found similar users.
- *IPCC* [8]: It is a service-based QoS prediction method that finds a set of similar services as the neighborhood of a target service with PCC, and combines the average QoS of the target service with the deviation migration based on the found similar services.
- *UIPCC* [8]: It is a hybrid CF method for QoS prediction by the combination of UPCC and IPCC, which applies a weighting coefficient to adjust the importance of UPCC and IPCC. It provides a fundamental way of predicting missing QoS by simultaneously integrating similar users and services.
- *PMF* [13]: It is a probabilistic matrix factorization method which utilizes probability model to optimize matrix factorization. It refers to a model-based representative approach for QoS prediction by improving the traditional matrix factorization.
- *FM* [31]: It is a factorization machine method which combines the generality of feature engineering with the superiority of factorization models. It is a model-based approach and can be introduced to predict unknown QoS.

TABLE 3
Parameter Settings

Parameter	Value
batch size	128
epoch	200
d (dimensionality)	16
$\mathcal{H}$ (hidden neurons)	<64, 128, 64, 64, 32, 32>
η (learning rate)	0.0005
θ (similarity threshold)	0.5
α (weight coefficient)	0.8 for RT, 0.7 for TP

- *NCF* [32]: It is a deep neural network based collaborative filtering method which solves the disadvantage on feature interaction of matrix factorization by inner product. It is a model-based approach based on MLP that can be applied for QoS prediction by learning complex nonlinear interaction relationships among users and services.
- *DNM* [19]: It is model-based deep neural network approach for multi-attributes QoS prediction, which maps contextual features into a shared latent space and captures their high-order interactions through the interaction layer and the perception layers.
- *NDMF* [21]: It is our previously proposed model-based deep neural network approach for QoS prediction. By taking advantage of both the historical QoS records and users' geographical information, it generates similar users as neighborhood, which is loosely integrated into a deep neural network model via multi-layer perceptron.
- *LDCF* [20]: It is a model-based deep neural network method for QoS prediction. It integrates MLP with a similarity adaptive corrector, which has good adaptability in exploiting contextual information such as locations of users and services.

5.4 Experiment Results and Analyses

To validate the effectiveness of our proposed NCRL, we tune the model parameters of competing methods directly as they are suggested with the best performance in the experiments of the references. As for NCRL, the parameters settings are shown in Table 3. In the experiments, historical QoS records from response time (RT) and throughput (TP) are partitioned into four different densities, including 2.5%, 5%, 7.5% and 10%, respectively. All the competing methods are run on both RT and TP training sets, and QoS prediction performance is evaluated on the test sets by calculating MAE and RMSE. To avoid the deviations, we run NCRL and competing methods several times to report the average results for the guarantee of fair performance comparisons.

The experimental results of QoS prediction among competing methods on RT and TP are shown in Tables 4 and 5, respectively. Here, the best results are marked in bold and the second-best results are highlighted in the gray background. We also calculate the performance gains on them. As can be seen from the results on MAE and RMSE, NCRL consistently and remarkably outperforms all competing methods by up to 11.0% in terms of MAE, and 8.7% in terms of RMSE. More specifically, as the increasing QoS density from 2.5% to 10% on RT and TP, it is observed that MAE and RMSE of all the competing methods become smaller and smaller, indicating better QoS prediction accuracy. It can be reasonably explained that more historical QoS training data can be provided for finding similar neighborhood and learning a better prediction model, as the QoS density continues to rise. As expected of our NCRL, it becomes gradually better and always receives superior QoS prediction performance across multiple QoS densities compared with the competing baselines.

UPCC, IPCC and UIPCC as basic CF methods perform poorly in QoS prediction because they mainly rely on historical QoS invocations to find similar users and services for QoS prediction, which is significantly vulnerable to the sparsity of user-service QoS invocations. As the basic matrix factorization method, PMF introduces a probability model to perform matrix decomposition, which can be used to partially solve the sparsity of QoS density [13] and applied for better missing QoS prediction than conventional CF methods. Moreover, FM pays more attention to linear feature interaction learning, which can generate better QoS prediction performance than basic MF methods.

To further improve the QoS prediction accuracy, NCF leverages multi-layer perceptron (MLP) that mines nonlinear interaction relationships from the embedded feature vectors of users and services. Although it can outperform MF competing methods, QoS prediction accuracy is still worse than DNM, NDMF and LDCF, since it has ignored the contextual information when extracting latent features of users and services. DNM and LDCF take full advantage

TABLE 4
Performance Comparisons of QoS Prediction Among Competing Methods on Response Time

