Hindawi
Mobile Information Systems
Volume 2021, Article ID 5167395, 12 pages
https://doi.org/10.1155/2021/5167395

Research Article

Daily Information Management System for Postgraduates to Check In and Out of the Dormitory Based on Mobile Edge Computing

Kewei Wen [1] **and Yong Fang** [2]

[1]*Graduate Office, Guangzhou Academy of Fine Arts, Guangzhou 510260, Guangdong, China*
[2]*Department of Student Affairs, Guangzhou Academy of Fine Arts, Guangzhou 510006, Guangdong, China*

Correspondence should be addressed to Kewei Wen; wkw@gzarts.edu.cn and Yong Fang; fy@gzarts.edu.cn

Received 13 May 2021; Revised 27 June 2021; Accepted 13 July 2021; Published 6 August 2021

Academic Editor: Sang-Bing Tsai

In recent years, the enrollment scale of graduate students has been increasing, and tutor resources are relatively scarce. Tutors often need to bring in dozens or even dozens of students, which puts a heavy burden on the tutor's work. In order to reduce the work pressure of tutors and to manage graduate students in a scientific and standardized manner, this paper has designed a daily information management system for graduate students entering and exiting the dormitory to check in and out, which realizes the management of student attendance, work, and training and is carried out to improve the quality of graduate education. In this paper, based on the actual situation of the student dormitory, we comprehensively analyze user needs, management modes, existing resources, etc. and propose a Web application system divided into fingerprint access control, attendance data manipulation, and B/S architecture. Using design and development ideas for four functional modules on the client and management side and MATLAB simulation, we compare a three-tier and two-tier processing architecture under an average resource allocation scheme in terms of model utility, model power consumption, task delay constraints, and differences in the number of users. The experiment proves that, through the test of LoadRunner software, the experimental result is the average value of multiple tests. After the analysis of the performance test result, the peak number of concurrent users of this system is 1500. The daily information management model designed in this paper for graduate students to enter and exit the dormitory shows that it is stable, easy to operate, and interface-friendly and provides preset functions. It offers a new way for graduate students to manage their daily lives and has high practical value.

1. Introduction

Edge computing is currently used to manage student check-in information. While some work is devoted to researching ways to improve the efficiency of deep learning, the above analysis shows that these solutions are more or less flawed. Therefore, edge intelligent collaborative computing and clock-in information protection methods based on deep learning research seek ways to protect junior high school students' clock-in information in edge computing while improving efficiency using deep learning mechanisms, taking advantage of deep learning to reduce mobile device overhead.

In terms of cloud computing user information management and protection, Kim and Jeong proposed a non-interactive deep learning algorithm that guarantees student check-in information. This method can ensure that check-in information is not leaked even when multiple deep learning participants collude [1]. While protecting student check-in information, Gaaloul et al. considered how to verify the correctness of the server's aggregated data and proposed a verifiable deep learning check-in information protection framework (VerifyNet) based on the analysis and solution of the above-mentioned problems. On the other hand, how to improve its communication efficiency and reduce communication overhead has gradually become an important

issue of current research [2]. de Bruin and Floridi proposed that the mobile terminal verifies the current parameter update according to the feedback information of the central server, so as to decide whether to upload its update amount to the server [3]. However, when the algorithm is executed, the mobile terminal needs additional overhead to determine whether the local update amount is related to the global convergence. Eivy and Weinman designed a control algorithm, under a given resource, according to the convergence constraints, adjusting the number of local updates and global aggregation, so as to maximize the resource utilization rate [4]. Asadi et al. improved communication efficiency by accelerating the speed of FedAvg algorithm aggregation. This method can reduce the communication overhead to a certain extent, but the implementation of the algorithm still uses the FedAvg framework and does not reduce the model learning time to the greatest extent [5].

