Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

QoS prediction in intelligent edge computing based on feature learning

Hongxia Zhang[1], Dengyue Wang[1], Wei Zhang[2*], Lizhuang Tan[2], Godfrey Kibalya[3], Peiying Zhang[1,4] and Kostromitin Konstantin Igorevich[5]

Abstract

With the development of 5G and 6G, more computing and network resources on edge nodes are deployed close to the terminal. Meanwhile, the number of smart devices and intelligent services has grown significantly, which makes it difficult for users to choose a suitable service. The rich contextual information plays an important role in the prediction of service quality. In this paper, we propose a quality of service(QoS) prediction approach based on feature learning, the contextual information represented as the explicit features and underlying relationship hidden in the implicit features are fully considered. Then, the multi-head self-attention mechanism is used in the interacting layer to determine which features should be combined to form meaningful high-order features interaction. We have implemented our proposed approach with experiments based on real-world datasets. Experimental results show that our approach achieved a better performance of service QoS prediction in an intelligent edge computing environment for future communication.

Keywords Intelligent edge computing, Service recommendations, QoS prediction, multi-head self-attention, feature learning

Introduction

The rise of 5G and future 6G technology will contribute a great deal to the construction of future communication networks. Meanwhile, the number of network terminal users and services has grown significantly, this presents a challenge for users to select services that match users' requirements with different devices. Intelligent Edge Computing(IEC) supports more nodes to load traffic and increase the communication rate. The European Telecommunication Standards Institute (ETSI) defines this emerging paradigm as Multi-access edge computing [1] which put computing power closer to the edge network, so as to enable the intelligent devices to invoke services through different network modes to meet the needs of low latency.

On the other hand, the number of intelligent services supported by Artificial intelligence (AI) technology provided by the edge nodes is getting enormous, more accurate intelligent services prediction model is needed. Quality of Service (QoS) has been a critical criterion in measuring the suitability of a service for a user. Thus, the prediction of QoS becomes a key issue in service recommendation [2]. In the future communication scenario [3], QoS is susceptible to the heterogeneous devices of users and the complexity of network environment [4], users can invoke services from a variety of smart devices such as wearable equipment, phones, tablets and laptops.

*Correspondence:
Wei Zhang
wzhang@sdas.org
[1] College of Computer Science and Technology, China University of Petroleum (East China), Qingdao, China
[2] Shandong Provincial Key Laboratory of Computer Networks, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan, China
[3] Department of Computer Engineering and Informatics, Busitema University, Tororo, Uganda
[4] State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China
[5] Department of Physics of Nanoscale Systems, South Ural State University, Chelyabinsk, Russia

Different devices [5] have different QoS perception for service, the QoS value may be entirely different even for the same service. This requires that the prediction model needs to be sensitive to the various devices. In addition, network access mode the of device also has a significant impact on QoS value. Users will get different QoS due to different network mode [6, 7] they accessed. Therefore, the prediction system needs to be aware of the access of the network mode. Besides, other contextual factors also affect QoS to a certain extent, such as the geo-location, IP address, ID number etc.

To make accurate QoS prediction, different efforts have been taken. Generally, researches on QoS prediction can be classified into Nearest Neighborhood(NN), Matrix Factorization(MF) and Deep Learning(DL) methods. The NN methods [8–12] are based on the assumption that if users had similar QoS experience in the past, they will have similar QoS experience in the future. However, NN methods only use neighborhood's information to make prediction, and ignore the beneficial information hidden in the whole user-service QoS matrix. Different from NN methods, MF methods [13–17] provide QoS prediction with user and service implicit feature matrices that are learned by using the whole available data in QoS matrix. Nevertheless, the performance of MF is limited by the simple linear dot product of user latent feature vector and service latent feature vector. Recent years have witnessed a great development in deep learning for service QoS prediction, Factorization Machine(FM) [18, 19] and multi-layer perceptron(MLP) [20–25] based deep learning methods provide unprecedented opportunities to advance service QoS prediction. AI technology in the direction of IEC has improved prediction accuracy. However, building more accurate QoS predictors still faces challenges.

1) The types of devices and network access modes in the IEC environment are constantly enriched. Meanwhile QoS is susceptible to heterogeneous devices and the complexity of the network environment. Existing researches [20–23] demonstrated that the QoS value is context-dependent. Still, the contextual information is not fully exploited and utilized effectively in the interaction between the user and the service.
2) Approaches based on feature learning [26, 27] typically focus only on the interaction of explicit features. However, the implicit features are of vital in QoS prediction, which is usually ignored by these methods in their interaction layer. Considering both explicit and implicit features will undoubtedly improve the prediction accuracy of QoS in IEC.
3) Currently, methods based on deep learning do not distinguish the importance of different feature interactions. Literatures [18, 19, 22] often integrate FM for QoS prediction, because FM works well under significant sparsity case through feature interaction learning. However, the existing methods assign the same weight to all feature interactions, while the identification and process of low correlation feature interactions are ignored, seriously affecting the accuracy of QoS prediction.

To alleviate these critical challenges, we proposed an approach named Matrix Factorization Automatic Interaction Network (MFAIN) to automatically learn high-order feature interactions. In our approach, we take the context as explicit features such as geo-location, IP address, ID number etc. To make use of the information related to devices, we take the device types and network modes obtained by the clustering algorithm as explicit features too. Meanwhile, the implicit features which contained latent relations are extracted through the MF. Then we utilize all these features generated vectors to feed the interaction network based on a multi-head self-attention mechanism to learn the low-order and high-order feature interactions. The attention mechanism is used to measure the correlations between features, and determine which features should be combined to form meaningful high-order features. Experimental results demonstrate that our model performs superior in prediction accuracy, compared with the state-of-the-art method, our approach achieves 11.1% and 11.6% improvement of average MAE over the best baseline for RT and TP respectively. Beyond that our model has a good robustness and extensibility on exploiting heterogeneous contextual features.

The main contributions of this work are summarized as follows:

1. The device awareness capability of the prediction model in IEC is improved by taking the characteristics of user devices into account. In order to reduce the vulnerability of QoS to the heterogeneity of user devices and the complexity of the network environment, the explicit features of devices are obtained by the clustering algorithm through the QoS matrix.
2. We use the matrix factorization to rich context implicit features, which are usually ignored by the existing feature learning methods in their interaction layer. The interaction of rich explicit and implicit features greatly improves the scalability of QoS prediction models in IEC.
3. We propose a novel approach named Matrix Factorization Automatic Interaction Network (MFAIN), which combined matrix factorization and feature interactive networks to predict the QoS in the IEC environment. A multi-head self-attention mechanism is used to automatically learning the low-order

and high-order feature interactions, which allows each feature to interact with the others and to determine the relevance through learning.

The remainder of this paper is organized as follows: Section Related work shows our related work, and Section Motivation introduces a motivation scenario. Section Our approach presents our service recommendation approach based on MF and feature learning. Section Experiment and evaluation describes the experiment and evaluation. Section Conclusion and future work concludes the paper and discusses the future works.

Related work

According to the existing researches on QoS prediction, we classify works into traditional NN, MF and DL methods.

The traditional method is NN-based QoS prediction, which has the advantages of easy implementation in a traditional service computing environment. The core of this method is to calculate similarity and then conduct collaborative filtering. Shao et al. [8] proposed the use of a collaborative prediction method based on the similarity [28] of users calculated by Pearson correlation coefficient (PCC). Zheng et al. [9] proposed a mixed model that integrated user-based and item-based approaches linearly by confidence weight. Sun et al. [10] consider the distribution characteristics of QoS data to calculate the similarity. However, note that in real scenario, a single user cannot invoke all services which leads to a problem known as a data sparsity issue. In that case, the neighbor of users and services can not be selected.To alleviate the impact of data sparsity, Liu et al. [11] proposed location-aware similarity metrics to find neighbors of users and services with the help of location information. Wang et al. [12] proposed a service recommendation approach through calculates user or edge server similarity. Even so, the NN methods still ignored the useful information that hidden in the whole data, while our proposed MFAIN approach aims to utilize the whole data to train and improve the accuracy of QoS prediction.

MF is another line of collaborative filtering method. MF is different from above methods that only uses neighborhood information for QoS prediction, but uses all available QoS data to learn the implicit feature representations of users and services. Although the existing MF methods [13–16] try to integrate location, similarity relationship and contextual information into MF, its performance is still limited by the simple linear inner product between user latent feature vector and service latent feature vector. As the amplification of context information in the IEC environment, more non-linear interactions between the user and service need to be captured.