Methods	Density = 2.5%		Density = 5%		Density = 7.5%		Density = 10%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UPCC	0.7679	1.7888	0.6166	1.5287	0.5724	1.4197	0.5550	1.3807
IPCC	0.7380	1.7769	0.6727	1.6981	0.6471	1.6728	0.6261	1.6367
UIPCC	0.7515	1.7549	0.6078	1.5023	0.5670	1.4064	0.5502	1.3684
PMF	0.6492	1.6149	0.5753	1.4422	0.5252	1.3370	0.4954	1.2778
FM	0.6876	1.5321	0.6203	1.4406	0.5592	1.3281	0.5392	1.3052
NCF	0.5444	1.5472	0.4652	1.3904	0.4159	1.3583	0.3783	1.3040
DNM	0.4777	1.4829	0.4147	1.4274	0.3843	1.3745	0.3628	1.3567
NDMF	0.5393	1.4036	0.4880	1.3495	0.4416	1.2793	0.4304	1.2349
LDCF	0.4525	1.3773	0.4031	1.3102	0.3799	1.2875	0.3642	1.2358
NCRL	**0.4098**	**1.3316**	**0.3589**	**1.2694**	**0.3420**	**1.2401**	**0.3385**	**1.2252**
Gains	9.4%	3.3%	11.0%	3.1%	10.0%	3.1%	6.7%	0.8%

TABLE 5
Performance Comparisons of QoS Prediction Among Competing Methods on Throughput

Methods	Density = 2.5%		Density = 5%		Density = 7.5%		Density = 10%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UPCC	37.84	93.33	25.42	65.68	22.96	58.83	21.25	57.24
IPCC	37.25	97.50	32.96	89.85	31.02	87.56	29.76	84.90
UIPCC	36.87	91.85	25.18	65.37	22.93	59.17	22.43	57.56
PMF	30.18	74.67	24.20	56.03	22.52	55.97	20.40	51.75
FM	28.57	72.30	21.59	57.60	19.47	50.51	17.69	48.62
NCF	24.21	64.15	18.68	54.65	15.88	48.38	14.40	46.22
DNM	18.29	65.65	14.85	59.33	13.82	56.55	12.92	54.50
NDMF	20.31	58.17	16.38	50.96	15.28	47.60	13.93	43.91
LDCF	16.47	59.02	14.24	48.46	13.52	47.70	12.42	43.79
NCRL	**16.05**	**53.09**	**13.49**	**46.25**	**12.63**	**43.72**	**11.84**	**41.45**
Gains	2.6%	8.7%	5.3%	4.6%	6.6%	8.2%	4.7%	5.3%

of contextual information when encoding the initial features, which are integrated into a multi-layer perceptron network for better learning the implicitly complex nonlinear interaction relationships. Thus, they are superior to NCF in terms of MAE across multiple QoS densities, even though in some cases DNM cannot perform well on RMSE. Compared to DNM and LDCF, NDMF considers both historical QoS records and users' geographical information for finding similar users as neighborhood, which is collaborated with MLP for training QoS prediction model, leading to better performance on RMSE at specified QoS densities.

Inspired by the competing baselines, NCRL effectively performs adaptive QoS prediction, where location-aware two-tower deep residual network is designed to more precisely extract latent features for neural QoS prediction that is positively upgraded by collaborative QoS prediction based on historical QoS invocations. Consequently, we conclude that the proposed NCRL obtains the best QoS prediction accuracy on MAE and RMSE among all the competing methods across multiple QoS densities.

As mentioned above, geographical distance between users and services has a great impact on QoS experience, so we use location-aware features to predict QoS values. Other competing methods based on neural networks like [19], [20], [21], [32] apply fully-connected layers to extract latent features, which are prone to overfitting when neural networks are complex and have multiple layers. As for NCRL, it can fully leverage the benefits of residual learning, which solves the problem of network degradation occurred in the fully-connected layer. Thus, despite the complex structure of NCRL, it still maintains the effectiveness of latent feature extraction.

The time consumption of NCRL is mainly from the two-tower deep residual network. Specifically, let N_u and N_s denote the dimensionality of location feature of a user or a service, E_u and E_s denote the dimensionality of embedded feature of a user or a service. By calculating in Eq. (3), the computational complexity of the fully-connected layer for location feature embedding is $O(N_uE_u + N_sE_s)$. Let d denote the dimensionality of hidden feature in Residual Units, h denote the number of Residual Units. By calculating in Eqs. (10) and (11), computational complexity of residual layer is $O(hdE_u^2 + hdE_s^2)$. By calculating in Eq. (13), computational complexity of prediction layer is $O(E_u + E_s)$.

From the above analyses, it is observed that the computational complexity of NCRL is $O(hd(E_u^2 + E_s^2) + N_uE_u + N_sE_s + E_u + E_s)$. Compared with conventional competing baselines, the network architecture of NCRL is more complex, resulting in additional time consumption when performing model training and parameter optimization in an offline way. However, neural QoS prediction layer only needs to receive latent feature vectors of users and services to perform the task of unknown QoS prediction with the computational complexity of $O(E_u + E_s)$ in an online way, which can be efficiently deployed for real-time response in service-oriented application scenarios.

In addition, we also observe that although the competing approaches have higher training accuracy of unknown QoS prediction than the proposed NCRL due to the overfitting problem of their latent feature learning and representation, while our designed two-tower deep residual learning-based unknown QoS prediction model has advantageous properties, such as high generalization learning capability with better test prediction accuracy. Therefore, NCRL can more effectively satisfy the demands of QoS prediction in real service-oriented application scenarios.

