Privacy-Aware Forecasting of Quality of Service in Mobile Edge Computing

Huiying Jin, Pengcheng Zhang, *Member, IEEE*, Hai Dong, *Senior Member, IEEE*, Yuelong Zhu, and Athman Bouguettaya, *Fellow, IEEE*

Abstract—We propose a novel privacy-aware Quality of Service (QoS) forecasting approach in the mobile edge environment – Edge-PMAM (Edge QoS forecasting with Public Model and Attention Mechanism). Edge-PMAM can make real-time, accurate and personalized QoS forecasting on the premise of user privacy preservation. Edge-PMAM comprises a public model for privacy-aware QoS forecasting in an edge region and a private model for personalized QoS forecasting for an individual user. An attention mechanism atop Long Short-Term Memory and an automated edge region division solution are devised to enhance the prediction accuracy of the public and private models. We conduct a series of experiments based on public and self-collected data sets. The results demonstrate that our approach can effectively improve forecasting performance and protect user privacy.

Index Terms—Mobile edge computing, joint training, independent learning, privacy-aware forecasting

1 Introduction

The digital revolution has made profound impact on all walks of life, including society, business, entertainment, etc [1]. The service paradigm is a powerful paradigm to leverage the large amount of data generated from desktop and mobile apps, social media, IoT devices, etc [2]. The service paradigm abstracts data applications into small and independent functions delivered within different Quality of Service (QoS) parameters (called non-functional attributes). Each service may encapsulate several functional and non-functional attributes. Web services are one of the key technologies to implement the service paradigm [3]. Web services are broadly used in such fields as business, service sector, IT services, etc [4]. QoS of Web services typically include *response time*, *throughput*, *reliability*, etc [5], [6].

Cloud computing is an emerging technology that provides virtualized computing resources, software and its runtime environment as services over the Internet, known as IaaS, SaaS and PaaS [7]. Web services provide standardized interfaces and protocols for cloud service interoperations and orchestration [8], [9]. Edge computing is a new distributed computing paradigm. It moves computing power from cloud data centers to edges of networks to minimize delays. Edge computing is the ability to transfer needed computing power closer to where it is needed [10]. Mobile Edge Computing (MEC) is a type of edge computing technologies deployed on wireless base stations [11]. MEC enables network services running closer to mobile users to reduce service latency [12], [13].

There are two major challenges in mobile edge environments: 1) *the elevated requirements for data privacy protection* [14], [15], and 2) *the needs of high accuracy and efficiency for QoS forecasting*. The following scenario illustrates the two challenges.

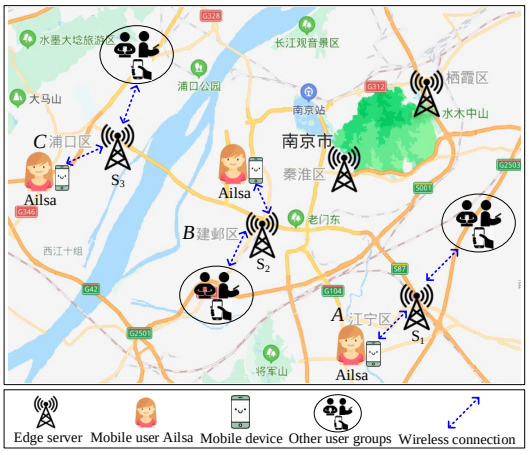


Fig. 1: A mobile edge service invocation scenario

Fig. 1 shows the distribution of edge servers in some municipal districts of *Nanjing*. Each edge server serves its located municipal district. Let us assume that a mobile user *Ailsa* is taking a metro train from district A to district C and passing through district B. She is watching a *Twitter* video on the journey. The video service quality needs to be forecasted. If the service quality is predicted to decline in the next few seconds, the video service can cache or download the effected video beforehand to maintain the smoothness. The sequence of the edge servers accessed by *Ailsa* is: S_1-S_2-S_3. There is no *Twitter* video service quality data (e.g., buffering speed, resolution, etc.) generated, when she initially accesses the service in S_1. The data is inadequate to forecast the forthcoming service quality. Thus, *Alisa*'s service

- *H. Jin, P. Zhang and Y. Zhu are with the Key Laboratory of Water Big Data Technology of Ministry of Water Resources and the College of Computer and Information, Hohai University, Nanjing 211100, China E-mail: 367046895@qq.com; pchzhang@hhu.edu.cn; ylzhu@hhu.edu.cn*
- *H. Dong is with the School of Computing Technologies, RMIT University, Melbourne, VIC 3000, Australia E-mail: hai.dong@rmit.edu.au*
- *A. Bouguettaya is with the School of Computer Science, The University of Sydney, NSW, Australia E-mail: athman.bouguettaya@sydney.edu.au*

Manuscript received XXX XX, XXXX; revised XXX XX, XXXX.

quality forecasting needs to employ other users' historical records in S_1. The data interchange increases the likelihood of data leakage and user privacy breach. Next, the edge server accessed by *Alisa* is switched from S_1 to S_2, when she reaches district B. Similarly, the video service quality in S_2 also needs to be predicted. *Alisa* does not have any historical data in S_2 when launching the service in S_2. The QoS forecasting may use *Alisa*'s data from S_1. However, S_1 and S_2 may have different network status. Therefore, the previous data in S_1 cannot be directly leveraged in S_2. The same problem will occur again when she reaches district C.

We summarize the challenges faced by the QoS forecasting in the above mobile edge scenario as follows:

i). Traditional privacy-aware QoS forecasting approaches cannot guarantee local user data privacy. The historical QoS data generated by the same users in different edge regions is less referential due to the distributed and diverse nature of mobile edge environment. Users need to obtain historical QoS data of other users in same edge regions for forecasting [15]. Thereby, the interactions between local users are more intensive in an edge region. The traditional privacy mechanisms usually employ random noise insertion or encryption to protect the original data [16]. The behavioral patterns and data encryption rules of users are easier to be mined and reasoned, with the increasing number of interactions between users [14]. In addition, the traditional mechanisms cannot completely avoid the existence of malicious users. Therefore, it is of great significance to propose a more secure personalized QoS forecasting approach in the mobile edge environment.

ii). Realizing real-time, accurate and efficient QoS forecasting is a challenging task in the mobile edge environment. MEC aims to reduce the latency of network operations and service delivery by providing service environment and computing capability on the edge of mobile network. Users' requests need to be quickly processed and answered to realize the goal of MEC. The similar goal also applies to QoS forecasting in the mobile edge environment. Accurate QoS forecasting will better insure users' service positive experience. MEC requires real-time and reliable data to realize the goal of accurate and real-time QoS forecasting. However, traditional forecasting methods mostly use historical data. They cannot meet the requirements of forecasting accuracy in the edge environment. They also lack real-time dynamic updating mechanism. Therefore the traditional forecasting methods cannot be applied to the edge environment with an intensified focus on real-time performance.

Our previous work [17] proposes a preliminary security-aware edge QoS forecasting approach, named Edge QoS Per-PM (Edge QoS forecasting with Personalized training based on Public Models). Edge QoS Per-PM employs the principle of integrating cooperative learning and independent learning. It contains 1) *a public model for privacy-aware QoS forecasting in an edge region* and 2) *a private model for personalized QoS forecasting for an individual user*. The public model is a machine learning model trained upon privacy-insensitive public data in a region to avoid local user data privacy breach. The private model is an extension of the public model trained upon the user's private data to enhance the QoS forecasting accuracy. However, Edge QoS Per-PM has the following major limitations:

- *Lack of the impact of long-term historical QoS data on forecasting.* The public and private models of Edge QoS Per-PM are both built upon Long Short-Term Memory (LSTM). It only considers the impact of the QoS data in the previous single moment on the forecasting results. It ignores the impact of the long-term QoS historical data on the forecasting results.
- *Lack of automated edge region division.* An edge region refers to a geographical area where its located edge servers share the stored users' historical QoS data for training a public model. The QoS data from the same edge region is usually subject to the similar environmental conditions, e.g., network bandwidth, communication rate, etc. Therefore, their stored QoS data is more correlated and suitable for training the same public model. Edge QoS Per-PM adopts a manual edge region division process. It is time-consuming and impractical for larger numbers of edge servers.

In this paper, we propose novel approach - Edge-PMAM (Edge QoS forecasting with Public Model and Attention Mechanism) to be employed in a privacy-aware edge QoS forecasting scheme.

The major contributions include:

- We adopt an attention mechanism on top of LSTM to improve the performance of the public and private models. Attention mechanism is a powerful neural network technique. It allows neural networks to pay more attention to some special parts of training data [18]. This attention mechanism allows us to assign different initial weights to historical moments. It enables the model to comprehensively consider the impact of several historical moments on prediction. This enhanced model shows improved prediction accuracy in comparison to our previous model in the experiments.
- We devise an automated optimal edge region division method. This method can 1) project the original spherical coordinates onto a plain to calculate the absolute distance between two edge servers based on Miller projection, 2) obtain the optimal k for edge region division by means of elbow method or silhouette coefficient, and 3) automatically divide the edge regions based on K-means clustering.
- We conduct a more comprehensive evaluation on the data sets of two geographical areas and a self-collected real world data set. The experimental results show that Edge-PMAM achieves the goal of secure, accurate and efficient forecasting.

The structure of the paper is organized as follows. Section 2 reviews the work relevant to QoS forecasting. Section 3 introduces the background knowledge and relevant theoretical basis of our approach. Section 4 presents the details of our approach. Section 5 elaborates the experimental design and result analysis. Section 6 summarizes the paper and plans our future work.

2 RELATED WORK

The existing QoS forecasting studies can be briefly divided into QoS forecasting in traditional environments [19], [20]

and mobile edge environments [15], [21].