Recently, some exploratory works have been based on DL models for QoS prediction. FM is an effective technique to tackle the problem of combining features in sparse data. Wu et al. [18] introduced FM and used the second-order feature interaction to model the real interaction between the user and the service. Yang et al. [19] regarded the neighborhood of user and service as a supplementary feature, and put them to FM for interactive learning. However, low-order feature interactions are not sufficient to fit complex feature rela-tionships. Considering that real interaction between user and service is complex and no-linear, Zhang et al. [20] then input user, service, and location information to a MLP after one-hot encoding, and implemented the nonlinear transformation of the complex interaction between user and service using ReLU function. To improve the prediction accuracy by using the user neighborhood information, Gao et al. [21] improved the embedding layer of MLP by fusing neighbors' latent cluster feature for the user and service. Shen et al. [22] combined the advantages of FM and MLP through a well-designed left and right structure, and used contextual information to improve the accuracy of QoS prediction. In addition to utilizing the contextual information, more researchers use implicit information to improve prediction accuracy. Wang et al. [23] proposed a location-aware feature interaction learning (LAFIL) method for predicting the QoS values of the user-service matrix and then making recommendation by learning the underlying relation, which is hidden in the features concerning location information. Considering the advantages of convolutional neural network (CNN), Yin et al. [24] proposed a joint MF and CNN model, which takes full advantage of neighborhood features, common implicit features and deep implicit features of the service invocation process. The proposed model tackled the problems of how to improve neighbor selection quality and how to learn deep implicit features from QoS records. Xia et al. [25] proposed JDNMFL, which builds a CNN-based joint deep network to learn local critical feature interaction, and MLP is applied to learn global feature interactions from raw features and local feature interactions for QoS prediction. Although many context factors are considered in the existing methods, when facing the IEC environment target to service the devices, the increment of equipment types and quantities make the information of device play an increasingly important role in the prediction, which has not been fully considered.

Meanwhile, the above DL based methods do not pay much attention to the effect of higher order interaction of features. To enrich the interacting manner between features of users and service, Chen et al. [26] proposed a context-aware feature interaction modeling (CFM) to capture both memorization and generalization by

jointly considering low-order feature interactions with FM and high-order feature interactions with a MLP and deep cross network(DCN). Wang et al. [23] proposed LAFIL leverage Compressed Interaction Network (CIN) for feature interaction learning of the location information only. Wu et al. [27] proposed a universal deep neural model(DNM) for QoS prediction with contexts, which respectively learning the features interaction from user-side and service-side, but the features interaction between the user and the service are ignored. Unfortunately, existing feature interaction learning methods [23, 26, 27] based on DL do not distinguish the importance of different feature interactions. They assign the same weight to all feature interactions, while the identification and process of low correlation feature interactions are ignored, seriously affecting the accuracy of QoS prediction.

Compared to the existing works, our work takes the characteristics of user devices into account in the IEC environment. And use the matrix factorization to rich context implicit features, which are usually ignored by the existing feature learning methods in their interaction layer. Finally, a multi-head self-attention mechanism is used to automatically learn the low-order and high-order feature interactions, which allows each feature to interact with the others and to determine the relevance through learning.

Motivation

An IEC environment consists of multiple mobile edge servers, which close to the base station and provide network access for mobile users in a certain range.

Suppose $ES = \{es_1, es_2, \cdots\}$ denotes the edge server banded with the base station, and $NM = \{nm_1, nm_2, \cdots\}$ denotes the network mode which base station used such as 4G, 5G, WIFI and so on. Suppose $U = \{u_1 \cdots u_i \cdots u_m\}$, $S = \{s_1 \cdots s_j \cdots s_n\}$ is a set of m users and n services respectively. And $D = \{d_1, d_2, \cdots d_K\}$ denotes the type of user devices for example wearable device, smartphone, and personal computer. Users can invoke services via different devices in D through ES in the way of access NM.

As depicted in Fig. 1, Consider a real-world service invoke scenario which start from the upper left corner of the Fig. 1. A user named Tony uses his smartphone to invoke the service $s_j \in S$ through the edge server es_1 access to 5G on a bus and obtained $QoSj$. After a period, Tony arrives at the bus station and changes to ride. Meanwhile Tony changes to use his wearable watch through es_1 in order to keep using the service s_j. But the QoS will change due to device change, because of a higher resolution ratio requested for the mobile phone than the wearable watch. As Tony moves, the edge server which he accessed alter from es_1 to es_2 and the network mode change from 5G to 4G. The QoS will

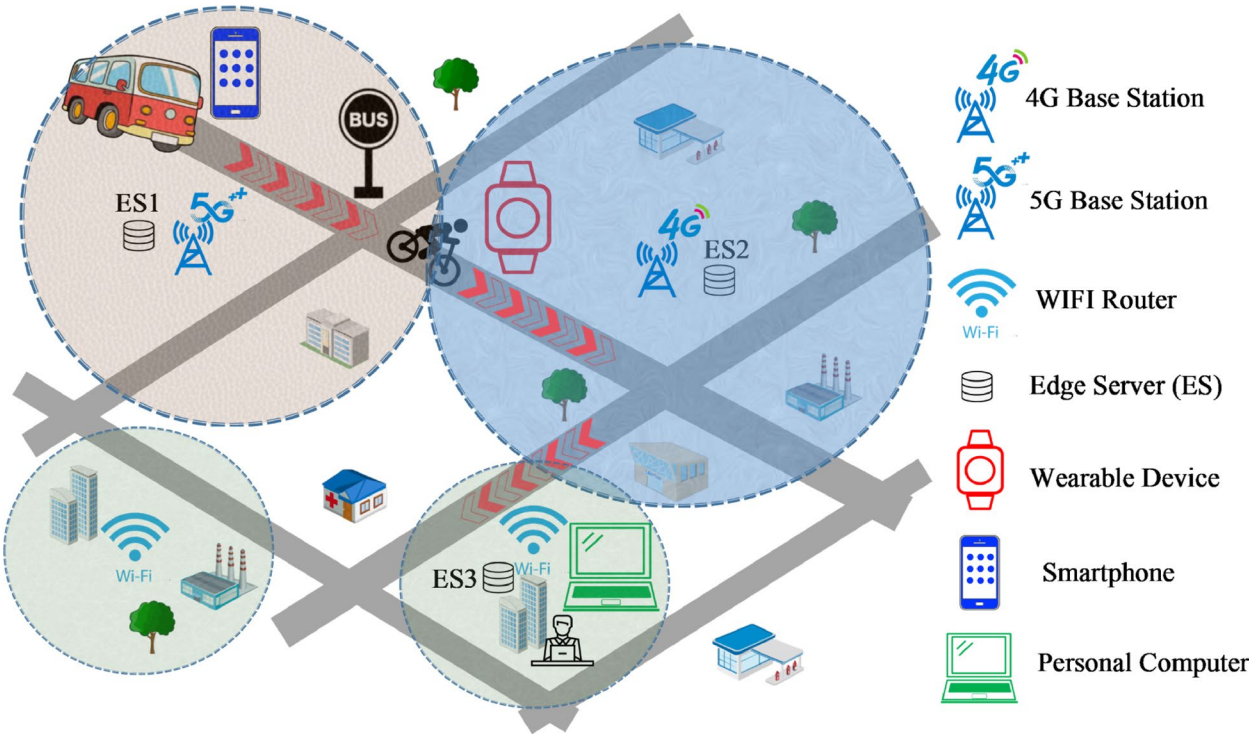

Fig. 1 Motivation scenario in intelligent edge computing environment

also change due to the volatility of the network environment. Finally, Tony arrives to his destination and start to use a personal computer to still invoke the service s_j through es_3 access to a WIFI network, and the QoS in this moment has already fluctuated at least two times. In this instance, the user device changes arbitrarily, and the network mode alternate frequently. To make the prediction model more aware of the context and predict the QoS value more accurately, the features of the device type and network mode should be fully considered in such an IEC scene. All notations used in this paper are summarized in Table 1.

Table 1 The summary of notations

Notation	Description
$es \in ES$	the edge sever
$nm \in NM$	the network mode
$u_i \in U$	a user, $i \in \{1, \cdots, m\}$
$s_j \in S$	a service, $j \in \{1, \cdots, n\}$
m	the number of users
n	the number of services
K	the number of user device types
s	the number of field(explicit feature)
d	interaction layer input vector dimension
$em \in EM$	explicit embedding vector
$Q \in R^{m \times n}$	QoS matrix
$U_{IM} \in R^{m \times d}$	user implicit feature matrix
$S_{IM} \in R^{n \times d}$	sevice implicit feature matrix
λ_{reg}	regularization term
U_{IM_i}	implicit feature of user i
S_{IM_j}	implicit feature of service j
$e_f \in e$	feature used to interact
$\alpha_{f,k}^{(h)}$	attention weights between feature f and k
$\tilde{e}_f^{(h)}$	a new combinatorial feature under head h
H	the number of head

Our approach

In this section, we will illustrate the architecture of Matrix Factorization Automatic Interaction Network (MFAIN) as shown in Fig. 2. First, we combine the original user-service feature collected from different users invoking various web services and the device feature extracted from the user-service invocation records by clustering algorithm into explicit features. These explicit features are transformed into embedding vectors through encoding and embedding operations. Then we extract the implicit feature through the MF based on the QoS invoke matrix and obtain the latent vector. Finally, feature interaction network is leveraged for the feature learning between the explicit embedding vector and the latent vector. Then, the QoS value will be predicted. This section can be divided into three parts. Subsection Explicit Feature Embedding shows explicit feature embedding, Subsection Implicit Feature Extraction describes the process of the implicit feature extraction through MF,