However, the performance of NCRL is highly influenced by the tuning and optimization of hyperparameters. Currently, there is not an effective theoretical methodology to guide the setting of hyperparameters. Therefore, we need to conduct further experiments to determine the best settings of model hyperparameters of NCRL, considering multiple factors such as application preference, density and distribution of user-service QoS invocations. Moreover, one main disadvantage of cosine similarity used in NCRL is that the magnitude of feature vectors is ignored and only their direction is taken into account, which reduces the performance of QoS prediction. It is expected that more sophisticated similarity measurements can be applied to calculate user and service neighborhood for better collaborative QoS prediction, such as euclidean distance, Pearson Correlation Coefficient, and Ratio-Based Similarity [5].

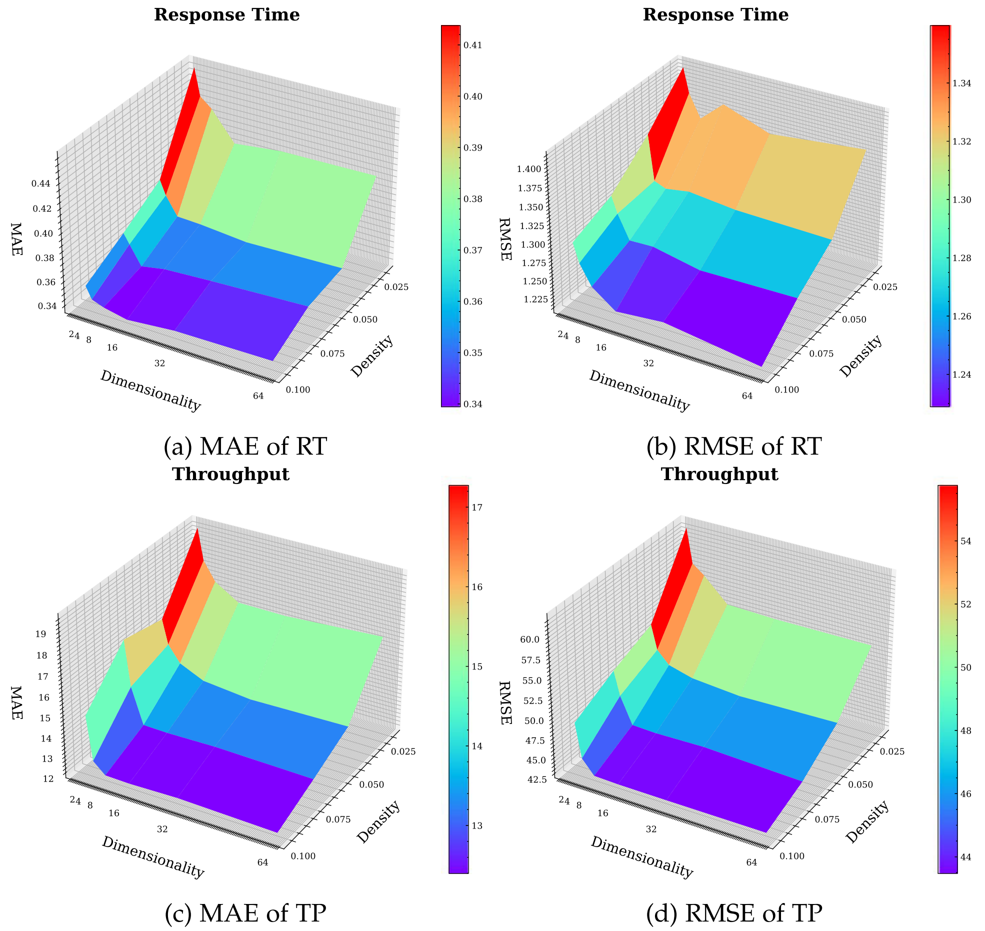
5.5 Performance Impact of Parameters

In the experiments, three main hyperparameters significantly impact the performance of our proposed approach NCRL, including the dimensionality of latent features of users and services, the number of Residual Units, and the weight coefficient of neural QoS prediction.

5.5.1 Impact of Dimensionality

The dimensionality d determines the dimension of embedding vectors in the location embedding layer. Due to the identity shortcut, feature vectors of users and services extracted in the latent feature extraction layer, have the same dimensionality as the embedding vectors. Thus, the dimensionality also impacts how much useful information is utilized to represent the latent features.

To test the performance impact of d, we vary its value by 2, 4, 8, 16, 32, 64 and set QoS matrix density as 0.025, 0.05, 0.075 and 0.1, respectively. The results are illustrated in Fig. 4. It can be seen from the three-dimensional graph of the experimental results that the MAE and RMSE show a decreasing trend with the increasing number of dimensionality. More



(a) MAE of RT

(b) RMSE of RT

(c) MAE of TP

(d) RMSE of TP

Fig. 4. Performance impact of dimensionality d on NCRL under different QoS densities.

specifically, when the dimensionality varies from 2 to 16, it reaches the largest decreasing changes, and then it becomes smaller or even increases within the variations from 16 to 64. The primary reason is that, when the dimensionality of the latent features of users and services is too small, it may result in partial hidden information lost from feature vectors, which affects feature representation ability and reduces QoS prediction accuracy. Conversely, when the dimensionality of the feature vectors tunes to be too large, it may potentially cause the issue of feature sparseness. It is not conducive to two-tower deep residual network in NCRL to mine the implicitly complex nonlinear interaction relationships among users and services. Therefore, choosing an appropriate dimensionality has an important impact on feature interaction learning and QoS prediction. Considering the variation trends of MAE and RMSE under different QoS densities among response time and throughput, the dimensionality is set as 16 to achieve the best performance of QoS prediction.