2.1 Traditional QoS Forecasting

Traditional QoS forecasting mainly include context-aware, time-based, location-based, and privacy preserving based approaches.

Wu et al. [19] proposed a context-aware matrix factorization based QoS prediction method for cloud services. They considered the complexity of service invoking and the implicit and explicit factors between QoS data. This method effectively solves the sparse data prediction problem. However, it is not suitable for prediction scenario with temporal attribute. Wang et al. [22] devised a spatial-temporal QoS prediction model for time-aware Web services. Temporal QoS prediction is expressed as a regression problem. The most similar QoS sequence is retrieved based on the location of the end user and the service to improve prediction accuracy. It cannot predict QoS values in the future time periods. Chen et al. [23] adopted a Web service recommendation method using location and QoS information. They clustered users and services based on their historical geographic location and service information. Personalized service recommendation is made for users based on the clustering results. Nevertheless, the scalability of the clustering method needs to be improved.

Liu et al. [20] proposed a privacy preserving Web service QoS prediction framework based on differential privacy. The framework generates random noises based on Laplace mechanism to protect the original data. Thereby, it ensures the privacy of users on the premise of data availability. However, this method disadvantages the forecasting accuracy. Polat et al. [24] proposed a cooperative filtering privacy protection approach based on the random disturbance technology. This approach realizes data disguising by adding random numbers to different disguising ratings. It then sends the results to the server to achieve the purpose of privacy and accuracy balancing. However, the accuracy of this approach is restricted by the degree of information disclosure. Li et al. [25] devised a privacy preserving QoS forecasting approach based on garbled circuit and homomorphic encryption. The final recommendation results and all the intermediate results in the recommendation process are in ciphertext or random shares. Hence, it ensures data security. The method has the problem of high computation and communication costs. It is generally suitable for off-line predictions.

2.2 Edge QoS Forecasting

With the development of mobile edge computing technologies, many scholars have been devoted to the research of QoS forecasting in the edge environment.

Wang et al. [15] designed a QoS prediction method based on collaborative filtering in mobile edge environments. This method performs QoS prediction and service recommendation based on user mobility. However, this method is constrained by the density of data matrix and distribution density of edge servers. White et al. [26] designed a short-term QoS forecasting algorithm deployed at edges of networks. The algorithm bases on the noisy echo state network to reduce training time and improve prediction accuracy. The universality of the algorithm in service-oriented edge applications is still a challenge. Yin et al. [27] presents a QoS prediction service recommendation model based on deep feature learning in mobile edge computing. This model employs learned neighbors' deep latent features to infer user or service features. Nonetheless, it does not consider the influence of time and environment changes on edge computing.

Privacy and security-aware QoS forecasting in mobile edge environments has also become a hot research area. Zhang et al. [21] proposed a privacy preserving QoS forecasting method in mobile edge environments. This method adds noises to original data based on differential privacy to protect user privacy. It achieves a trade-off between prediction accuracy and privacy protection. This method can be cracked once attackers master the noise addition rules. Zhang et al. [28] comprehensively considered the influence of user preferences on the edge environment. They developed a distributed edge QoS prediction model with privacy protection. Laplace vector mechanism was utilized for privacy protection in edges. However, the usability of this method in real edge environments needs further studies. Zhang et al. [29] designed a credible privacy protection QoS prediction model based on federated learning and reputation mechanisms. The training framework of this model is jointly constructed by a central server and users. Users do not need to transmit data to avoid the risk of privacy leakage. Nevertheless, this model cannot completely avoid the existence of malicious users.

3 PRELIMINARIES

3.1 Privacy Issues in Mobile Edge Environment

In recent years, mobile computing technology has gradually shifted from centralized mobile cloud computing to Mobile Edge Computing (MEC). This shift results from the rapid development of mobile networks and the diversification of service scenarios [30]. MEC has the characteristics of location dependency, mobile support, low delay, decentralization and distribution. It can better meet the new requirements of the Internet of Things, 5G, mobile devices, etc [31]. It is an effective supplement and extension of cloud computing. MEC aims to push mobile computing, network control and storage to the edge of the network. It provides IT service environments and cloud computing functions at the edge of the network. It can effectively reduce delay to ensure efficient network operations and service delivery.

The computing power of distributed nodes in MEC can reduce the load of cloud data centers. However, MEC nodes adjacent to end users may collect sensitive data related to user identity, locations, application usage, etc. In addition, centralized control in MEC is difficult due to the large-scale and decentralized MEC nodes. Therefore, the edge nodes with security vulnerabilities may become the entrance for intruders to attack MEC networks. Intruders can mine and steal the privacy data exchanged between entities once they enter the network. MEC is an open ecosystem. Different trust domains are controlled by different infrastructure owners. It is difficult to locate attackers. Obtaining evidences becomes a challenge once privacy breach occurs [32]. Therefore, security is a significant challenge for MEC [11].

3.2 Notion of Privacy

Privacy has been defined in a variety of ways. For example, Saltzer et al. [33] defined privacy as "the ability of an individual (or organization) to determine whether, when, and to whom to disclose information about an individual (or organization)". Zhou et al. [34] defined it as "sensitive information or data characteristics that the data owner is unwilling to be disclosed". We provide the following definitions related to the privacy in the context of QoS forecasting guided by these existing definitions.

Definition 1 (QoS training data privacy). It refers to a user's personal information or data characteristics that the user is unwilling to be disclosed, including the user's geographic location, the area he/she belongs to, the user's service invocation information, and the user's original service quality data.

Definition 2 (QoS forecasting model privacy). It refers to the information related to a QoS forecasting model, i.e., the structure, algorithm, weight parameters, and activation function of the model.

Definition 3 (QoS forecasting data privacy). It is a special form of a user's personal data. This type of data can generally be used to infer the user's historical and future personal data.

Definition 4 (QoS forecasting result privacy). It refers to the results for a user through the QoS forecasting model.

The above types of privacy need to be protected during the QoS forecasting process. Their leakage will lead to serious consequences.

3.3 Threat Model

QoS forecasting in the edge environment usually involves three parts: user groups, mobile networks, and edge servers. As shown in Fig. 2, when *Alisa* first uses the service, she needs to reference the QoS data of the service from other users through wireless connection P_1 due to the lack of QoS data. When she generates a certain amount of QoS data, she transfers the data to an edge server to train a forecasting model through wireless connection P_2. The server will send back the forecasting result to her. During this process, user privacy may be threatened as follows: 1) the data of other users transmitted may be subject to leakage, tampering or inference attacks during P_1; 2) the training data, inputs and outputs of the forecasting model might be leaked or tempered during P_2.

The existing privacy-aware QoS forecasting approaches are built upon traditional clouds [20], [24], [25] or mobile edge environments [15], [21], [28]. They usually rely on random disturbance technologies, data encryption or random noise mechanisms for privacy protection. Most of these technologies directly interact with user data (through P_1). User data can be leaked or tempered, causing these technologies to be less secure in theory. In addition, there are model-based approaches [29], [35]. They usually rely on transferring parameters of the local (i.e. private) model (through P_2) for protection. These private model parameters are vulnerable to the attacks during the transfer process [36]. Therefore, how to ensure user data privacy and model security is a key issue in dealing with QoS forecasting privacy threats in the edge environment.

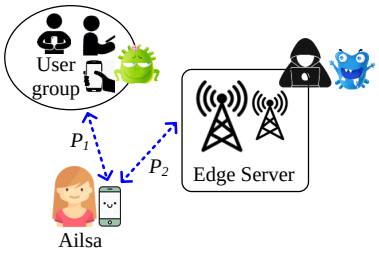


Fig. 2: A privacy threat scenario during edge QoS forecasting

3.4 LSTM and Attention Mechanism

Long Short-Term Memory (LSTM) is a type of time recurrent neural networks. LSTM is specially designed to solve the long-term dependence problem of Recurrent Neural Network (RNN) [37]. It was first proposed by Sepp Hochreiter and Jürgen Schmiduber in 1997 [38]. There are four network layers in LSTM. The cell information is controlled by the gate structure. The gate can selectively decide which information can pass through to exchange information. LSTM consists of three gates: forget gate, input gate and output gate. Forget gate determines the reserved information of the cell state. Input gate determines the updated information of the current network. Output gate controls the output information of the cell state.

Attention mechanism is formed on the basis of human vision. Attention attracts people to pay more attention to important parts of captured information and obtain details of targets as much as possible. Similarly, less attention is paid to irrelevant information around targets, i.e., suppressing the irrelevant information. Attention mechanism was only applied to image recognition in computer vision at first [18]. Now it has become a key concept in the field of neural networks. Attention has become a critical part of neural network structure in the field of AI. It has been widely used in question answering systems [39], machine translation [40], speech recognition [41], image capture [42] and other fields.

Attention mechanism is a part of a prediction model. It can allow the prediction model to focus on different parts of the input in order. The order is based on the impact of each part on the final prediction. Using attention mechanism can enable a system to extract and output the most relevant information. This can effectively improve the output quality of the model. Attention mechanism can be introduced into LSTM to effectively employ the LSTM output of historical moments [43]. The output of a LSTM network in different time slices can be weighted to express the correlation between the current time slice and the historical time slice. This can improve the output accuracy of the network.

4 THE EDGE-PMAM APPROACH

The scheme of privacy-aware forecasting is presented in Section 4.1. The workflow of Edge-PMAM is outlined in Section 4.2. The four steps of Edge-PMAM are introduced in details in Section 4.3, Section 4.4, Section 4.5 and Section 4.6.