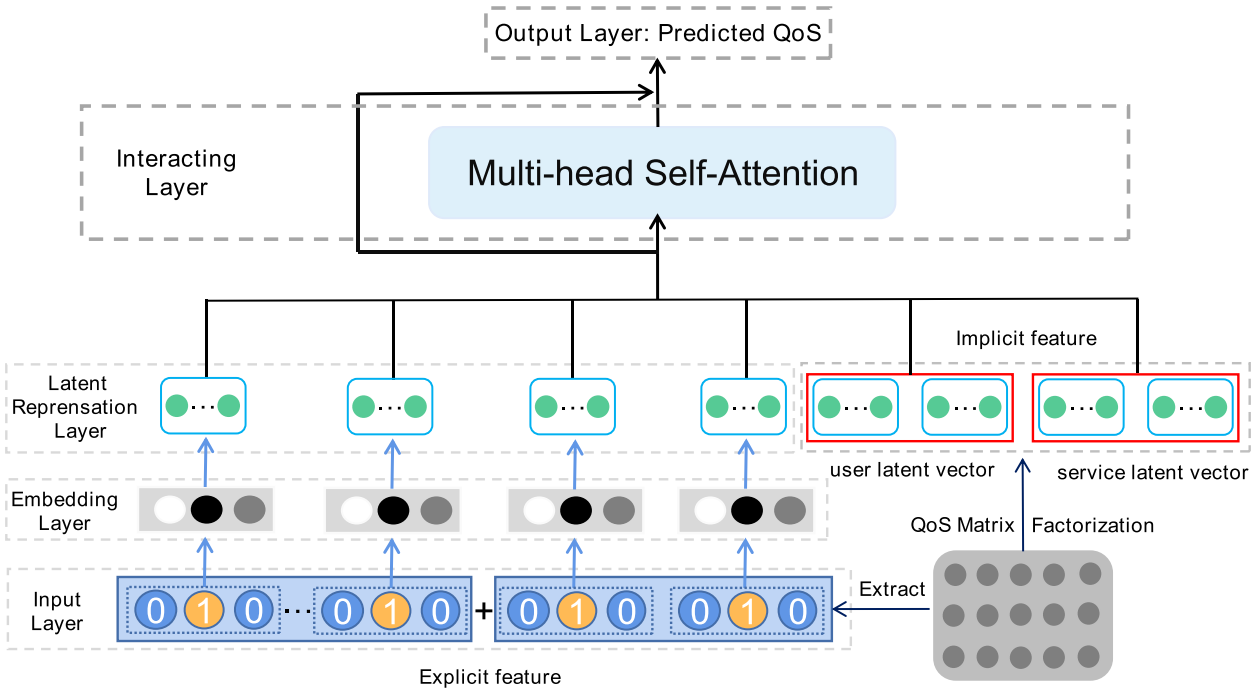

Fig. 2 The architecture of Matrix Factorization Automatic Interaction Network (MFAIN)

Subsection Feature Interaction Learning shows the feature interaction learning for QoS prediction.

Explicit Feature Embedding

Explicit Feature Encoding

To consider various contextual features in the IEC environment, we take the features associated with a service invocation in the user side and service side as the explicit features. In two side, we take 12 explicit features into account, which contain the user-id, the service-id, the geo-location of user/service, the IP address of user/service, the autonomous system of user/service, the IPNo of user/service obtained directly from user and service information in original dataset. We get the types of user devices and the user accessed network modes by cluster algorithm on TP and RT matrix respectively.

Considering the multiple access of equipment in the complex IEC environment, it is necessary to consider the features of the user device. In order to expand the device features, we found that the QoS of different users for the same service have clustering characteristics, as shown in Table 2 for some users in the dataset, it can be assumed that the same type of users utilized the same type of equipment and network access mode. In this case, we use the K-means clustering algorithm to obtain two realistic characteristics of users' device type and accessed network method based on service quality. K-means clustering is one of the simplest and popular unsupervised machine learning algorithms, it identifies k number of centroids and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. First, we cluster all users who invoke the same each service, and then average the clustering results for all services. Since the type of device is more related to throughput, and the network access mode of the device is more related to the response time. We performed the above operations on RT and TP datasets to obtain these two explicit device features respectively.

Feature transformation is critical in many classification or regression predictive tasks because using raw features alone rarely leads to optimal prediction results. In order to achieve the best results, raw feature transformation usually necessitates a significant amount of work. Here we use the one-hot encoding to preprocess the explicit features.

The concept of the field was firstly proposed in click-through rate (CTR) prediction [29]. The fields have two types of data. One type is discrete data or categorical data and the other is continuous data [30]. In this paper, we take all fields as the categorical tabledata, and formalize them into one-hot encoding which is a high-dimensional zero vector with a specified dimension that is set to be one. Suppose that if there are three users, the dimensionality of one-hot vector is set to be three. Then the user whose ID is 1 can be represented as [1,0,0], and the user with ID 2, 3 can be represented as [0,1,0] and [0,0,1] respectively. In our approach, we generate 18 fields from 12 explicit features where the redundant fields are generated from the IP Address of user and service separated by the symbol ".

As shown in Table 3. We generate the user IP Address into four fields separate by the symbol ". And the same operate to the service IP Address. There are two benefits of this operation. On the one hand, considering the same local field in IP Address may share the same geographic information. On the other hand, more features mean more feature interactions in the interacting layer.

Table 2 Service quality of some users with different devices

User-id	RT	network modes	User-id	TP	user devices
26	0.647	5G	26	20.092	laptops
34	0.604		34	21.523	
219	0.745	4G	219	17.449	tablets
274	0.749		274	17.356	
13	1.264	WIFI	13	10.284	phones
17	1.254		17	10.366	
111	6.415	3G	111	1.879	wearable equipment
144	6.475		144	2.007	

Embedding

It's obvious that this type of data form as one-hot encoding is sparse and high dimensional, directly leading to increase extra memory consumption and reduce the training efficiency of model because the most encoding values are 0 in interacting layer. An embedding layer is one sort of mapping transformation, applied on the raw feature input to compress it into a dense, low dimensional real-value vector before further feeding into the first hidden layer.

The embedding layer is illustrated concretely in Fig. 3. The output of embedding layer is a fixed size vector as follows:

Table 3 One-hot Encoding for Fields

User-id	...	UserIP[1]	UserIP[2]	UserIP[3]	UserIP[4]	...
2	...	128	10	19	52	...
01...0	...	0...10...	0...10...	0...10...	0...10...	...

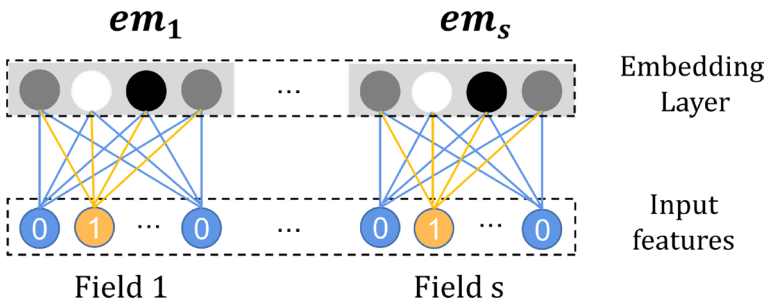

Fig. 3 Illustration of input and embedding layer, where convert the explicit features into embedding vector

$$EM = [em_1, em_2, em_3 \cdots em_s] \tag{1}$$

where s represents the field number, $em_i \in R^d$ denotes the embedding of i-th field and d is the number of dimension. The length of one-hot encoding raw feature input is various since different fields have different number of features, but their embedding vector is of the same fixed length d.

Implicit Feature Extraction

In order to obtain the implicit feature of user and service. Here, we use the MF to dig the latent relation between users and services. Concretely, MF is to map both users and services into a joint latent factor space, such that values of the user-service QoS matrix can be captured as inner products of latent factors in that space. Then the latent factors can be employed as the implicit features for further learning in the interaction layer.