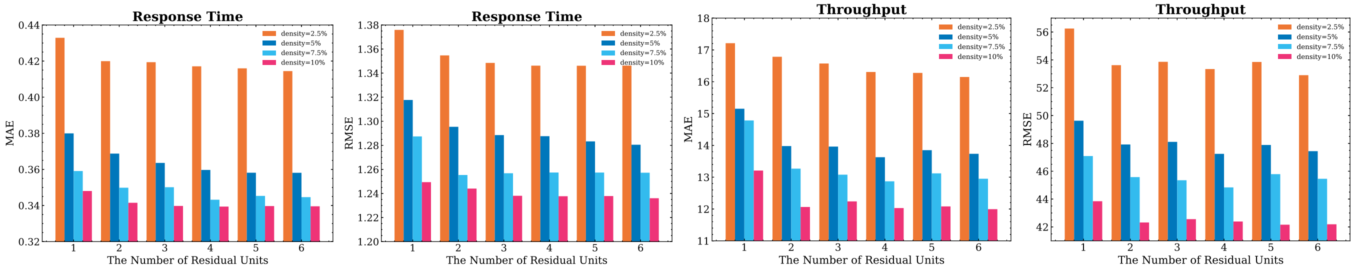
5.5.2 Impact of the Number of Residual Units

The number of Residual Units affects the depth of the latent feature extraction layer. Generally, the deeper the residual network is, the more nonlinear interactions among users and services it can learn for better QoS prediction performance. In NCRL, since each Residual Unit contains two nonlinear layers, the depth of the latent feature extraction layer is twice the number of Residual Units. In the experiments, we vary the number of Residual Units from 1 to 6, and correspondingly set the number of hidden neurons as 64, 128, 64, 64, 32 and 32, respectively.

Fig. 5 illustrates the performance impact of the number of Residual Units. It can be observed that both MAE and RMSE sharply decrease as the number of Residual Units varies from 1 to 2, and then slowly decline until the number of Residual Units arrives at 4. However, MAE and RMSE begin to fluctuate with slightly ascending trend after the number of Residual Units constantly increases from 4 to 6. It can be reasonably explained that when the number of Residual Units is too small, our designed two-tower deep residual network cannot reach a powerful learning capacity to effectively learn the implicitly complex nonlinear interaction among users and services, which significantly lowers QoS prediction accuracy. In another extreme case, if the number of Residual Units is set to be a large value, it is prone to the phenomenon of model training overfitting that may also incur the performance reduction of QoS prediction. Based on the above analyses, when the number of Residual Units is set to 4 in our experiments, two-tower deep residual network achieves the best QoS prediction accuracy under different QoS densities.

5.5.3 Impact of Weight Coefficient

NCRL adaptively predicts an known QoS by integrating the neural and collaborative predicted values with two weight



(a) MAE of RT (b) RMSE of RT (c) MAE of TP (d) RMSE of TP

Fig. 5. Performance impact of the number of Residual Units on NCRL under different QoS densities.

coefficients, which impacts the MAE and RMSE of QoS prediction. The parameters α and β in (21) adjust the adaptive correction of $\hat{r}_{u,s}$ and $\hat{r}'_{u,s}$. Due to the constraint between α and β (i.e., $\alpha + \beta = 1$), we only analyze the performance impact of α. It varies from 0 to 1 with a step size of 0.1, and QoS density is set from 0.025 to 0.1.

Fig. 6 illustrates the performance impact of weight coefficient. Apparently, when α is set to 1, NCRL completely degenerates into neural QoS prediction; when α is set to 0, it turns to be neighborhood-based collaborative QoS prediction. We can observe from Fig. 6 that NCRL can receive the lowest MAE and RMSE for the best QoS prediction accuracy, when α is set at a certain value between 0 and 1. It indicates that both neural QoS prediction and neighborhood-based collaborative prediction are beneficial to boosting the performance QoS prediction. However, it is still difficult to find an ubiquitous value of α that makes NCRL perform optimally across multiple QoS densities on both MAE and RMSE. For example, when QoS density is set to 5% for response time, NCRL performs optimally on MAE and RMSE with the settings of α = 0.9 and 0.5, respectively. Considering the performance of α on MAE and RMSE comprehensively, it relatively achieves superior QoS prediction accuracy, when α is set to 0.8 on RT and 0.7 on TP.