4.1 Privacy-Aware Forecasting Scheme

We realize privacy-aware edge QoS forecasting by combining edge clouds and users' mobile devices. Fig. 3 shows our proposed privacy-aware forecasting scheme, where Edge-PMAM is employed for QoS forecasting. The service providers own copies of users' QoS data [44]. They provide the privacy-insensitive public data (i.e. median QoS values) for the offline public model training in edge clouds. Mobile users can download public models to their devices. The devices perform user-side online training and forecasting based on users' private data. The scheme prevents users from the situation where their privacy defined in Section 3.2 is breached. Therefore, this architecture can effectively solve the issues faced by the existing solutions when confronting the threat model. In addition, the public model can be continuously reinforced and generate sound private models for each individual user.

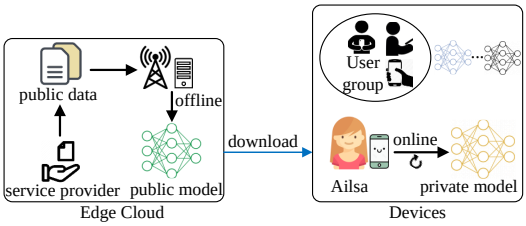


Fig. 3: Privacy-aware edge QoS forecasting scheme

4.2 Overview of Edge-PMAM

We propose a privacy-aware QoS forecasting approach (Edge-PMAM) in the mobile edge environment. Edge-PMAM works towards the goals of privacy-aware, accurate and real-time personalized forecasting. The system workflow is shown in Fig. 4. It is mainly divided into four steps:

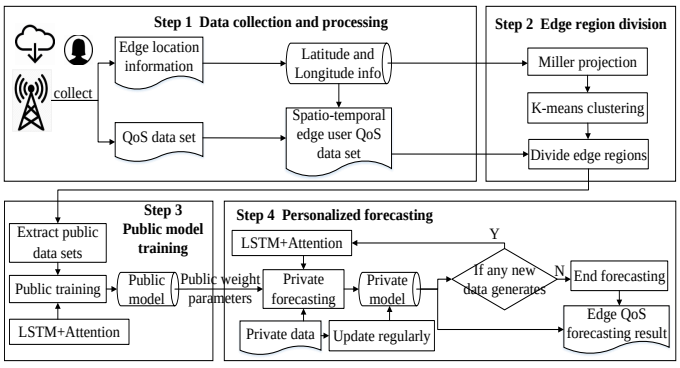


Fig. 4: The overview of Edge-PMAM

1) *Data collection and processing.* First, the edge location information and the QoS data sets are collected to form the spatio-temporal edge user QoS data set. The longitude and latitude values in the edge location information are employed to determine the geographical distribution of edge servers. Here we adopt the scenario of *Ailsa* in Fig. 1 as an example. First, we obtain the longitude and latitude location information of edge servers in three municipal districts of Nanjing. Next, we collect the QoS attribute values generated by users, e.g., the QoS data of *Ailsa* watching the *Twitter* video across different districts. Finally, we form the spatio-temporal edge QoS data set of mobile users.
2) *Edge region division.* The longitude and latitude values of the edge servers collected in step 1 is converted into the plane coordinate values based on Miller projection. K-means clustering is then employed to determine the number of clusters k ($k \geq 2$) of the plane coordinate data by means of the elbow method or silhouette coefficient. The whole area is therefore divided into k regions. In *Ailsa*'s example, districts A, B and C are the final result of the edge region division.
3) *Public model training.* The spatio-temporal edge user QoS data set collected in step 1 is divided according to the edge regions identified in step 2. We then make use of the public data set of each region to train an LSTM model with attention mechanism to obtain the public model. The weight parameters of these public models will be transferred to users of corresponding regions for personalized forecasting. In *Ailsa*'s example, we extract and train the QoS data of certain users in district A. All these users watch *Twitter* videos in a specific period of time. *Ailsa* can then obtain the public training results.
4) *Personalized forecasting.* A user uses the weight parameters of the public model in his/her belonged region as the initial parameters of his/her own private model. The private model is then trained upon the user's private data to make personalized forecasting. A user's private data is the temporal QoS data generated in the process that the user interacts with a service. The private data is used to train the private model in each time interval to update its weight parameters continuously. This ensures the real-time performance of the weight parameters in the dynamic edge environment and improves the prediction accuracy. The private model is continuously optimized with increasing training iterations. QoS forecasting results will be generated for future time slices. In *Ailsa*'s example, the prediction of *Twitter* video quality is first based on the public model of district A. A private model is then continuously trained based on the data steam generated by *Ailsa*. New QoS is therefore predicted in terms of the trained private model. The personalized QoS forecasting will become increasingly more accurate with the time.

4.3 Data Collection and Processing

Let us make use of the scenario of *Ailsa* to explain this process. Edge servers S_1, S_2, S_3 respectively log quality data about the *Twitter* video service, e.g., frame rates and resolution, from many users. We collect the QoS data from these scattered servers and the location information of the edge servers (i.e., longitude and latitude values). For the QoS data generated on the user side, such as response time, we collect it from users through a trusted third party. We fuse the QoS data and the location information to form the spatio-temporal edge user QoS data set. We then adopt a map service [1] to locate the edge servers on the map based on the longitude and latitude information of the edge servers.

1. https://zt.changjing.com.cn/

4.4 Edge Region Division

We employ Miller projection to convert the longitude and latitude values of the edge servers into plane coordinates. We then make use of K-means clustering to cluster the edge servers into k edge regions based on their plane coordinates. In an edge region, the QoS is affected by many factors, e.g., bandwidth, network traffic or the number of concurrent service requests. Servers in the same edge region can be assumed to share the similar edge environment based on the survey result from [45]. They concurrently respond to multi-user service requests in the region. The logged and predicted QoS data is the result of the concurrency. Thus, a public model is trained for each region. A user will use a new public model for initial QoS forecasting when the user's region changes. The clustering can improve the forecasting accuracy. The edge region division consists of the following steps.

4.4.1 Miller Projection

The longitude and latitude values of edge servers collected are points on the sphere. We use Miller projection method to convert the longitude and latitude values into plane coordinates. The calculation formulas of Miller projection are as follows:

$$L = 2\pi R \tag{1}$$

L is the circumference of the earth, where R = 6381.372. The longitude and latitude values are respectively converted from degrees to radians. Let x_r and y_r represent the radians of the longitude and latitude values, then

$$x_r = lon * \pi/180 \tag{2}$$

$$y_r = lat * \pi/180 \tag{3}$$

A plane projection is also needed for y_r. The projection can be realized by the following formula:

$$y_p = 1.25 * log(tan(0.25 * \pi + 0.4 * y_r)) \tag{4}$$

Finally, the radian is converted to the actual distance as follows:

$$X = (W/2) + (W/(2 * \pi)) * x_r \tag{5}$$

$$Y = (H/2) - (H/(2 * mill)) * y_p \tag{6}$$

When the plane is expanded, the x-axis is equal to the circumference, so $W = L$. The y-axis is about half the circumference, so $H = L/2$. $mill$ is a constant in Miller projection.

We get a set of coordinates (X, Y) after the above transformation. The coordinates can be directly used in the operation of a two-dimensional rectangular coordinate system.

4.4.2 K-means Clustering

Clustering refers to dividing samples into several groups. The samples in each group have smaller difference according to the distance or similarity between samples. The k value of K-means clustering is the number of clusters. The optimal k value can be determined by elbow method or silhouette coefficient [46].

The mathematical principle of elbow method is expressed as follows. Let us assume that the cluster is divided into $(C_1, C_2, ..., C_k)$. The goal is to minimize the squared error E:

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|_2^2 \tag{7}$$

where $x \in (X, Y)$, μ_i is the mean vector of cluster C_i, also known as the centroid. It is expressed as:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \tag{8}$$

Silhouette coefficient is another way to evaluate the clustering effect. Its value is between [-1, 1]. The closer it is to 1, the better cohesion and resolution the clustering. For a point i in the cluster, its silhouette coefficient is:

$$S(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \tag{9}$$

where $i \in (X, Y)$, b_i is the mean of the distance from sample point i to points in the nearest cluster except its own cluster, and a_i is the mean of the distance from point i to other points in the same cluster. The mean value of all samples is called silhouette coefficient of clustering results. We select the k value corresponding to the maximum coefficient.

We randomly select k points in the data set as the centroid after determining the k value. The detailed clustering steps are as follows. **Step 1**: The Euclidean distance from each point to each centroid in the data set is calculated. The points are divided based on their distance to the nearest centroid. **Step 2**: k clusters are formed after classifying all the data. We recalculate the centroid of each cluster. **Step 3**: If the distance between the new centroid and the original centroid is less than a certain threshold value, it means that the clustering has achieved the desired effect. The algorithm is terminated. **Step 4**: If the distance between the new centroid and the original centroid changes greatly, we repeat Steps 1∼3 until the position of the centroid no longer changes or it reaches the maximum number of iterations. The algorithm is terminated. Finally, the clustering is formed.

4.5 Public Model Training

The spatio-temporal edge user QoS data set fused in Section 4.3 is divided according to the edge regions identified in Section 4.4. A public model is trained offline based on the divided QoS data in each region. The purpose of the public model training is to provide an initial QoS forecasting model for all users in an edge region. The public model can capture general video quality in that region. Its training bases on privacy-insensitive public data, e.g., the average frame rate and resolution of the video published by *Twitter* formed upon mass user data. In addition, it does not expose users' personal data. The trained public model is passed to each individual user to make initial forecasting. The public model training improves the private model training efficiency. It also ensures the performance of the personalized forecasting in addition to achieving the purpose of data security. The public model training consists of the following steps.