As shown in Fig. 4, MF is to decompose the user-service QoS matrix $Q \in R^{m \times n}$ into user implicit feature matrix and service implicit feature matrix, which are denoted as $U_{IM} \in R^{m \times d}$ and $S_{IM} \in R^{n \times d}$ where d denotes the dimension of implicit feature. The objective is to fit the product of the two matrices to Q.

$$Q \approx U_{IM} S_{IM}^T \tag{2}$$

To fit the matrix Q by the product of U_{IM} and S_{IM}, a loss function L is built to minimize the errors.

$$L = min\phi(Q, U_{IM} S_{IM}^T) + \lambda_{reg} \tag{3}$$

Where $\phi(Q, U_{IM} S_{IM}^T)$ measures the degree of approximation between Q and $U_{IM} S_{IM}^T$, λ_{reg} denotes the regularization term to avoid over-fitting. Here we utilize the Cauchy loss [31] as the measurement of the discrepancy between the observed QoS values and the product of two implicit feature matrix, because it is more robust to outliers. Cauchy loss is shown as follows

$$ln(1 + \frac{(x - \hat{x})^2}{\gamma^2}) \tag{4}$$

where γ is constant. So the specific objective function can be clearly given as follows:

$$L = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} ln\left(1 + \frac{\left(R_{ij} - U_{IM_i} S_{IM_j}^T\right)^2}{\gamma^2}\right) + \lambda_{reg} \tag{5}$$

Then, we use the Stochastic gradient descent (SGD) [32] and back propagation to update $U_{IM}$and S_{IM}:

$$U_{IM_i} \leftarrow U_{IM_i} - \eta u \frac{\partial L}{\partial U_{IM_i}} \tag{6}$$

$$S_{IM_j} \leftarrow S_{IM_j} - \eta s \frac{\partial L}{\partial S_{IM_j}} \tag{7}$$

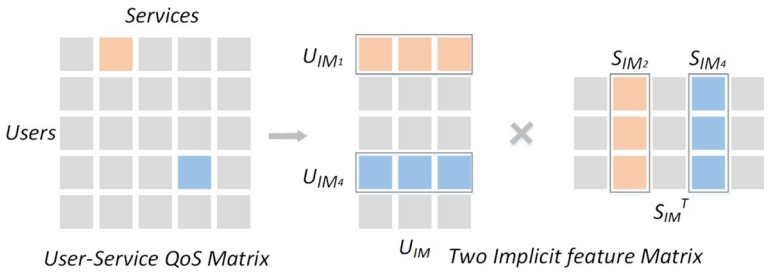

Fig. 4 The implicit feature generated through MF

where ηu and ηs denote the learning rates for U_{IM} and S_{IM}. And $U_{IM_i} \in R^d$ and $S_{IM_j} \in R^d$ represent the implicit vector for user i and service j (m users and n services total).

Feature Interaction Learning

Before the process of feature interaction, we need to uniform the input vector. In Explicit Feature Embedding subsection, the one-hot encoding for explicit feature transformed into embedding vector of d dimension through embedding layer. The user and service implicit vector of d dimension has obtained through MF in Implicit Feature Extraction subsection. Then the vectors of d dimension in two part is concatenated as the input of interaction layer. For a user invoke record, the concatenated vector can be formulated as:

$$e = [e_1, e_2, \cdots e_f, \cdots e_F] = EM \oplus U_{IM_i} \oplus S_{IM_j} \tag{8}$$

Next, we feed the concatenation vectors into the interacting layer, which is implemented as a multi-head self-attentive [33] neural network. The architecture of interacting layer is shown as Fig. 5.

For each interacting layer, high-order features are combined through the attention mechanism, and multiple types of combinations can be evaluated by the multi-head mechanisms which map the features into distinct subspaces. Different orders of combinatorial features can be modeled by stacking multiple interacting layers. The main issue is determining which features should be combined to generate relevant high-order features. Traditionally, this is accomplished by domain experts who create meaningful combinations based on their knowledge. However, this is quite time-consuming to accomplish by hand and impossible to enumerate, even use the complex neural network to fitting could not get the ideal result. In this paper, the multi-head self-attention technique we used can address this issue through the attention mechanism. Besides, this network is very general and can be applied to both numerical and categorical input features. The only drawback of this network is the large amount of feature data required, but this can be offset by the implicit features generated through MF. It is exactly because of the large number of features in the IEC environment that the multi-head self-attention mechanism can perform effectively.

We adopt the key-value attention mechanism [34] to determine which feature combinations are meaningful. Using feature f as an example, we will show how to identify multiple meaningful high-order features that involve feature f. We begin by defining the correlation between features f and k under a specific attention head h as follows:

$$\alpha_{f,k}^{(h)} = \frac{\exp(sim^{(h)}(e_f, e_k))}{\sum_{i=1}^{F} \exp(sim^{(h)}(e_f, e_i))} \tag{9}$$

$$sim^{(h)}(e_f, e_k) = \left\langle W_{Query}^{(h)} e_f, W_{Key}^{(h)} e_k \right\rangle \tag{10}$$

where $sim^{(h)}$ is an attention function which defines the similarity between the feature f and k. Here, we use inner product due to its simplicity and effectiveness to calculate the similarity between feature f and k. $W_{Query}^{(h)}$, $W_{Key}^{(h)} \in R^{d' \times d}$ in (12) are transformation matrices which map the original embedding space R^d into a new space $R^{d'}$. Then, we update the representation of feature f in subspace h via combining all relevant features guided by coefficients $\alpha_{f,k}^{(h)}$.

$$\tilde{e}_f^{(h)} = \sum_{k=1}^{F} \alpha_{f,k}^{(h)} (W_{Value}^{(h)} e_k) \tag{11}$$

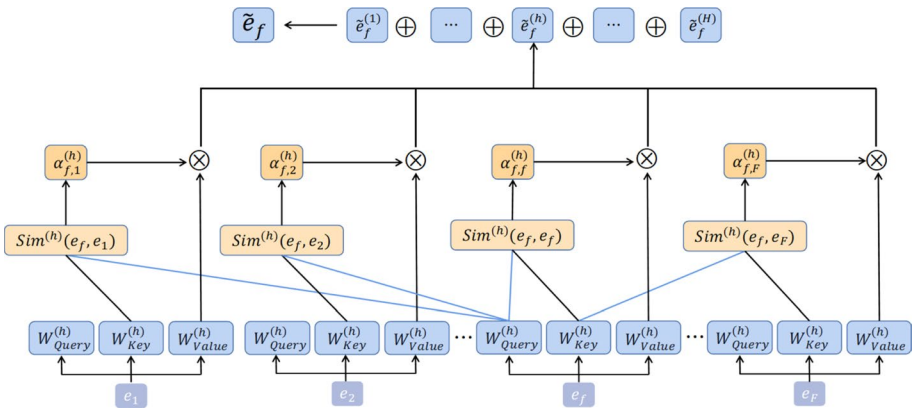

Fig. 5 The architecture of interacting layer. Combinatorial features are conditioned on attention weights, i.e., $\alpha_{f,k}^{(h)}$

where $W_{Value}^{(h)} \in R^{d' \times d}$. And $\tilde{e}_f^{(h)} \in R^{d'}$ is a new combinatorial feature learned by our approach which combined its relevant features under head h. Additionally, a feature is likely to be implicated in many combinatorial features, which we achieve by employing multiple heads that form different subspaces and learn diverse feature interactions individually. As follows, we collect combinatorial features learned in all subspaces:

$$\tilde{e}_f = \tilde{e}_f^{(1)} \oplus \tilde{e}_f^{(2)} \oplus \cdots \oplus \tilde{e}_f^{(H)} \tag{12}$$

where $\oplus$ is the concatenation operator, and H is the number of total heads.

We add typical residual connections [35] to our network to maintain previously learned combinatorial information, including raw individual (i.e., first-order) features. Formally,

$$e_f^{Res} = ReLU(\tilde{e}_f + W_{res} e_f) \tag{13}$$

where $W_{Res} \in R^{d'H \times d}$ is the projection matrix in case of dimension mismatching, and $ReLU(z) = \max(0, z)$ is a non-linear activation function. With such an interacting layer, the representation of each feature e_f is changed into a new feature representation e_f^{Res}, which is a representation of high-order features. We can stack successive such layers, with the previous interacting layer's output served as the input for the next interacting layer. This enables us to model arbitrary-order combinatorial features. The output of the interacting layer is a set of feature vectors $\{e_f^{Res}\}_{f=1}^{F}$, which contains raw individual features reserved by the residual block as well as combinatorial features gained by the multi-head self-attention mechanism. For final QoS prediction, we concatenate all of them and then apply a non-linear projection as follows:

$$\hat{y} = \sigma(W^T(e_1^{Res} \oplus e_2^{Res} \oplus \cdots \oplus e_M^{Res}) + b) \tag{14}$$

where $W \in R^{d'HF}$ is a column projection vector which linearly combines concatenated features, b is the bias, and $\sigma(x) = 1/(1 + e^{-x})$ transforms the values to the predicted QoS value.

Experiment and evaluation

This section, we will evaluate the proposed approach Matrix Factorization Automatic Interaction Network (MFAIN) comprehensively. First, the dataset we used are described. Then, we introduce the evaluation metrics. Finally, we construct a large number of experiments to verify the performance and effectiveness on our approach.

Dataset

We conduct our experiments based on the real world WS-DREAM dataset, which is public and has large number of real world web services collected and maintained by Zheng et al. [36]. It contains 1,974,675 historical QoS records of service invocations (both response time and throughput) historical service invocation records originating from 339 users on 5,825 services. Tang et al. [37] extended this dataset with location and autonomous systems information. The QoS dataset is represented in the form of a user-service matrix, where a row represents the QoS of a user who invokes all of the services, and a column represents the QoS of a service that is invoked by all of the users. To use the dataset to train our model, we wipe off the records which has the negative values of QoS and the null values of feature information. Our Response Time datasets and Throughput datasets contains 1483030 and 1448114 historical QoS records of service invocations respectively.