5.6 Prediction Efficiency

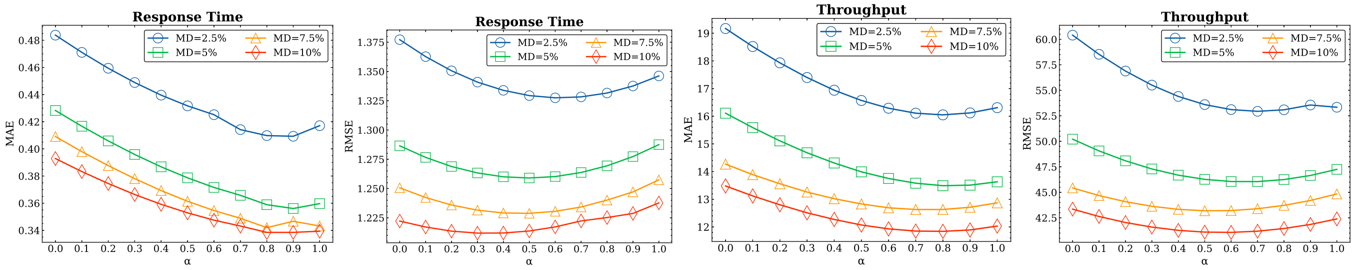
To verify the efficiency of our proposed NCRL, we conduct experiments to compare QoS prediction time with competing methods. PMF, NCF, LDCF and NDMF are chosen to compare with NCRL in the experiments. To guarantee the comparison fairness, all the competing methods are trained and equipped by the same hardware and software environments. In the experiments, 10,000 QoS samples are selected as test data and fed into five competing methods. They are run repeatedly for 20 rounds, and the duration of each round is recorded to QoS prediction.

Fig. 7 illustrates the prediction efficiency among competing methods. From the results, PMF achieves the best efficiency because it has the least parameters and makes QoS prediction by the dot product between feature vectors of a user and service. Subsequently, NCRL dominates other competing methods based on deep neural networks, while NDMF consumes the most time for QoS prediction. The reason is that NCRL stores the extracted latent features of users and services by offline learning at the training stage, which are used to both perform neural QoS prediction based on a shallow network and calculate similar users and services for neighborhood-based collaborative prediction. At the prediction stage, corresponding feature vectors can be efficiently retrieved for adaptive QoS prediction. Therefore, NCRL can obtain higher efficiency compared to state-of-the-art baselines based on deep neural networks.

6 Related Work

6.1 Memory-Based Approaches

This kind of QoS prediction approaches first performs similarity calculation to find a set of similar users or services, and then predicts the unknown QoS values by combining the average QoS and deviation migration based on historical QoS invocations. Shao et al. [9] proposed a user-based CF approach to predict QoS values by finding similar users with Pearson Correlation Coefficient. Chen et al. [33] proposed a service-based CF approach where similarity is calculated between services and their combination with the location of service provider to predict QoS values. Zheng et al. [7] proposed a hybrid CF approach called WSRec to predict QoS values, which combines the predicted QoS values by user-based CF and service-based CF with a weight coefficient. It achieves better prediction accuracy than previous memory-based approaches, since both the similarity of users and services are taken into account for predicting



(a) MAE of RT (b) RMSE of RT (c) MAE of TP (d) RMSE of TP

Fig. 6. Performance impact of α on NCRL under different QoS densities.

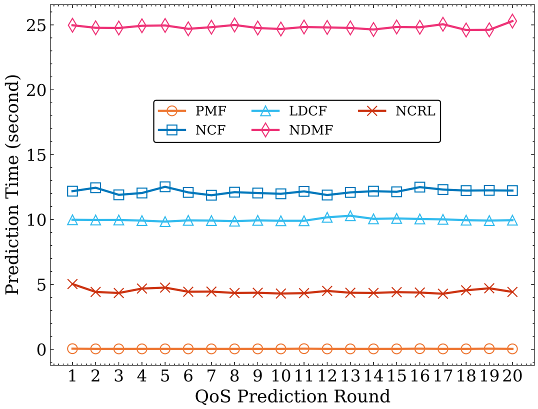


Fig. 7. Prediction efficiency among competing methods.

unknown QoS. To further improve the effectiveness of similarity, Sun et al. [34] proposed a new similarity measure for collaboratively predict QoS values, called normal recovery collaborative filtering. Wu et al. [5] proposed a ratio-based similarity approach to select neighborhoods of users and services, leading to better QoS prediction. IoTPredict [35] is a novel neighborhood-based QoS prediction approach for the IoT, which uses an alternative similarity calculation mechanism and achieves higher QoS prediction accuracy. Zou et al. [10] proposed a reinforced collaborative filtering approach, where noisy users or services are filtered out for average QoS and deviation migration, significantly improving the accuracy of QoS prediction.

Although many efforts of this kind of approaches have been made for QoS prediction, the performance is still unsatisfactory for predicting unknown QoS, because they are significantly influenced by the sparsity of historical user-service QoS invocations.

6.2 Model-Based Approaches

This kind of approaches tries to learn a model from historical user-service QoS invocations to predict QoS values. Matrix Factorization (MF) as the fundamentally applied techniques have been widely investigated for model-based QoS prediction. It can partially solve the issue of data sparsity and scalability when predicting vacant QoS values. Probabilistic Matrix Factorization (PMF) [13] as the basic approach utilizes probability model to optimize matrix factorization and improve the prediction accuracy. Based on PMF, Xu et al. [17] proposed a location-aware QoS prediction approach that integrates feature vectors of neighbors into probabilistic matrix factorization of latent user feature vectors based on their longitudes and latitudes. Zheng et al. [14] proposed an MF-based QoS prediction approach called NIMF that calculates users' neighborhood by PCC and integrate similar users into the matrix factorization, which effectively improves QoS prediction performance.