4.5.1 Public Data Set Extraction

The public data extraction process is divided into four steps. **Step 1**: The QoS data of each edge region is rearranged in the order of User ID, Time slice ID, Service ID and QoS attribute Value. **Step 2**: The QoS attribute values of all users that invoke service I_1 in T_1 time slice in each region are extracted. These values are expressed in the form of I_1-T_1 column vector. The medians in statistics can reflect the middle level characteristic for a set of data. The machine learning models trained upon the medians can cater for the set of data [47]. It is not affected by extreme values. Therefore, we take the median of I_1-T_1 column vector to represent the public data of the users invoking service I_1 in each region in T_1 time slice. **Step 3**: We take the attribute values of all users invoking services I_2, I_3, ..., I_m in turn. We repeat Step 2 to get the median column vector $I_{1\to m}$-T_1 as the public data of users invoking all services in the region in T_1 time slice. **Step 4**: We take the attribute values of T_2, T_3, ..., T_n in turn. We repeat Step 2 and Step 3 to respectively get the median column vectors $I_{1\to m}$-T_2, $I_{1\to m}$-T_3, ..., $I_{1\to m}$-T_n $(m, n \geq 2)$. Finally, the n median column vectors are synthesized to obtain the $m\times n$ (service-time slice) public data set matrix, which is privacy-insensitive. At this point, the public data set extraction process is complete.

4.5.2 Public Data Set Training

Each regional public model is trained based on the regional public data set. We reach a consensus between the learning rate and the number of training iterations before the model training. The larger learning rate in a certain range will make the error adjustment speed faster. In other words, it will need less training iterations to achieve the same training effect. In contrast, the more training iterations, the more computing resources and time cost needed. Therefore, we use a set of parameter combinations (i.e., learning rate and training iterations) to measure the training effect. Generally, there is a negative correlation between them. When the learning rate increases, the number of training iterations decreases.

We use LSTM and the attention mechanism to train the public data set after tuning the parameters. Here, LSTM uses switch to control. The switch is realized in the form of a gate. Suppose W is the weight vector of the gate and b is the bias term, then the gate can be expressed as:

$$g(x) = \sigma(Wx + b) \tag{10}$$

Where σ is the sigmoid function and its value range is (0, 1).

The following are the formulas of the forget gate, the input gate, the current time cell state and the output gate.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{11}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{12}$$

Where W_f and W_i are the weight matrices, h_{t-1} and x_t respectively represent the output of the previous time slice and the input of the current time slice, b_f and b_i are the bias terms. Next, we calculate the cell state c_t in the current time slice. It is based on the aggregation between the previous cell state c_{t-1} multiplied by the forget gate f_t and the current input cell state $\tilde{c}_t$ multiplied by the input gate i_t, as shown in equation (13) and (14):

$$\tilde{c}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{13}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \tag{14}$$

W_c and b_c in equation (13) are respectively the weight matrix and the bias term of the current cell state. In equation (14), only the previous cell state c_{t-1} is included in the calculation. The cell state in the current time slice may be affected by the latest λ time slices in the prediction of time-dependent data series. However, the influence degree is positively correlated with the closeness of time slices. That is, the closer the time slice, the greater the influence degree. Therefore, we introduce the attention mechanism to further refine the calculation of the current cell state. In attention mechanism, we take the weighted sum of cell states in the previous few time slices as the historical cell states. We assign unequal initial weights to the previous few time slices, i.e., $w_{t-\lambda} = 1/(1 + 2 + ... + \lambda)$. Its structure diagram is shown in Fig. 5.

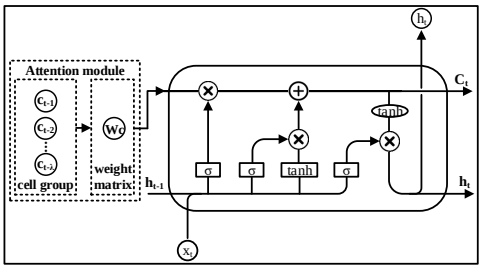


Fig. 5: The structure of LSTM with Attention Mechanism

We add the input cell group calculation to LSTM and replace c_{t-1} with s_λ. It is calculated as follows:

$$s_\lambda = W_c * C \tag{15}$$

Where W_c is the weight matrix of the cell group, $C = \left[c_{t-\lambda}, c_{t-(\lambda-1)}, ..., c_{t-1}\right]$ $(\lambda \geq 2)$. The following equation can be obtained by substituting equation (15) into equation (14):

$$c_t = f_t \circ s_\lambda + i_t \circ \tilde{c}_t \tag{16}$$

In this way, we combine LSTM's current memory $\tilde{c}_t$ and long-term memory s_λ to form a new cell state c_t. They can not only save the information in the previous time slices, but also effectively avoid the irrelevant content from entering the memory, by controlling the forget gate and the input gate. Equation (17) is the calculation process of the output gate. It controls the influence of the long-term memory on the current output. W_o and b_o are the weight matrix and the bias term of the output gate respectively.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{17}$$

The final output of LSTM is determined by the output gate and the cell state.

$$h_t = o_t \circ tanh(c_t) \tag{18}$$

We estimate the loss function by calculating $RMSE$ (Root Mean Square Error):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(h_{t_i} - h'_{t_i})^2} \tag{19}$$

Where h_{t_i} is the output value of LSTM at time slice t_i, h'_{t_i} is the real value at time slice t_i, and N is the total number of time slices.

All the parameters in the LSTM back-propagation algorithm are calculated based on the partial derivative of the loss function. These parameters are updated iteratively by the gradient descent method. The training error decreases continuously with the increasing number of training iterations. Thus, it can provide the best initial weight parameter values for personalized forecasting.

4.6 Personalized Forecasting

A user makes personalized QoS forecasting online based on LSTM and attention mechanism with the initial weight parameters passed from the public model. A private model is later trained upon a user's own QoS data (i.e. private data that is privacy-sensitive and owned by the user). In our scenario, the QoS data is generated when *Ailsa* watches the *Twitter* video using her mobile device. Only model fine-tuning is performed on the user side to save costs. The forecasting process is shown in equation (11)∼(18). The user's actual QoS is obtained from a time slice T after a QoS forecasting is made for T. We adjust the private model weight parameters with the error between the predicted and actual QoS data. The weight parameters are adjusted by the following equation:

$$weight_i = weight_0 - l * error \tag{20}$$

Where $weight_0$ is the initial weight parameter, l is the learning rate and $error = 2 * (h_{t_i} - h'_{t_i})'$ is the derivation of the prediction error.

Algorithm 1 Privacy-Aware edge QoS forecasting

Require: Mobile user u accesses edge server b in edge region r_A, followed by edge server c in edge region r_N. p_{u-A} and p_{u-N} are private data generated by u in r_A and r_N respectively. v_i are other users in edge regions r_A and r_N. s are the same services invoked by users u and v_i in the two regions in $T_{1\rightarrow i}$ time period.
Ensure: Edge QoS forecasting result for u

```
Collect QoS data of users u and v_i;
Extract the public data set d_A of r_A based on s-T_{1→i};
Extract the public data set d_N of r_N based on s-T_{1→i};
Train the public model m_A by d_A;
Train the public model m_N by d_N;
if there is new QoS data generated from user u then
    if u.location ∈ r_A then
        Use the weight parameters of m_A;
        Make personalized forecasting based on p_{u-A};
        Output edge QoS forecasting result;
    else
        Obtain the public model m_N of r_N;
        Use the weight parameters of m_N;
        Make personalized forecasting based on p_{u-N};
        Output edge QoS forecasting result;
    end if
end if
```

Based on Algorithm 1 and equation (20), the forecasting will be performed for the next time slice after completing the forecasting and training in the first time slice T. If there is actual QoS data obtained from the next time slice after the forecasting, the training will be executed again to further adjust the weight parameters. The forecasting followed by the training process will repeat until no new QoS data is obtained from the next time slice (i.e., the user terminates the service(s)). Finally we obtain the personalized edge QoS forecasting result.

5 EVALUATION

We conduct the experiments in a computer system with Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz, 16.0 GB RAM, Windows 10, Python 3.8.5 and MATLAB R2018b. We divide edge region, train public model and carry out personalized forecasting.

The evaluation process is divided into two parts to verify the usability and effectiveness of the proposed model: (1) We carry out simulation experiment based on several existing data sets; (2) We conduct real world experiments at the campus of Hohai University.

5.1 Simulation Experiment

5.1.1 Data Set Description

We use three data sets – a time series QoS data set and two edge server location data sets. These data sets can be downloaded from the data sources used in [48], [49], [50].

The first data set [2] describes the real-world QoS evaluation results from 142 users (IDs: 0-141) on 4500 Web services over 64 consecutive time slices (with a 15-minute interval between each two slices). The QoS attributes mainly include *Response Time (RT)* and *Throughput (TP)*. The second data set [3] is provided by *Shanghai Telecom*. It contains more than 7.2 million Internet accessing records of 9481 mobile phones collected from 3,233 base stations during 6 months. We use the latitude and longitude information of the base stations as the edge server positions. The third data set [4] is the radio-comms license dataset that contains the geographical locations of all the cellular base stations in Australia published by Australian Communications and Media Authority (ACMA). We use them as the locations of the edge servers.

5.1.2 Experimental Procedure

It is expected that the experiments can prove that the proposed privacy-aware edge QoS forecasting approach can achieve real-time and accurate prediction while protecting user data privacy.