Evaluation Metrics

Mean absolute error (MAE) and root mean square error (RMSE) used as the two evaluation metrics to measure the accuracy of service QoS prediction among the competing approaches in the experiments. The explanations of MAE and RMSE are shown as follows. MAE is defined as:

$$MAE = \frac{\sum_{i,j} |r_{i,j} - \hat{r}_{i,j}|}{N} \tag{15}$$

$$RMSE = \sqrt{\frac{\sum_{i,j} (r_{i,j} - \hat{r}_{i,j})^2}{N}} \tag{16}$$

where $\hat{r}_{i,j}$ and $r_{i,j}$ are the predicted and ground truth value of the target user invoking a service, respectively; N is the number of the predicted QoS values. It is obvious that the smaller MAE and RMSE are, the better QoS prediction accuracy it indicates.

Performance

To show the prediction accuracy of our proposed method, we compared our method with neighborhood-based CF method UPCC [3], IPCC [38], UIPCC [39] and the feature learning based methods.

DNM [27] is a deep neural model which performs QoS prediction with contextual information based on deep neural networks. LAFIL [23] is a location-aware feature interaction learning method for web service recommendation, which leverages neural networks to learn underlying interaction relations among location features.

We will introduce the default experimental parameter settings and certain parameter studies refer to the

following part. The optimizer is set to Adam Optimizer and the learning rate is set to 0.001 initially in DNM and our method MFAIN. However, the LAFIL use $L2$ regulation with λ =0.0001 for its DNN and CIN module according to their paper. Besides, the default settings for the number of neurons per hidden layer are 200 for a 3-layer CIN and [100, 100, 50] for DNN. And both CIN and DNN share the same embedding layer with dimension of a fixed value of 10. In DNM, embedding layer is fixed at 50, the number of neurons in the hidden layer is configured at 128 for the perception layer, the dropout rate is at the rate of 0.2 in the interaction layer. In our proposed MFAIN, the head number is set to 4, the dropout rate set to 0.1 in the interaction layer. And the remaining parameters are given by the contrast experiments.

To validate the effectiveness of our proposed MFAIN approach for service QoS prediction. We run all these competing methods on the same datasets, where the model parameters with the best performance of the references. We run the experiments of one method for several times and take the mean to avoid the deviations. To explore the performance of our proposed approach, we conducted various experiments under different matrix densities with 5%, 10%, 15%, 20% (both on response time and throughput).

Tables 4 and 5 illustrate the experimental results on response time (RT) and throughput (TP) compared with state-of-the-art approaches. Here, lower MAE and RMSE indicate better performance on service QoS prediction. As can be seen from the tables, the neighborhood-based CF method UPCC, IPCC and UIPCC perform worse than the method based on neural network significantly both on the RT and TP datasets. This shows that the neural network methods using context information for complex nonlinear relationship learning can significantly improve the prediction accuracy. Besides, according to the experimental results, LAFIL is more effective than DNM in most cases, which benefit from its implicit high-order feature interaction learning module. At last, our proposed MFAIN remarkably outperforms all the baseline approaches on both RT and TP datasets as shown in the tables. We make the best results of competing methods and calculate the performance gains on them. Experimental results show that our proposed approach outperforms 14.1% MAE,12.5% RMSE and 14.5% MAE, 14.9% RMSE at most in Response Time and Throughput respectively. It proved the multi-head self-attention mechanism used in the interacting layer is well designed to determine which features should be combined to form meaningful high-order features interaction to improve the prediction accuracy.

Table 4 Performace comparison for response time

RT	Matrix density=5%		Matrix density=10%		Matrix density=15%		Matrix density=20%	
Methods	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UPCC	0.7934	1.6724	0.6787	1.6278	0.6114	1.6038	0.5975	1.5302
IPCC	0.8551	1.4988	0.7091	1.4732	0.6515	1.4548	0.6351	1.4481
UIPCC	0.7354	1.4879	0.6234	1.4502	0.5669	1.4367	0.538	1.4272
DNM	0.4125	1.3463	0.3726	1.2673	0.3678	1.249	0.3575	1.2298
LAFIL	0.3968	1.3234	0.374	1.2808	0.3475	1.2279	0.3321	1.2146
MFAIN	0.3557	1.1909	0.3202	1.1238	0.3114	1.0904	0.3012	1.0629
Gains	10.40%	10.00%	14.10%	11.30%	10.40%	11.20%	9.30%	12.50%

Table 5 Performace comparison for throughput

TP	Matrix density=5%		Matrix density=10%		Matrix density=15%		Matrix density=20%	
Methods	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UPCC	29.4326	71.3528	22.3658	63.6902	20.8361	57.6302	18.1835	55.3903
IPCC	28.7652	62.4814	23.8052	60.1082	22.3727	58.2644	21.2426	56.9312
UIPCC	26.2808	60.8961	22.4295	54.7023	20.219	50.6028	18.9276	48.1629
DNM	18.4903	63.2993	16.2861	55.0821	15.1406	49.394	14.7933	48.4284
LAFIL	17.3753	55.46	14.692	48.552	13.7041	45.2544	12.8145	43.1456
MFAIN	16.0053	47.1833	12.5607	43.0931	11.9322	41.4552	11.4106	40.1253
Gains	7.90%	14.90%	14.50%	11.20%	12.90%	8.40%	11.00%	7.00%

Influence of the Parameters

Impact of matrix density

To explore the impact of matrix density on prediction results in our MFAIN, we vary the matrix density used in training dataset from 5% to 40% in steps of 5% and the rest are test dataset. The results are shown in Fig. 6, which include both MAE and RMSE on RT and TP, respectively. We can observe that MAE and RMSE decrease with the increment of matrix density, indicating the increasing prediction accuracy. Both MAE and RMSE decrease largely at the beginning and then decrease smoothly. These observed results coincide with the intuition that relatively larger matrix density may generate better accuracy of the prediction.

Impact of Dimensionality in Embedding Layer

Dimensionality determines how many latent factors are utilized to represent contextual features in the Embedding Layer. Meanwhile, it shows how many implicit features are utilized to characterize the underlying relation between users and services. To examine the performance impact of dimensionality, we vary the dimensionality d from 16 to 128 in steps of 16, and test on the matrix density of 5%, 10%, 15% and 20% in this experiment. Experimental results of QoS prediction value along with the changes of dimensionality and matrix density on MAE and RMSE are shown in Fig. 7.

It is observed that the accuracy of service QoS prediction improves along with the matrix density, but the amplitude of improvement becomes smaller as the density of the matrix increases. That is because the influence of the amount of data on the prediction results is the main factor when the data are too sparse. Moreover, the best dimensionality d for service QoS prediction changes along with different matrix densities. The best dimensionality d varies from 32 to 48 then to 80, 96, for md=5 % to 20% respectively. The reason of these phenomena is that when service QoS matrix is too sparse, the parameters cannot be fully learned and optimized by model training, so that the implicit features of users and services are poorly represented via interacting layer. On the contrary, as the service QoS dataset becomes denser, more implicit features can be mined and more feature interaction can be learned.

Impact of the Residual block

The standard MFAIN makes use of residual connections, which carries all learned combinatorial features through and thus allow modeling of very high-order combinations. To justify the contribution of residual units, we separate them from our standard model while leaving other structures alone. As shown in Table 6, removing residual connections reduces performance, indicating that residual connections are critical for modeling high-order feature interactions in our proposed method.

Impact of Clustering Parameter

In this experiment, to explore the effect of the number of explicit feature clusters, namely edge device type and network connectivity mode, we fix the data density to 10% and 20%, and the number of interaction layers is set to 4.

As shown in Fig. 8, when the cluster number K is set to 1, it degenerates to the method that excludes device features. When the number of clusters increases, the