Yin et al. [36] demonstrated that QoS values such as response time and throughput are highly dependent on the performance of networks, where users located in the same location have similar invoked QoS values for the same service and the services located in the same location generally share the similar QoS values when invoked by the same user. Inspired by the above observation, they [16] proposed LoNMF, which first identifies a set of highly relevant local neighbors by a two-level selection mechanism and then integrates geographical information to build up an extended matrix factorization approach for personalized QoS prediction. Tang et al. [37] proposed a network-aware QoS prediction called NAMF that also integrates users' neighborhood information into matrix factorization to learn a linear interaction relationship. However, it measures the network distances between users with network map to calculate a user's similar neighborhood rather than historical user-service QoS invocations. In addition, Xu et al. [38] proposed a highly credible approach for predicting unknown QoS values called reputation-based matrix factorization (RMF), which quantifies the credibility of users based on their contributed QoS, and then integrates users' reputation into matrix factorization for achieving more accurate QoS prediction accuracy.

Recently, deep learning techniques have been widely applied to improve the recommendation quality and solve QoS prediction problems. NCF [32] as a neural collaborative filtering approach based on deep neural network leverages multi-layer perceptron to learn nonlinear interaction relationships, which can be applied for effective QoS prediction. Wu et al. [19] proposed a deep neural model (DNM) for making multiple attributes QoS prediction, where contextual features are mapped into a shared latent space and their high-order interactions are captured through multi-layer perceptron network. Zhang et al. [20] proposed a location-aware deep collaborative filtering (LDCF) approach by integrating MLP with a similarity adaptive corrector, extending the adaptability for QoS prediction. Zou et al. [21] proposed a novel approach for QoS prediction called neighborhood-integrated deep matrix factorization (NDMF), which firsts exploits both historical user-service QoS invocations and users' geographical information to find a target user's neighborhood, and then loosely integrates it into MLP for learning nonlinear interaction relationships among users and services. Wang et al. [23] proposed a novel location-aware feature interaction learning (LAFIL) approach for predicting QoS values, which can effectively solve the issues on data sparsity and cold-start by leveraging location features of both users and services. Xia et al. [39] proposed a QoS prediction approach called JDNMFL that builds CNN-based joint deep networks to learn multi-source feature interaction. Compared to matrix factorization based QoS prediction approaches, it has obvious advantage that these approaches based on deep neural networks can more effectively learn the implicitly complex nonlinear interactive relationships among users and services. Wang et al. [40] proposed a hidden-state aware network named HSA-Net that is mainly based on the idea of initializing hidden-state information of users and services. It can overcome the challenge of independent features and enhance the compatibility to different datasets.

Most of the existing approaches have exploited shallow deep neural network such as MLP, where collaborative relationships of similar neighborhood have not been fully taken into account. In contrast to these approaches, NCRL addresses the problem by combining the advantages of neural prediction and collaborative prediction together to perform adaptive QoS prediction. We leverages a designed two-tower deep residual network to separately extract users' and services' latent features, which are used to make neural prediction and calculate similar neighborhoods to perform collaborative prediction. Therefore, our proposed NCRL can significantly improve the performance of QoS prediction.

7 Conclusion and Future Work

In this paper, we propose a novel framework called NCRL, which aims at improving the performance of QoS prediction in an adaptive way. First, we proposed a location-aware two-tower deep neural network to learn the implicitly complex user-service nonlinear interactive relationships for neural QoS prediction, where multi-source location information are taken into account to extract the latent features of users and services. Then, we detect similar users and services based on the extracted latent feature vectors for neighborhood-based collaborative prediction, where user-service historical QoS invocations are taken to calculate the missing QoS. Finally, we adaptively perform QoS prediction by combining neural and collaborative predicted QoS with weight coefficients. Extensive experiments are conducted and the results demonstrate that NCRL achieves the best performance in terms of effectiveness and efficiency compared with state-of-the-art competing baselines. In the future, we plan to further explore advanced neural networks to improve the QoS prediction accuracy with the consideration of temporal characteristics on user-service invocations in edge computing paradigm.

References

[1] A. Bouguettaya et al., “A service computing manifesto: The next 10 years,” *Commun. ACM*, vol. 60, no. 4, pp. 64–72, 2017.

[2] M. N. Huhns and M. P. Singh, “Service-oriented computing: Key concepts and principles,” *IEEE Internet Comput.*, vol. 9, no. 1, pp. 75–81, Jan./Feb. 2005.

[3] S. H. Ghafouri, S. M. Hashemi, and P. C. K. Hung, “A survey on web service QoS prediction methods,” *IEEE Trans. Services Comput.*, vol. 15, no. 4, pp. 2439–2454, Jul./Aug. 2022.

[4] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “GroupLens: An open architecture for collaborative filtering of netnews,” in *Proc. ACM Conf. Comput. Supported Cooperative Work*, 1994, pp. 175–186.