First, we try to find the optimal k value for dividing the edge regions. Second, we adjust the learning rate and the number of training iterations to determine their best combination for the public model training in the subsequent experiments. Next, we analyze the variation of the error with the increased number of training iterations. We attempt to select the best initial weight parameters of a public model passing to private users to make real-time personalized forecasting. The experimental steps are described as follows:

2. https://github.com/wsdream/wsdream-dataset
3. http://sguangwang.com/TelecomDataset.html
4. https://github.com/swinedge/eua-dataset

(1) We randomly select 71 base station locations respectively from the *Shanghai Telecom* data set and the *Australian cellular base station* data set. The distribution of the cellular base stations in *Australia* is more sparse than it in *Shanghai*. Therefore, our selection is mainly concentrated in the *Melbourne* area to ensure the density of the base station locations in the experiment. Next, we label the 142 base stations with IDs from 0 to 141. We use the IDs to identify the corresponding QoS data set. Thereby, we obtain the spatio-temporal QoS data set of the two areas. There are a few null values or outliers far from the average (e.g., the *RT* value is 0s or 20s) in the QoS data set. The data presents a right skewed distribution (i.e., the RT data is more concentrated in 0s-5s and less in 5s-20s). It is inconducive to the network learning and negatively affects the training effect. We use standardization to stabilize data variance, thereby improving the convergence speed and learning efficiency of the network. The converted value is mapped to [0, 1] through normalization. Finally, the best k value is determined by elbow method (or contour coefficient). K-means is then used to realize the edge region division. The spatio-temporal QoS data set of each edge region is obtained correspondingly.

(2) We independently train a public model based on the public data set for each edge region (refer to Section 4.5.1 for the extraction process of the public data set). In this experiment, we extract the public data sets of *RT*-time slice and *TP*-time slice from each edge region. We assign unequal initial weights ranging from 1-5 to the previous fives time slices, i.e., $w_{t-5} = 1/(1+2+3+4+5)$. We try a total of 10 combinations between *learning rate* and *number of training iterations* for the public model training. The learning rate increases from 1 to 10 with an increment of 1 each time. The number of training iterations decreases from 1000 to 100 with a decrement of 100 each time. The best combination for the public model training is obtained according to the lowest error value. We train LSTM and attention mechanism with the best combination based on the public data set.

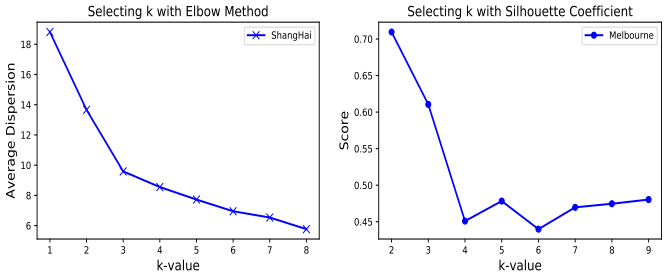
(3) We use the weight parameters provided by the public models as the initial values of the private models. We then make personalized forecasting based on the trained private models. In theory, the smaller the training interval and the higher the training frequency, the higher the accuracy of the model. However, we need to balance between training cost and training performance. Thereby, our training frequency is set to 4 time slices (i.e., 1 hour) per cycle. The private model is periodically updated based on the private data generated by a user in real time to ensure the timeliness and accuracy of the prediction.

5.1.3 Experimental Results

(1) **The optimal k value for edge region division**

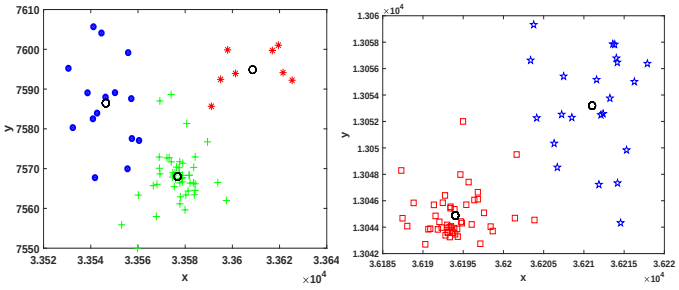
We first convert the latitude and longitude values into plane coordinates by Miller Projection. Then we respectively use elbow method and contour coefficient to determine the optimal k values. These two methods have same functions. Therefore, we can use them alternatively. If the highest point of the curvature in elbow method is not obvious, then we use contour coefficient instead.

The process of the optimal k-value determination in *Shanghai* and *Melbourne* is shown in Fig. 6(a) and Fig. 6(b). It can be seen that, the curvature is the highest, and the degree of distortion is greatly improved, when k=3 in Fig. 6(a). Thus, the optimal number of clusters is 3 for the *Shanghai* data set. The coefficient is the largest when k=2 in Fig. 6(b). This means that the clustering at this value is most effective. Hence, the optimal number of clusters for the *Melbourne* data set is 2. The server clustering results of the two areas are shown in Fig. 7(a) and Fig. 7(b). Here different legends represent different clusters. The black hollow circle is the center point of each cluster. It can be clearly seen that the servers in *Shanghai* are clustered based on three regions. The servers in *Melbourne* are clustered based on two regions.



(a) (b)

Fig. 6: Determination of the k value: (a) *Shanghai* area, (b) *Melbourne* area.



(a) (b)

Fig. 7: K-means clustering results: (a) *Shanghai* area, (b) *Melbourne* area.

The distribution of edge servers and the division of edge regions in *Shanghai* are shown in Fig. 8. We label the three regions as *Shanghai* region *SH_a*, *SH_b*, and *SH_c*. The server distribution and edge region division in *Melbourne* are shown in Fig. 9. We label them as *Mel_a* and *Mel_b*.

(2) **Impact of weight parameters of public model**

Our next experiment is to assess how different weight parameters of the public models impact training results. The default experimental learning rate is 0.001. Table 1 and Table 2 show the training errors of the two public data sets based on the weighted sum of the cell states at different historical moments. It can be seen that the minimum training errors are diverse for different QoS parameters and edge regions. For instance, the training error of the *RT* public data set of the *SH_a* region is lowest when it is based on the weighted sum of the cell states at 3 historical moments. The minimum error appears when 4 historical moments are used in the *TP* public data set of the *Mel_b* region. It can be found that the minimum errors all appear within 5 historical moments. This is because the earlier the historical moment in the time series data, the lower the influence of the data at

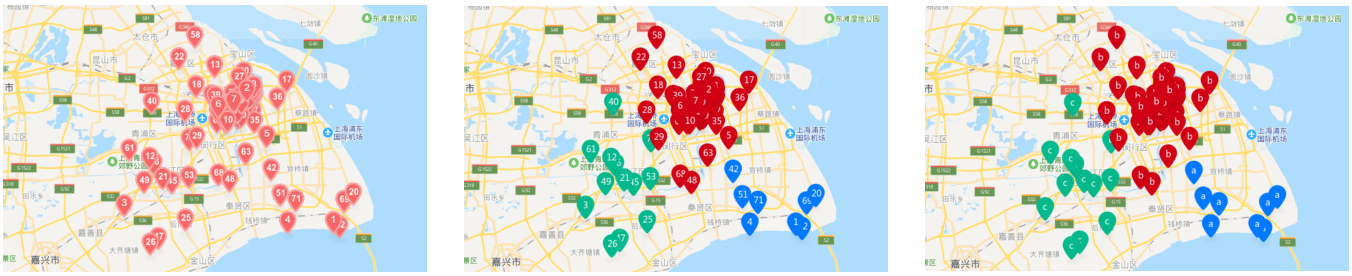


Fig. 8: The edge server distribution and edge region division of *Shanghai* area.