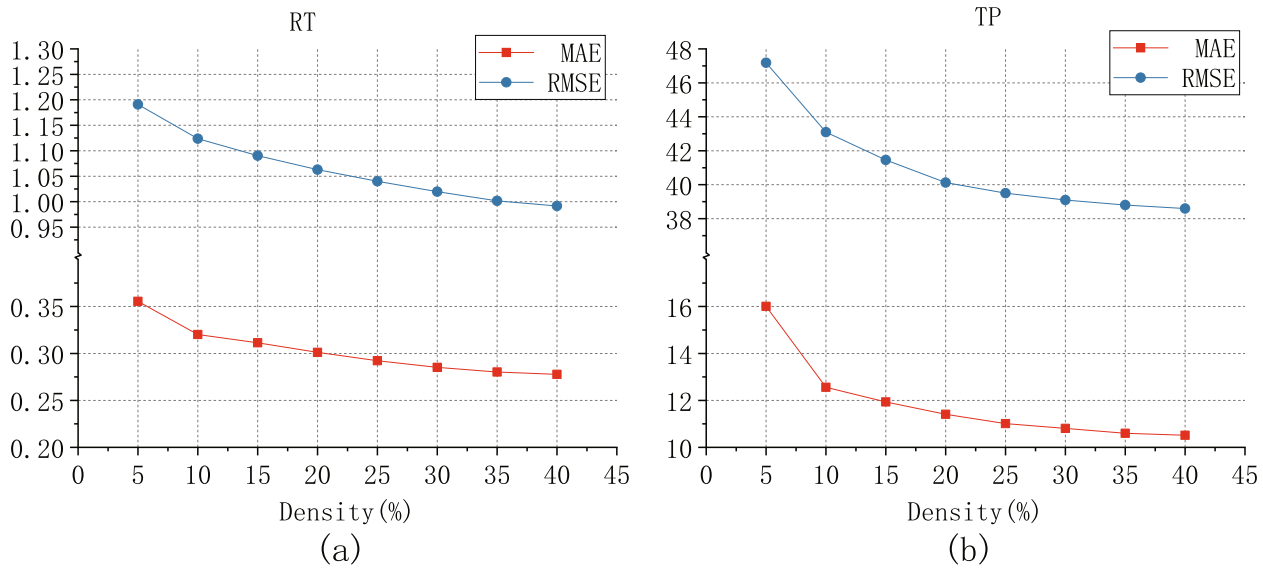

Fig. 6 Impact of parameter matrix density

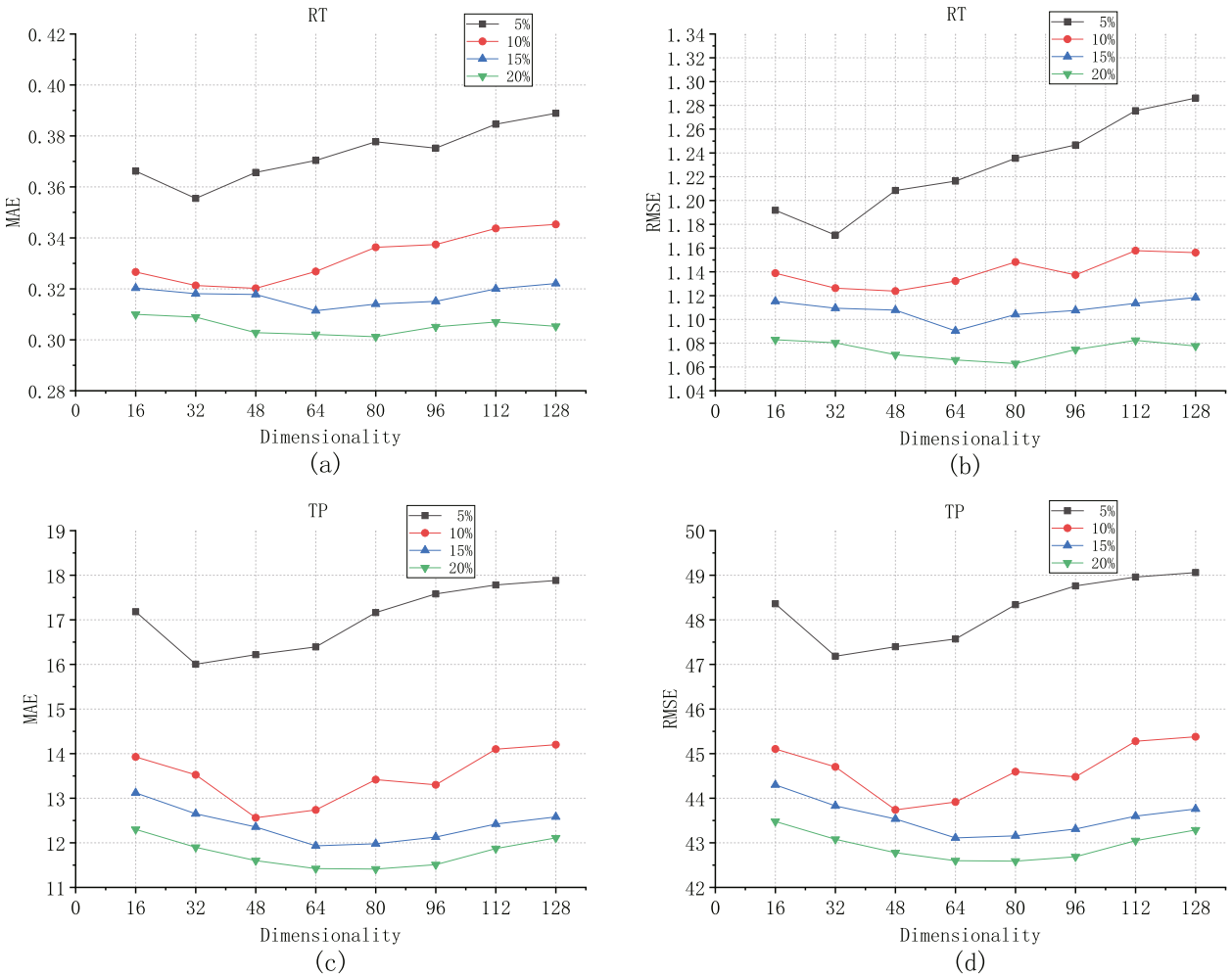

Fig. 7 Impact of Dimensionality in Embedding Layer

Table 6 Impact of the Residual block

Method on MAE	QOS	5%	10%	15%	20%
MFAIN-with residual block	RT	0.3557	0.3202	0.3114	0.3012
MFAIN-without residual block		0.3678	0.3523	0.3363	0.322
MFAIN-with residual block	TP	16.0053	12.5607	11.9322	11.4106
MFAIN-without residual block		16.3561	13.1452	12.3415	11.8635

feature interaction layer can learn more useful information about the interaction, which can improve the prediction accuracy of the model. When the number of clusters reaches 6, the improvement of accuracy by device features becomes slow because the useful information has been fully utilized.

Impact of Interaction Layer

Interaction layer is a key component in our approach MFAIN which used to learn high-order feature combinations. Therefore, we are interested in how the performance change w.r.t. the number of interacting layers, i.e., the order of combinatorial features.

The experimental results are summarized in Fig. 9. We can see that when one interacting layer is used, i.e., the model can learn the low-order feature interactions, it performs well, showing that combinatorial features are very informative for prediction. As the number of interacting layers further increases, i.e., higher-order combinatorial features are taken into account, the performance of the model further increases. When the number of layers reaches four, the performance becomes stable, showing that adding extremely high-order features are not informative for prediction.

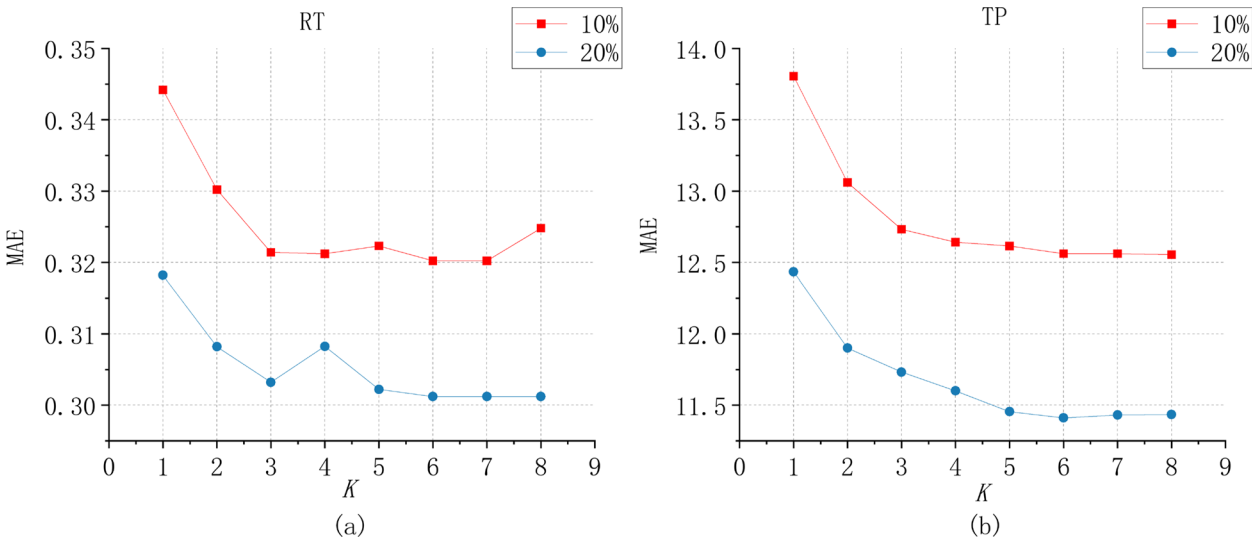

Fig. 8 Impact of the number of cluster

The Effectiveness of Implicit Feature and Device Feature

To explore the effectiveness of the implicit features and device features, we conduct a series of experiments on both RT and TP metrics from 5% density to 20% respectively. The Fig. 10 results show that the implicit features obtained through MF can provide useful hidden information to improve the prediction accuracy. Compared to the implicit features, accuracy improvement caused by device features may be less than implicit features, the reason we analyze is that the number of implicit features are more than device features, even so, device features still very helpful in improving prediction accuracy. In terms of our interaction networks, more features feed into the model, more high-order feature interaction will be learned in the interaction layers, then we will get higher prediction accuracy. Meanwhile, it proved that our approach offers good model expandability, if more contextual information collected, the predictive performance will continue to improve.