[5] X. Wu, B. Cheng, and J. Chen, “Collaborative filtering service recommendation based on a novel similarity computation method,” *IEEE Trans. Services Comput.*, vol. 10, no. 3, pp. 352–365, May/Jun. 2017.

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

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

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

[9] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, “Personalized QoS prediction for web services via collaborative filtering,” in *Proc. IEEE Int. Conf. Web Serv.*, 2007, pp. 439–446.

[10] G. Zou, M. Jiang, S. Niu, H. Wu, S. Pang, and Y. Gan, “QoS-aware web service recommendation with reinforced collaborative filtering,” in *Proc. Int. Conf. Service-Oriented Comput.*, 2018, pp. 430–445.

[11] 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 Serv.*, 2010, pp. 9–16.

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

[13] A. Mnih and R. R. Salakhutdinov, “Probabilistic matrix factorization,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.

[14] Z. Zheng, H. Ma, M. R. Lyu, and I. King, “Collaborative web service QoS prediction via neighborhood integrated matrix factorization,” *IEEE Trans. Services Comput.*, vol. 6, no. 3, pp. 289–299, Third Quarter 2013.

[15] L. Ren and W. Wang, “An SVM-based collaborative filtering approach for Top-N web services recommendation,” *Future Gener. Comput. Syst.*, vol. 78, pp. 531–543, 2018.

[16] W. Lo, J. Yin, Y. Li, and Z. Wu, “Efficient web service QoS prediction using local neighborhood matrix factorization,” *Eng. Appl. Artif. Intell.*, vol. 38, pp. 14–23, 2015.

[17] Y. Xu, J. Yin, W. Lo, and Z. Wu, “Personalized location-aware QoS prediction for web services using probabilistic matrix factorization,” in *Proc. Int. Conf. Web Inf. Syst. Eng.*, 2013, pp. 229–242.

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

[19] H. Wu, Z. Zhang, J. Luo, K. Yue, and C.-H. Hsu, “Multiple attributes QoS prediction via deep neural model with contexts,” *IEEE Trans. Services Comput.*, vol. 14, no. 4, pp. 1084–1096, Jul./Aug. 2021.

[20] Y. Zhang, C. Yin, Q. Wu, Q. He, and H. Zhu, “Location-aware deep collaborative filtering for service recommendation,” *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 51, no. 6, pp. 3796–3807, Jun. 2021.

[21] G. Zou, J. Chen, Q. He, K.-C. Li, B. Zhang, and Y. Gan, “NDMF: Neighborhood-integrated deep matrix factorization for service QoS prediction,” *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 4, pp. 2717–2730, Dec. 2020.

[22] Y. Zhang et al., “Covering-based web service quality prediction via neighborhood-aware matrix factorization,” *IEEE Trans. Services Comput.*, vol. 14, no. 5, pp. 1333–1344, Sep./Oct. 2021.

[23] Z. Wang, Y. Xiao, C. Sun, W. Zheng, and X. Jiao, “Location-aware feature interaction learning for web service recommendation,” in *Proc. IEEE Int. Conf. Web Serv.*, 2020, pp. 232–239.

[24] Y. Shan, T. R. Hoens, J. Jiao, H. Wang, D. Yu, and J. Mao, “Deep crossing: Web-scale modeling without manually crafted combinatorial features,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 255–262.

[25] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[26] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[27] D. Hendrycks and K. Gimpel, “Gaussian error linear units (GELUs),” 2016, *arXiv:1606.08415*.

[28] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 630–645.

[29] Q. Zhao, D. Meng, Z. Xu, W. Zuo, and Y. Yan, “L_1-norm low-rank matrix factorization by variational Bayesian method,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 4, pp. 825–839, Apr. 2015.

[30] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, *arXiv:1412.6980*.

[31] S. Rendle, “Factorization machines with libFM,” *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 3, pp. 1–22, 2012.

[32] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *Proc. Int. World Wide Web Conf.*, 2017, pp. 173–182.

[33] Z. Chen, L. Shen, and F. Li, “Exploiting web service geographical neighborhood for collaborative QoS prediction,” *Future Gener. Comput. Syst.*, vol. 68, pp. 248–259, 2017.

[34] H. Sun, Z. Zheng, J. Chen, and M. R. Lyu, “Personalized web service recommendation via normal recovery collaborative filtering,” *IEEE Trans. Services Comput.*, vol. 6, no. 4, pp. 573–579, Fourth Quarter 2013.

[35] G. White, A. Palade, C. Cabrera, and S. Clarke, “IoTPredict: Collaborative QoS prediction in IoT,” in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2018, pp. 1–10.

[36] J. Yin, W. Lo, S. Deng, Y. Li, Z. Wu, and N. Xiong, “Colbar: A collaborative location-based regularization framework for QoS prediction,” *Inf. Sci.*, vol. 265, pp. 68–84, 2014.

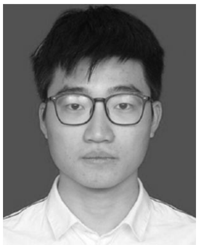
[37] M. Tang, Z. Zheng, G. Kang, J. Liu, Y. Yang, and T. Zhang, “Collaborative web service quality prediction via exploiting matrix factorization and network map,” *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 1, pp. 126–137, Mar. 2016.