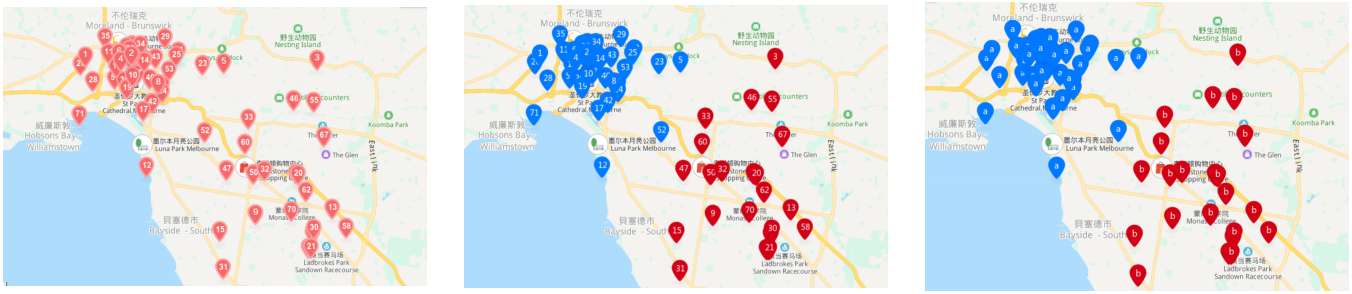
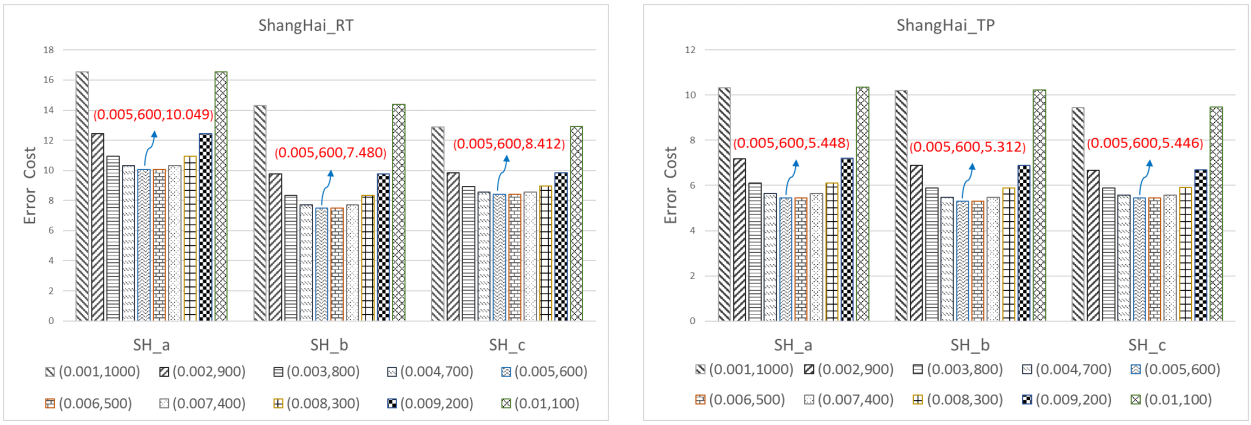
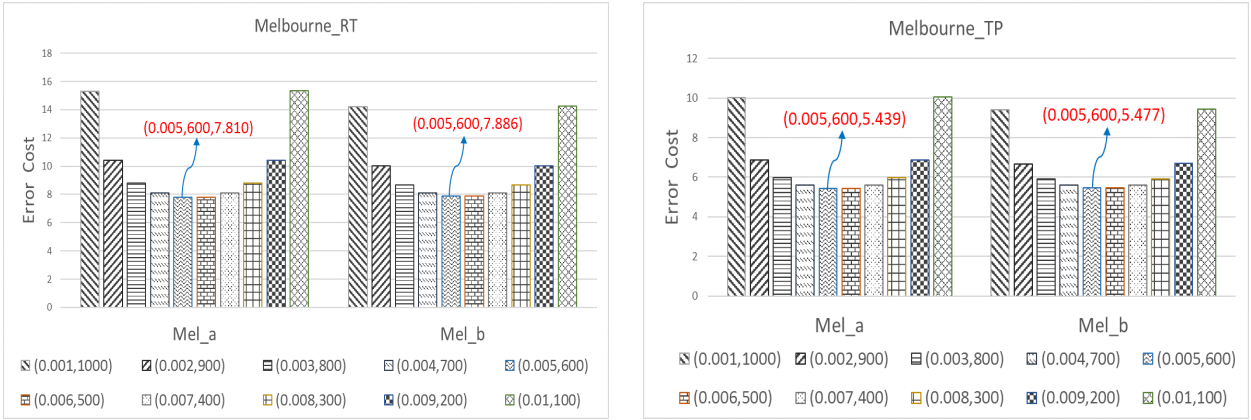


Fig. 9: The edge server distribution and edge region division of *Melbourne* area.



(a) (b)

Fig. 10: Training errors of parameter combinations in *Shanghai*: (a) *RT* public data set, (b) *TP* public data set.



(a) (b)

Fig. 11: Training errors of parameter combinations in *Melbourne*: (a) *RT* public data set, (b) *TP* public data set.

that moment on the prediction result. We use the moments with the smallest error for the public model training.

TABLE 1: Training errors in *Shanghai*

Region \ Moments		2	3	4	5
SH_a	RT	16.549	**16.532**	16.538	16.556
	TP	10.341	10.317	**10.312**	10.321
SH_b	RT	**14.328**	14.340	14.366	14.399
	TP	10.215	**10.199**	10.201	10.216
SH_c	RT	12.879	**12.872**	12.880	12.897
	TP	9.458	9.438	**9.433**	9.443

TABLE 2: Training errors in *Melbourne*

Region \ Moments		2	3	4	5
Mel_a	RT	**15.292**	15.300	15.325	15.358
	TP	10.056	**10.040**	10.043	10.058
Mel_b	RT	**14.215**	14.219	14.238	14.267
	TP	9.447	9.428	**9.425**	9.436

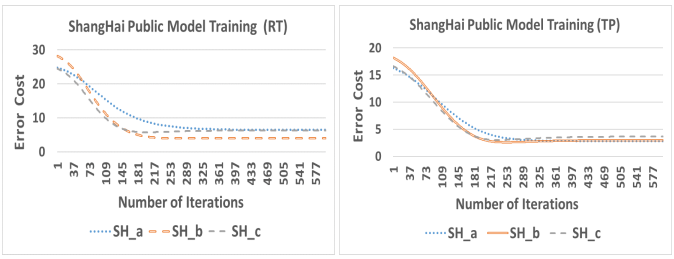
Fig. 10 and Fig. 11 are respectively the results of the public model training under different parameter combinations in the *Shanghai* and *Melbourne* data sets. It can be seen that, the trend of the training errors of the *RT* and *TP* public data sets in both of the areas is all shown as a u-shaped curve with the increased learning rate and the decreased number of training iterations. The error value reaches the minimum, when the learning rate is 0.005 and the number of training iterations is 600. At this moment, the two parameters achieve the most balanced trade-off. Therefore, the point (0.005, 600) is used as the optimal parameter combination for the public model training. The model size is 13.3MB and the training time is about 6 hours.

(3) **The initial values for private model**

The next experiment is to explore the optimal initial values that each region's public model should provide to private users for personalized forecasting.

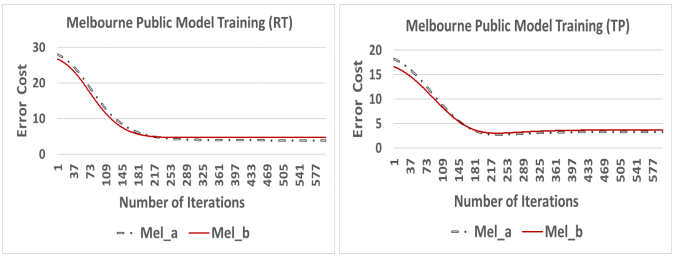
We perform training on both of the *RT* and *TP* data sets of each region. Fig. 12 and Fig. 13 show the loss function value variation of the public model training in the *Shanghai* and *Melbourne* data sets with the increasing number of iterations. The public models are trained on the optimal weight parameters (i.e., 0.005 and 600) obtained in Section 5.1.3 (2). It can be seen from Fig. 12(a) that the initial *RT* public data set training in the *SH_a* region is slightly slower. The error gradually decreases and tends to be flat, as the number of iterations increases. In Fig. 13, the training error in the *TP* data set of *Melbourne* incurs a small increase after reaching the minimum. It eventually tends to be flat. The *RT* and *TP* training errors in the other regions first show a smooth downward trend, followed by a relatively plain and stable trend in the iteration range of 300-600. There is no gradient explosion or gradient disappearance during the model training process. The training errors for the *RT* and *TP* data sets decrease from nearly 20 or 30 to 5 approximately. We accordingly use the model weight values obtained at 600 iterations as the initial parameters of the public model for users' personalized forecasting.

(4) **Personalized forecasting performance**



(a) (b)

Fig. 12: Loss function values with different training iterations in *Shanghai*: (a) *RT* data set, (b) *TP* data set model.



(a) (b)

Fig. 13: Loss function values with different training iterations in *Melbourne*: (a) *RT* data set, (b) *TP* data set.

Users access different edge servers when they move into different edge regions. Hereby, each edge server contains user access records. We choose 6 random users to access 6 edge servers in the regions of *Shanghai* and *Melbourne* respectively. Each of the 3 regions in *Shanghai* is accessed by 2 users. Each of the 2 regions in *Melbourne* is accessed by 3 users. The duration of each user's access lasts 64 time slices. The user QoS data is obtained from the accessed servers or trusted third parties. We perform two parallel experiments on the data sets of *RT* and *TP* in *Shanghai* and *Melbourne*.

The most common neural networks include Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Deep Neural Network (DNN). Our forecasting model aims to predict future QoS attribute values based on their historical values. The experimental data sets are in the form of time series. There is no correlation between different services. The RNN is the most applicable network based on such data characteristics. Thereby, we set the comparative models as follows: 1) a pure LSTM model and 2) the proposed model with different combinations of the devised strategies. The detailed descriptions of these candidate models are shown in Table 3.

TABLE 3: The four candidate models

Models	Attributes	Descriptions
LSTM	\	An RNN for learning long-term dependence on information
With-PM	1	No edge region division
Edge QoS Per-PM	1,2	Our previous model [17]
Edge-PMAM	1,2,3	The proposed model

1.Public model 2.Edge region division 3.Attention mechanism

Table 4 and Table 5 [5] are respectively the average prediction errors of the *RT* data set in *Shanghai* and the *TP* data set in *Melbourne*. Each user's private model is updated 16 times (i.e., 4 slices per training cycle). It can be seen that 1) the prediction accuracy of With-PM is far better than LSTM, proving that the public model strategy can greatly improve the prediction effect; 2) Edge QoS Per-PM and Edge-PMAM with edge region division perform better than With-PM, since each region can obtain a relatively more accurate public model; 3) the average errors of Edge-PMAM with attention mechanism is the lowest among all the candidate models, as the attention mechanism comprehensively considers the impact of several previous moments. Our experiments validate the QoS forecasting performance of Edge-PMAM.