Conclusion and future work

The advancement of network technology enables people to invoke the services to meet the needs of high-density and low-latency. Quality-of-Service(QoS) prediction is

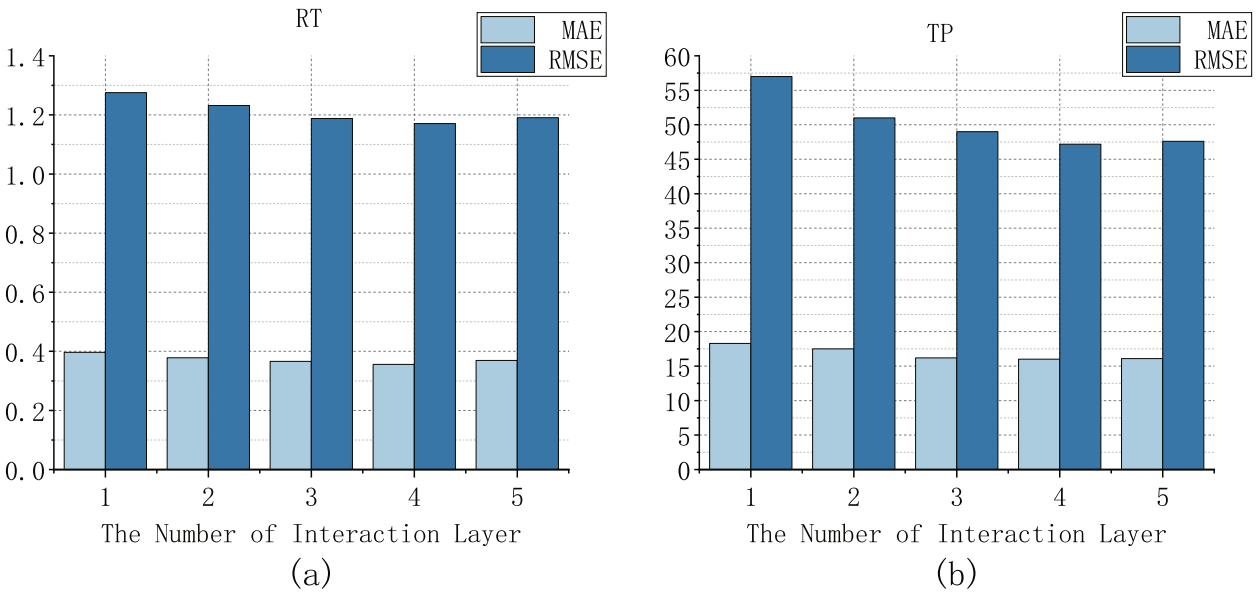

Fig. 9 Impact of the number of Interaction layer

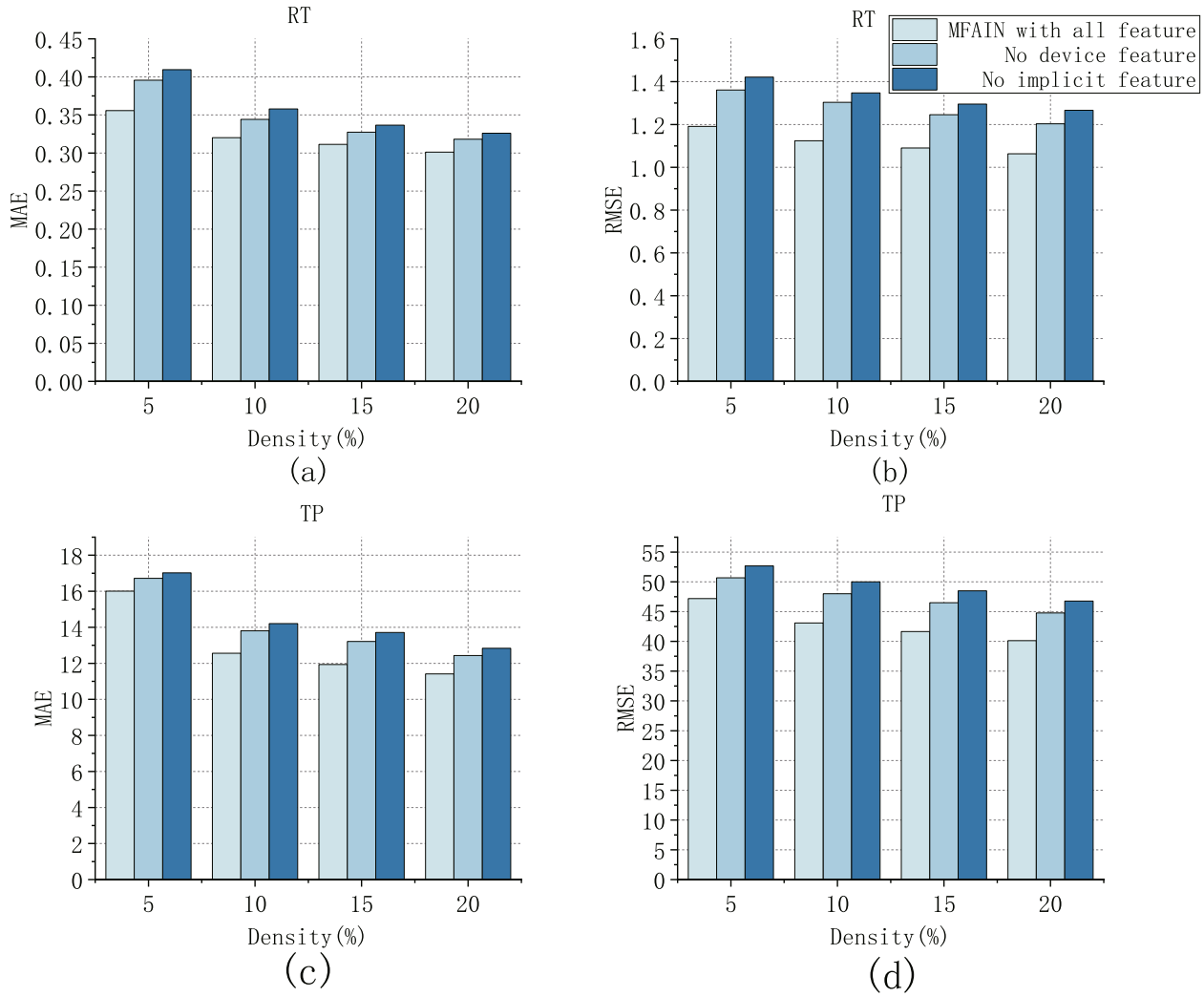

Fig. 10 The effectiveness of Implicit Feature and Device Feature

of vital importance for users to find the proper services among huge numbers of functionally similar web services in the complex future communication scenarios. But current QoS prediction algorithms could not fully consider the contextual information around the smart devices and the underlying relation between user and service. In this paper, we proposed a prediction model to combine the contextual information as the explicit feature and the implicit feature, and then feed into the interaction layer based multi-head self-attention mechanism to learning the low-order and high-order feature interactions. The key to our approach is the newly-introduced interacting layer, which allows each feature to interact with the others and to determine the relevance through learning. Experimental results on real-world data sets demonstrate the effectiveness and efficiency of our proposed model. Besides, our approach offers excellent model expandability, the more features we have, the higher prediction accuracy we get. When integrating with implicit feature captured by MF, we achieve better MAE and RMSE compared to the previous state-of-the-art methods. For future work, we will attempt to take the time series into account to tackle the real-time prediction problem of dynamic QoS.

Abbreviations

QoS	Quality of service
RT	Response time
TP	Throughput
CTR	Click-through rate

Authors' contributions

Wei Zhang and Peiying Zhang formed the conceptions and the methodology; Hongxia Zhang and Dengyue Wang research on formal analysis and wrote the main manuscript text; Lizhuang Tan and Godfrey Kibalya prepared Figs. 1, 2, 3,

4, 5, 6, 7, 8, 9, and 10. All authors reviewed the manuscript. The author(s) read and approved the final manuscript.

Funding
This work is partially supported by the Natural Science Foundation of Shandong Province under Grant ZR2020MF006, ZR2022LZH015, ZR2019LZH013 and ZR2020LZH010, partially supported by the research was supported by RSF (project No. 22-71-10095) and program "Priority-2030", partially supported by the Pilot International Cooperation Project for Integrated Innovation of Science, Education and Industry of Qilu University of Technology (Shandong Academy of Sciences) under Grant 2022GH007, partially supported by the Jinan Scientific Research Leader Studio Project under Grant 2021GXRC091, partially supported by the One Belt One Road Innovative Talent Exchange with Foreign Experts under Grant DL2022024004L, partially supported by the Industry-university Research Innovation Foundation of Ministry of Education of China under Grant 2021FNA01001, partially supported by the Major Scientific and Technological Projects of CNPC under Grant ZD2019-183-006, partially supported by the Open Foundation of State Key Laboratory of Integrated Services Networks (Xidian University) under Grant ISN23-09.