[38] J. Xu, Z. Zheng, and M. R. Lyu, "Web service personalized quality of service prediction via reputation-based matrix factorization," *IEEE Trans. Rel.*, vol. 65, no. 1, pp. 28–37, Mar. 2016.

[39] Y. Xia, D. Ding, Z. Chang, and F. Li, "Joint deep networks based multi-source feature learning for QoS prediction," *IEEE Trans. Services Comput.*, vol. 15, no. 4, pp. 2314–2327, Jul./Aug. 2022.

[40] Z. Wang, X. Zhang, M. Yan, L. Xu, and D. Yang, "HSA-Net: Hidden-state-aware networks for high-precision QoS prediction," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 6, pp. 1421–1435, Jun. 2022.

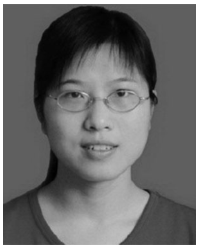
Guobing Zou received the PhD degree in computer science from Tongji University, Shanghai, China, in 2012. He is an associate professor and dean with the Department of Computer Science and Technology, Shanghai University, China. He has worked as a visiting scholar with the Department of Computer Science and Engineering, Washington University in St. Louis from 2009 to 2011, USA. His current research interests mainly focus on services computing, edge computing, data mining and intelligent algorithms, recommender systems. He has published more than 90 papers on premier international journals and conferences, including the *IEEE Transactions on Services Computing*, *IEEE Transactions on Network and Service Management*, IEEE International Conference on Web Services, International Conference on Service-Oriented Computing, etc.



Shaogang Wu received the bachelor's degree in computer science and technology from Shanghai University, China in 2020. He is currently working toward the master degree in the School of Computer Engineering and Science, Shanghai University. His research interests include service quality management, deep learning, and intelligent algorithms. He has published a paper on International Conference on Service-Oriented Computing (ICSOC 2022). He has led research and development group to successfully design and implement a service-oriented enterprise application platform, which can intelligently classify and recycle, cultivate citizens' habit of throwing recyclables, and produce significant economic and social benefits by providing high QoS.

Shengxiang Hu received the bachelor's and master's degrees in computer science and technology from Shanghai University, China in 2018 and 2021, respectively. He is currently working toward the PhD degree in the School of Computer Engineering and Science, Shanghai University. His research interests include QoS prediction, graph neural network, and natural language processing. He has published two papers on the *Knowledge-Based Systems*, International Conference on Parallel Problem Solving from Nature (PPSN), and published a paper on International Conference on Service-Oriented Computing (ICSOC 2022).

Chenhong Cao received the BS and MS degrees in computer science from Northeastern University, China, in 2011 and 2013, respectively, and the PhD degree from the College of Computer Science and Technology, Zhejiang University, in 2018. She is currently working as an assistant professor with the School of Computer Engineering and Science, Shanghai University, China. Her research interests mainly focus on the network and system of the Internet of Things, network measurement and security, and wireless sensing. She has published more than 10 papers on premier international journals and conferences, including the *IEEE Transactions on Mobile Computing*, IEEE INFOCOM, *Computer Networks*, and *IEEE/ACM Transactions on Networking*.



Yanglan Gan received the PhD degree in computer science from Tongji University, Shanghai, China, in 2012. She is a full professor with the School of Computer Science and Technology, Donghua University, Shanghai, China. Her research interests include bioinformatics, service computing, and data mining. She has published more than 50 papers on premier international journals and conferences, including the *Bioinformatics*, *BMC Bioinformatics*, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *IEEE Transactions on Services Computing*, *IEEE Transactions on Network and Service Management*, IEEE International Conference on Web Services, International Conference on Service-Oriented Computing, *Neurocomputing*, and *Knowledge-Based Systems*.

Bofeng Zhang received the PhD degree from the Northwestern Polytechnic University (NPU), China, in 1997. He is a full professor and dean with the School of Computer and Information Engineering, Shanghai Polytechnic University, Shanghai, China. He experienced a postdoctoral research with Zhejiang University from 1997 to 1999, China. He worked as a visiting professor with the University of Aizu from 2006 to 2007, Japan. He worked as a visiting scholar with Purdue University from 2013 to 2014, US. His research interests include personalized service recommendation, intelligent human-computer interaction, and data mining. He has published more than 200 papers on international journals and conferences.

Yixin Chen received the PhD degree in computer science from the University of Illinois at Urbana Champaign, in 2005. He is currently a full professor of computer science with Washington University in St. Louis, MO, USA. His research interests include artificial intelligence, data mining, deep learning, and big data analytics. He has published more than 100 papers on premier international journals and conferences, including the *Artificial Intelligence*, *International Journal of Artificial Intelligence Research*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Services Computing*, *IEEE Transactions on Computers*, *IEEE Transactions on Industrial Informatics*, IJCAI, AAAI, ICML, KDD, etc. He won the Best Paper Award at AAAI and a best paper nomination at KDD.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.