TABLE 4: The *RT* forecasting accuracy in *Shanghai*

	SH_a		SH_b		SH_c	
Server ID	a-03	a-08	b-27	b-46	c-03	c-13
LSTM	112.418	92.762	97.615	72.707	85.567	77.215
With-PM	29.329	21.339	25.960	18.975	23.474	19.841
Edge QoS Per-PM	18.191	14.382	24.836	18.344	21.795	17.589
Edge-PMAM	**18.179**	**14.352**	**24.291**	**18.083**	**21.341**	**17.245**

TABLE 5: The *TP* forecasting accuracy in *Melbourne*

	Mel_a			Mel_b		
Server ID	a-03	a-29	a-47	b-02	b-16	b-19
LSTM	112.175	108.043	105.274	87.974	110.242	103.025
With-PM	21.522	19.714	20.047	14.829	21.301	19.337
Edge QoS Per-PM	18.644	17.105	17.389	12.450	17.979	16.483
Edge-PMAM	**18.568**	**17.027**	**17.300**	**12.303**	**17.733**	**16.209**

Table 6 and Table 7 show the forecasting time required for the four models to achieve the same forecasting performance (e.g., the average forecasting error is less than 20). The online QoS forecasting requires both training and forecasting to be conducted online. The training of Edge-PMAM private model is relatively lightweight due to the adoption of the public model. In contrast, the training of LSTM needs to start from scratch. In addition, the public models in Edge-PMAM are better trained with edge region division and attention mechanism. Therefore, the private models on top of the public models can quickly converge in comparison to those of With-PM and Edge QoS Per-PM. Thus, the time required for Edge-PMAM is less than the other three models.

In summary, our proposed Edge-PMAM forecasting approach is proved to be privacy-aware, accurate and efficient via the experiments.

5.2 Real-world Experiment

5.2.1 Scene and Data Set Description

21 students repeatedly invoked a few services from a bucket of 510 services (i.e., these services are pre-installed into the edge servers so that each student can invoke arbitrary services from them) in 6 locations of the Hohai University campus during a specific time period. Each location shares an edge server. The scenario is shown in Fig. 14. It assumes that students access services from nearby edge servers and then switch to other nearby edge servers during their movement. Most of the mobile users in the real world are believed to follow the similar pattern. We record user access time, service transmission bytes, and response time. The data set can be accessed from [6].

TABLE 6: The *RT* forecasting time cost in *Shanghai*

	SH_a		SH_b		SH_c	
Server ID	a-03	a-08	b-27	b-46	c-03	c-13
LSTM	9.517s	6.710s	6.884s	3.807s	5.494s	4.441s
With-PM	2.574s	1.835s	1.841s	1.026s	1.535s	1.192s
Edge QoS Per-PM	1.824s	1.528s	1.665s	0.752s	1.442s	0.708s
Edge-PMAM	**1.769s**	**1.519s**	**1.596s**	**0.717s**	**1.288s**	**0.701s**

TABLE 7: The *TP* forecasting time cost in *Melbourne*

	Mel_a			Mel_b		
Server ID	a-03	a-29	a-47	b-02	b-16	b-19
LSTM	9.688s	8.390s	8.186s	5.879s	9.227s	7.961s
With-PM	2.653s	2.404s	2.272s	1.708s	2.756s	2.342s
Edge QoS Per-PM	2.097s	1.833s	1.651s	1.028s	2.122s	1.905s
Edge-PMAM	**1.970s**	**1.638s**	**1.536s**	**0.946s**	**2.026s**	**1.861s**

Fig. 14: The real-world experimental scenario

5.2.2 Experimental Process and Results

All the servers are clustered into a single edge region (i.e. $k = 1$) based on our experiment. We then extract the *RT* public data set and train a public model for all the students. Finally, we make personalized forecasting for each student. The experimental results are as follows.

(1) **Impact of weight parameters of public model**

Fig. 15(a) is the result of the public model training under different parameter combinations in the *HHU-RT* data set. The error value reaches the minimum, when the learning rate is 0.005 and the number of training iterations is 600. Therefore, the point (0.005, 600) is used as the optimal parameter combination for the public model training. The model size is 0.4MB and the training time is about 5 minutes.

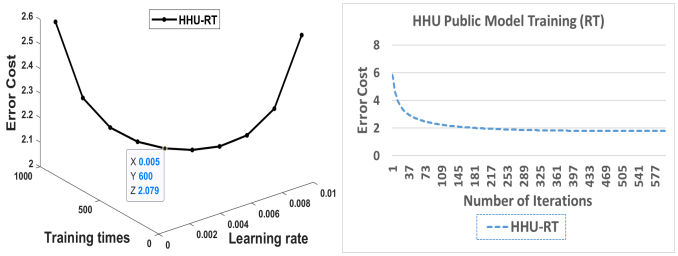
(2) **The initial values for private model**

Fig. 15(b) shows the loss function value variation of the *HHU-RT* public model training with the increasing number

5. With-PM only distinguishes Server IDs rather than edge regions in Table 4, 5, 6 and 7

6. https://github.com/hyjin1996/Hohai-University-Data-set

of iterations. We accordingly use the model weight values obtained at 600 iterations as the initial parameters of the public model for students' personalized forecasting.



(a) (b)

Fig. 15: HHU *RT*: (a) Training errors, (b) Loss and iterations.

(3) **Personalized forecasting performance**

Table 8 and Table 9 show the average prediction error and the forecasting time of the *HHU-RT* data set for the three models (the number of edge region division is 1). Each student's private model is updated 6 times, with an interval of about 8 minutes.

TABLE 8: The *HHU-RT* forecasting accuracy

Server ID	ES_1	ES_2	ES_3	ES_4	ES_5	ES_6
LSTM	16.271	15.730	14.986	15.948	16.312	14.985
Edge QoS Per-PM	1.747	1.680	1.874	1.786	1.748	1.891
Edge-PMAM	**1.744**	**1.671**	**1.864**	**1.784**	**1.744**	**1.886**

TABLE 9: The *HHU-RT* forecasting time cost

Server ID	ES_1	ES_2	ES_3	ES_4	ES_5	ES_6
LSTM (ms)	5.205	5.118	4.966	6.120	5.428	5.222
Edge QoS Per-PM (ms)	3.859	2.396	3.485	5.569	3.129	2.830
Edge-PMAM (ms)	**2.530**	**2.309**	**2.510**	**3.956**	**3.049**	**2.285**

In summary, it can be seen that the usability and effectiveness of our proposed method in the real environment.

6 CONCLUSIONS AND FUTURE WORK

Existing privacy preserving QoS forecasting approaches cannot meet the demand of the edge environment on secure and accurate QoS forecasting. We propose a novel edge QoS forecasting approach with privacy-aware named Edge-PMAM. It is based on LSTM and attention mechanism. It combines public and private model training to achieve the goal of secure, accurate and efficient forecasting.

We will work on the following directions in the future. First, the current approach only achieves a trade-off between learning rate and training iterations. Therefore, we will further optimize the public model to improve its forecasting performance. Second, we will study multivariate QoS forecasting in the edge environment. Finally, we will perform large-scale real-world experiments and release the experimental data to the public.

ACKNOWLEDGMENTS

This work is funded by the National Natural Science Foundation of China under Grant No. U21B2016, the Natural Science Foundation of Jiangsu Province under grant number BK20191297, and the Fundamental Research Funds for the Central Universities under grant number B210203070 and B210202075.

This work was also made possible by NPRP9-224-1-049 grant from the Qatar National Research Fund (a member of Qatar Foundation) and DP160100149 grant from Australian Research Council. The statements made herein are solely the responsibility of the authors.

REFERENCES

[1] L. Westerlund and J. Kaivo-oja, "Digital evolution–from information society to ubiquitous society," *Service Design: On the Evolution of Design Expertise, Lahti University of Applied Sciences Series A, Research reports, Part*, vol. 16, pp. 137–153, 2012.

[2] W.-T. Tsai, "Service-oriented system engineering: a new paradigm," in *IEEE International Workshop on Service-Oriented System Engineering (SOSE'05)*, pp. 3–6, IEEE, 2005.

[3] T. YANG and J.-d. LIU, "Survey of web services: a service-oriented distributed computing paradigm [j]," *Computer Applications*, vol. 8, 2004.

[4] A. Bouguettaya, M. Singh, M. Huhns, Q. Z. Sheng, H. Dong, Q. Yu, A. G. Neiat, S. Mistry, B. Benatallah, B. Medjahed, *et al.*, "A service computing manifesto: the next 10 years," *Communications of the ACM*, vol. 60, no. 4, pp. 64–72, 2017.

[5] S. Mistry, A. Bouguettaya, H. Dong, and A. K. Qin, "Metaheuristic optimization for long-term IaaS service composition," *IEEE Transactions on Services Computing*, vol. 11, no. 1, pp. 131–143, 2016.

[6] P. Zhang, H. Jin, H. Dong, W. Song, and L. Wang, "LA-LMRBF: Online and long-term web service QoS forecasting," *IEEE Transactions on Services Computing, 2019, DOI: 10.1109/TSC.2019.2901848*.

[7] S. Mistry, A. Bouguettaya, H. Dong, and A. K. Qin, "Predicting dynamic requests behavior in long-term IaaS service composition," in *2015 IEEE International Conference on Web Services*, pp. 49–56, IEEE, 2015.

[8] Z. Ye, S. Mistry, A. Bouguettaya, and H. Dong, "Long-term QoS-aware cloud service composition using multivariate time series analysis," *IEEE Transactions on Services Computing*, vol. 9, no. 3, pp. 382–393, 2014.

[9] S. Mistry, A. Bouguettaya, H. Dong, *et al.*, *Economic Models for Managing Cloud Services*. Springer, 2018.

[10] B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick, and D. S. Nikolopoulos, "Challenges and opportunities in edge computing," in *2016 IEEE International Conference on Smart Cloud (SmartCloud)*, pp. 20–26, IEEE, 2016.

[11] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[12] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2017.

[13] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[14] Y.-L. Huang, C.-R. Dai, F.-Y. Leu, and I. You, "A secure data encryption method employing a sequential–logic style mechanism for a cloud system," *International Journal of Web and Grid Services*, vol. 11, no. 1, pp. 102–124, 2015.