Availability of data and materials
The dataset used in this paper is available for downloading at: http://wsdream.github.io/dataset/wsdream_dataset1.html.

Declarations

Ethics approval and consent to participate
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 28 October 2022 Accepted: 8 January 2023
Published online: 04 February 2023

References

1. Filali A, Abouaomar A, Cherkaoui S, Kobbane A, Guizani M (2020) Multi-access edge computing: A survey. IEEE Access 8:197017–197046. https://doi.org/10.1109/ACCESS.2020.3034136
2. Sermpezis P, Kastanakis S, Pinheiro JI, Assis F, Nogueira M, Menasché D, Spyropoulos T (2019) Towards QoS-Aware Recommendations. arXiv preprint arXiv:1907.06392
3. Zhang P, Wang Y, Kumar N, Jiang C, Shi G (2022) A security- and privacy-preserving approach based on data disturbance for collaborative edge computing in social IoT systems. IEEE Trans Comput Soc Syst 9(1):97–108. https://doi.org/10.1109/TCSS.2021.3092746
4. Zhang P, Jiang C, Pang X, Qian Y (2021) Stec-IoT: A security tactic by virtualizing edge computing on IoT. IEEE Internet Things J 8(4):2459–2467. https://doi.org/10.1109/JIOT.2020.3017742
5. Zhang P, Gan P, Aujla GS, Batth RS (2021) Reinforcement learning for edge device selection using social attribute perception in industry 4.0. IEEE Internet Things J 1. https://doi.org/10.1109/JIOT.2021.3088577
6. Zhang P, Wang C, Jiang C, Benslimane A (2021) Security-aware virtual network embedding algorithm based on reinforcement learning. IEEE Trans Netw Sci Eng 8(2):1095–1105. https://doi.org/10.1109/TNSE.2020.2995863
7. Zhang P, Gan P, Kumar N, Jiang C, Liu F, Zhang L (2022) Survivable virtual network embedding algorithm considering multiple node failure in IIoT environment. J Netw Comput Appl 205:103437
8. Shao L, Zhang J, Wei Y, Zhao J, Xie B, Mei H (2007) Personalized QoS prediction for web services via collaborative filtering. Proceedings - 2007 IEEE International Conference on Web Services, ICWS 2007 (Icws), 439–446. https://doi.org/10.1109/ICWS.2007.140
9. Zheng Z, Ma H, Lyu MR, King I (2011) Qos-aware web service recommendation by collaborative filtering. IEEE Trans Serv Comput 4(2):140–152. https://doi.org/10.1109/TSC.2010.52
10. Sun H, Zheng Z, Chen J, Lyu MR (2013) Personalized web service recommendation via normal recovery collaborative filtering. IEEE Trans Serv Comput 6(4):573–579. https://doi.org/10.1109/TSC.2012.31
11. Zhang Y, Yin C, Wu Q, He Q, Zhu H (2019) Location-Aware Deep Collaborative Filtering for Service Recommendation. IEEE Trans Syst Man Cybern Syst 1–12. https://doi.org/10.1109/tsmc.2019.2931723
12. Wang S, Zhao Y, Huang L, Xu J, Hsu CH (2019) QoS prediction for service recommendations in mobile edge computing. J Parallel Distrib Comput 127:134–144. https://doi.org/10.1016/j.jpdc.2017.09.014
13. Li S, Wen J, Wang X (2019) From Reputation Perspective: A Hybrid Matrix Factorization for QoS Prediction in Location-Aware Mobile Service Recommendation System. Mob Inf Syst 2019. https://doi.org/10.1155/2019/8950508
14. Wu H, Yue K, Li B, Zhang B, Hsu CH (2018) Collaborative QoS prediction with context-sensitive matrix factorization. Futur Gener Comput Syst 82:669–678. https://doi.org/10.1016/j.future.2017.06.020
15. Tang M, Zheng Z, Kang G, Liu J, Yang Y, Zhang T (2016) Collaborative web service quality prediction via exploiting matrix factorization and network map. IEEE Trans Netw Serv Manag 13(1):126–137. https://doi.org/10.1109/TNSM.2016.2517097
16. Chang Z, Ding D, Xia Y (2021) A graph-based QoS prediction approach for web service recommendation. Appl Intell 51(10):6728–6742. https://doi.org/10.1007/s10489-020-02120-5
17. Zhu J, He P, Zheng Z, Lyu MR (2014) Towards online, accurate, and scalable qos prediction for runtime service adaptation. In: 2014 IEEE 34th International Conference on Distributed Computing Systems. Madrid, pp 318–327. https://doi.org/10.1109/ICDCS.2014.40
18. Wu Y, Xie F, Chen L, Chen C, Zheng Z (2017) An embedding based factorization machine approach for web service qos prediction. In: International Conference on Service-Oriented Computing, Springer, pp 272–286
19. Yang Y, Zheng Z, Niu X, Tang M, Lu Y, Liao X (2021) A location-based factorization machine model for web service qos prediction. IEEE Trans Serv Comput 14(5):1264–1277. https://doi.org/10.1109/TSC.2018.2876532
20. Zhang Y, Yin C, Wu Q, He Q, Zhu H (2019) Location-aware deep collaborative filtering for service recommendation. IEEE Trans Syst Man Cybern Syst 51(6):3796–3807
21. Gao H, Xu Y, Yin Y, Zhang W, Li R, Wang X (2019) Context-aware qos prediction with neural collaborative filtering for internet-of-things services. IEEE Internet Things J 7(5):4532–4542
22. Shen L, Pan M, Liu L, You D, Li F, Chen Z (2020) Contexts enhance accuracy: On modeling context aware deep factorization machine for web api qos prediction. IEEE Access 8:165551–165569
23. Wang Z, Xiao Y, Sun C, Zheng W, Jiao X (2020) Location-aware feature interaction learning for web service recommendation. In: 2020 IEEE International Conference on Web Services (ICWS), IEEE, pp 232–239
24. Yin Y, Chen L, Xu Y, Wan J, Zhang H, Mai Z (2020) Qos prediction for service recommendation with deep feature learning in edge computing environment. Mob Netw Appl 25(2):391–401
25. Xia Y, Ding D, Chang Z, Li F (2021) Joint deep networks based multi-source feature learning for qos prediction. IEEE Trans Serv Comput 15(4):2314–2327
26. Chen Y, Yu P, Zheng Z, Shen J, Guo M (2022) Modeling feature interactions for context-aware qos prediction of IoT services. Futur Gener Comput Syst 137:173–185. https://doi.org/10.1016/j.future.2022.07.017
27. Wu H, Zhang Z, Luo J, Yue K, Hsu CH (2018) Multiple attributes qos prediction via deep neural model with contexts. IEEE Trans Serv Comput 14(4):1084–1096
28. Zhang P, Huang X, Zhang L (2021) Information mining and similarity computation for semi- / un-structured sentences from the social data. Digit Commun Netw 7(4):518–525. https://doi.org/10.1016/j.dcan.2020.08.001
29. Juan Y, Zhuang Y, Chin WS, Lin CJ (2016) Field-aware factorization machines for ctr prediction. In: Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16). Association for Computing Machinery, New York, pp 43–50. https://doi.org/10.1145/2959100.2959134
30. Lian J, Zhou X, Zhang F, Chen Z, Xie X, Sun G (2018) xdeepfm: combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18). Association for Computing Machinery, New York, pp 1754–1763. https://doi.org/10.1145/3219819.3220023

31. Ye F, Lin Z, Chen C, Zheng Z, Huang H (2021) Outlier-resilient web service qos prediction. In: Proceedings of the Web Conference 2021 (WWW '21). Association for Computing Machinery, New York, pp 3099–3110. https://doi.org/10.1145/3442381.3449938
32. Bottou L (2010) Large-scale machine learning with stochastic gradient descent. In: Lechevallier Y, Saporta G (eds) Proceedings of COMPSTAT'2010. Physica-Verlag HD, Heidelberg, pp 177–186
33. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN et al (2017) Attention is all you need. In:Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17, pp 6000–6010
34. Miller A, Fisch A, Dodge J, Karimi AH, Bordes A, Weston J (2016) Key-value memory networks for directly reading documents. arXiv preprint arXiv:1606.03126
35. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, pp 770–778. https://doi.org/10.1109/CVPR.2016.90
36. Zheng Z, Zhang Y, Lyu MR (2014) Investigating qos of real-world web services. IEEE Trans Serv Comput 7(1):32–39. https://doi.org/10.1109/TSC.2012.34
37. Tang M, Zhang T, Liu J, Chen J (2015) Cloud service qos prediction via exploiting collaborative filtering and location-based data smoothing. Concurr Comput Pract Experience 27(18):5826–5839
38. Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web (WWW '01). Association for Computing Machinery, New York, pp 285–295. https://doi.org/10.1145/371920.372071
39. Zheng Z, Ma H, Lyu MR, King I (2009) Wsrec: A collaborative filtering based web service recommender system. In: 2009 IEEE International Conference on Web Services, pp 437–444. https://doi.org/10.1109/ICWS.2009.30

Publisher's Note