[15] S. Wang, Y. Zhao, L. Huang, J. Xu, and C.-H. Hsu, "QoS prediction for service recommendations in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 134–144, 2019.

[16] S. Liu, A. Liu, Z. Li, G. Liu, J. Xu, L. Zhao, and K. Zheng, "Privacy-preserving collaborative web services QoS prediction via differential privacy," in *Asia-Pacific Web (APWeb) and web-age information management (WAIM) joint conference on web and big data*, pp. 200–214, Springer, 2017.

[17] H. Jin, P. Zhang, and H. Dong, "Security-aware QoS forecasting in mobile edge computing based on federated learning," in *2020 IEEE International Conference on Web Services (ICWS)*, pp. 302–309, IEEE, 2020.

[18] A. Mańkowska, M. Harciarek, J. B. Williamson, and K. M. Heilman, "The influence of rightward and leftward spatial deviations of spatial attention on emotional picture recognition," *Journal of clinical and experimental neuropsychology*, vol. 40, no. 9, pp. 951–962, 2018.

[19] H. Wu, K. Yue, B. Li, B. Zhang, and C.-H. Hsu, "Collaborative QoS prediction with context-sensitive matrix factorization," *Future Generation Computer Systems*, vol. 82, pp. 669–678, 2018.

[20] S. Liu, A. Liu, Z. Li, G. Liu, J. Xu, L. Zhao, and K. Zheng, "Privacy-preserving collaborative web services QoS prediction via differential privacy," in *Asia-pacific Web*, 2017.

[21] P. Zhang, H. Jin, H. Dong, W. Song, and A. Bouguettaya, "Privacy-preserving QoS forecasting in mobile edge environments," *IEEE Transactions on Services Computing, 2020, DOI: 10.1109/TSC.2020.2977018*.

[22] X. Wang, J. Zhu, Z. Zheng, W. Song, Y. Shen, and M. R. Lyu, "A spatial-temporal QoS prediction approach for time-aware web service recommendation," *ACM Transactions on the Web (TWEB)*, vol. 10, no. 1, pp. 1–25, 2016.

[23] X. Chen, Z. Zheng, Q. Yu, and M. R. Lyu, "Web service recommendation via exploiting location and QoS information," *IEEE Transactions on Parallel and distributed systems*, vol. 25, no. 7, pp. 1913–1924, 2013.

[24] H. Polat and W. Du, "Privacy-preserving collaborative filtering using randomized perturbation techniques," in *Third IEEE International Conference on Data Mining*, pp. 625–628, IEEE, 2003.

[25] L. Li, A. Liu, Q. Li, L. Huang, W. Yang, and G. Liu, "Privacy-preserving collaborative web services QoS prediction via yao's garbled circuits and homomorphic encryption.," *J. Web Eng.*, vol. 15, no. 3&4, pp. 203–225, 2016.

[26] G. White and S. Clarke, "Short-term QoS forecasting at the edge for reliable service applications," *IEEE Transactions on Services Computing*, 2020.

[27] Y. Yin, L. Chen, Y. Xu, J. Wan, H. Zhang, and Z. Mai, "QoS prediction for service recommendation with deep feature learning in edge computing environment," *Mobile Networks and Applications*, pp. 1–11, 2019.

[28] Y. Zhang, J. Pan, L. Qi, and Q. He, "Privacy-preserving quality prediction for edge-based IoT services," *Future Generation Computer Systems*, vol. 114, pp. 336–348, 2021.

[29] Y. Zhang, P. Zhang, Y. Luo, and L. Ji, "Towards efficient, credible and privacy-preserving service QoS prediction in unreliable mobile edge environments," in *2020 International Symposium on Reliable Distributed Systems (SRDS)*, pp. 309–318, IEEE, 2020.

[30] E. Ahmed and M. H. Rehmani, "Mobile edge computing: opportunities, solutions, and challenges," 2017.

[31] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5g networks: New paradigms, scenarios, and challenges," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, 2017.

[32] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges," *Future Generation Computer Systems*, vol. 78, pp. 680–698, 2018.

[33] J. H. Saltzer and M. D. Schroeder, "The protection of information in computer systems," *Proceedings of the IEEE*, vol. 63, no. 9, pp. 1278–1308, 1975.

[34] S. Zhou, F. Li, Y. Tao, and X. Xiao, "Privacy preservation in database applications: A survey," *Chinese Journal of Computers*, vol. 32, no. 5, pp. 847–861, 2009.

[35] Y. Qian, L. Hu, J. Chen, X. Guan, M. M. Hassan, and A. Alelaiwi, "Privacy-aware service placement for mobile edge computing via federated learning," *Information Sciences*, vol. 505, pp. 562–570, 2019.

[36] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction apis," in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pp. 601–618, 2016.

[37] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. Interspeech 2014*, pp. 338–342, 2014.

[38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[39] X. Yao and B. Van Durme, "Information extraction over structured data: Question answering with freebase," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 956–966, 2014.

[40] M. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *CoRR*, vol. abs/1508.04025, 2015. [Online]. Available: http://arxiv.org/abs/1508.04025.

[41] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, pp. 577–585, 2015.

[42] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, pp. 2048–2057, 2015.

[43] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd International Conference on Learning Representations, San Diego, CA, USA, May 7-9, 2015*. [Online]. Available: http://arxiv.org/abs/1409.0473.

[44] H. Hacigümüş, B. Iyer, and S. Mehrotra, "Encrypted database integrity in database service provider model," in *IFIP World Computer Congress, TC 11*, pp. 165–174, Springer, 2002.

[45] Z. Li, R. Xie, L. Sun, and T. Huang, "A survey of mobile edge computing," *Telecommun Sci*, vol. 1, pp. 87–101, 2018.

[46] T. M. Kodinariya and P. R. Makwana, "Review on determining number of cluster in k-means clustering," *International Journal*, vol. 1, no. 6, pp. 90–95, 2013.

[47] C. Kasemtaweechok and W. Suwannik, "Training set reduction using geometric median," in *2015 15th International Symposium on Communications and Information Technologies (ISCIT)*, pp. 153–156, IEEE, 2015.

[48] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of real-world web services," *IEEE transactions on services computing*, vol. 7, no. 1, pp. 32–39, 2012.

[49] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 160–168, 2019.

[50] P. Lai, Q. He, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, and Y. Yang, "Optimal edge user allocation in edge computing with variable sized vector bin packing," in *International Conference on Service-Oriented Computing*, pp. 230–245, Springer, 2018.

Huiying Jin is a PHD candidate with the College of Computer and Information, Hohai University, Nanjing, China. She received her bachelor degree in Software Engineering from Yangzhou University, Yangzhou, China in 2017. Her current research interests include services computing and data mining. She has won National scholarship for doctoral students. She has published in international journals such as *IEEE Transactions on Services Computing* and *Information and Software Technology*.

Pengcheng Zhang received the Ph.D. degree in computer science from Southeast University in 2010. He is currently a full professor with the College of Computer and Information, Hohai University, Nanjing, China. His research interests include software engineering, services computing and data science. He has published in premiere or famous computer science journals, such as *IEEE Transactions on Big Data*, *IEEE Transactions on Cloud Computing*, *IEEE Transactions on Emerging Topics in Computing*, *IEEE Transactions on Services Computing*, *IEEE Transactions on Software Engineering*, *IEEE Transactions on Knowledge and Data Engineering*, and *ACM Transactions on Data Science*. He was the co-chair of IEEE AI Testing 2019 conference. He served as technical program committee member on various international conferences. He is a member of the IEEE.

Hai Dong is a Lecturer at School of Science in RMIT University, Melbourne, Australia. He was previously a Vice-Chancellor's Research Fellow in RMIT University. He received a PhD from Curtin University of Technology, Perth, Australia. He has published a monograph and over 80 research publications in international journals and conferences, such as Communications of the ACM, IEEE Transactions on Services Computing, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics, Journal of Computer and System Science, World Wide Web, ICSOC, ICWS, etc. He received the best paper award in ICSOC 2016. His primary research interests include: Services Computing, Distributed Systems, Cyber Security, Artificial Intelligence and Data Mining. He is a member of the IEEE and the ACM.

Yuelong Zhu is currently a Professor and a Ph.D. Supervisor with the College of Computer and Information, Hohai University, Nanjing, China. He is the Deputy Chairman of the Water Resources Informatization Special Committee of the China Hydraulic Engineering Society, and the Vice Chairman of the Jiangsu Water Resources Protection Association. His main research interests include intelligent information processing and data mining, and water conservancy informatization. He was awarded the National Second Prize for Scientific and Technological Progress, the First Prize for Excellent Achievements of the Ministry of Education, and the Dayu Water Science and Technology First Prize.

Athman Bouguettaya is Professor in the School of Computer Science at the University of Sydney, Australia. He received his PhD in Computer Science from the University of Colorado at Boulder (USA) in 1992. He was previously Science Leader in Service Computing at CSIRO ICT Centre, Canberra. Australia. Before that, he was a tenured faculty member and Program director in the Computer Science department at Virginia Tech. He is or has been on the editorial boards of several leading journals. He is a Fellow of the IEEE and a Distinguished Scientist of the ACM. including, the IEEE Transactions on Services Computing, IEEE Transactions on Knowledge and Data Engineering, ACM Transactions on Internet Technology, ACM Computing Surveys, and VLDB Journal. He has published more than 250 books, book chapters, and articles in journals and conferences in the area of databases and service computing (e.g., the IEEE TKDE, the ACM TWEB, WWW Journal, VLDB Journal, SIGMOD, ICDE, VLDB, and EDBT). He was the recipient of several federally competitive grants in Australia (e.g., ARC) and the US (e.g., NSF, NIH).