The edge computing servers are decentralized, so they can reduce the punch-in delay. For example, Almarabeh and Majdalawi studied the cooperative caching and processing of multirate information based on mobile edge computing networks and established an integer plan with the goal of minimizing the transmission cost of the link. Model's, information content can be obtained through edge transcoding or transmission from adjacent servers [6]. Taylor et al. proposed a cooperative joint caching and processing strategy based on the X2 network interface to share information and data among multiple caching nodes, aiming to minimize the network cost of transmission in the CDN (Content Delivery Network) [7]. Abdel-Basset et al. assumed that the popularity of different information blocks of information is different. According to student preferences, a probability model is proposed. Different blocks of the same information are placed in different edge servers. Adjacent servers perform collaboration on these information blocks. Through Storage and transcoding processing, it improves the utilization of limited edge resources [8]. With the goal of minimizing the overall transmission delay, Nawrocki and Reszelewski proposed an offline information caching model based on transcoding [9]. Baldassarre et al. studied the buffer status for information streaming media applications in software-defined mobile networks and used Lyapunov's theory to convert the cache transcoding problem into a Markov decision process problem and optimize model energy consumption and student QoE [10]. The above work aims to study the optimization of the resource utilization or transmission delay of the edge cache transcoding model from the perspective of computing mode and does not consider the economic cost of resources such as student check-in scenarios and terminal calculations. The premise of the above algorithm is that local updates consume the same amount of resources. Considering different devices and different data sizes, the resources consumed by partial updates must be different, which affects the number of partial updates.

At present, there is no application model that integrates attendance and training information management, with the purpose of assisting educational affairs and daily management. According to the actual needs of a college dormitory, this paper proposes the design and development of the daily information management model for graduate students in the dormitory environment. The daily information management model for graduate students entering and exiting the dormitory based on mobile edge computing has become a professional student management model starting with the needs of dormitory management, based on ensuring the safety of the model from the ground up. Increasingly, the process of management work, the overall design of the model, the functional modular design, the design of the database, and the construction of the overall model are the amount of data required for the processing task, latency, computing capabilities, channel conditions, etc. This is carried out with comprehensive consideration of comprehensive elements.

2. Mobile Edge Computing and Daily Information Management Model

2.1. Factors Influencing the Latency of Mobile Edge Computing and IoT Devices. It is a multiend task for graduate students to enter and exit the dormitory to check in, and the statistics of the check-in results should be safely protected in the school and can be received by the school management in time. Edge computing has the advantages of distributed and highly asynchronous processing of student clock-in tasks [11]. In edge computing, servers are deployed at the edge of the network. Edge servers are closer to students, making the migration time of computing tasks shorter, so it has become the preferred platform for computing task execution [12, 13]. However, edge server resources are limited. With the increase in the number of mobile terminals, application functions become more complicated, and the queuing time when computing tasks are migrated to the edge server for processing is longer, which increases the computing delay [14]. At the same time, the computing tasks that students migrate to the server will inevitably include their own check-in information. This information is stored in the server, making students have no control over private data [15]. In the deep learning of mobile edge computing, all participants run the same machine learning model; that is, all terminals cooperate to complete the training of the model, which reduces the learning overhead of the terminal and improves the learning efficiency [16]. At the same time, the participants migrated the parameter updates generated by the calculation model instead of the original data to the server for aggregation, so that the student data is only stored locally, and there is no backup on the server, so the student's check-in information is protected [17]. After the server completes the aggregation of the updated parameters generated by the participants, it returns the results to each participant as the initial value of the parameters of the next learning model [18]. The interaction between the server and the participants and the collaboration between the participants solve the problem of data islands and at the same time avoid the negative situation caused by the instability of the network of some participants [19]. Research on edge intelligent collaborative computing and clock-in information protection methods based on deep learning applies the

mechanism of deep learning to edge computing to alleviate the leakage of student clock-in information caused by edge computing [20]. At the same time, it considers how to ensure the accuracy of data analysis, reduce the communication overhead of model aggregation, and improve the parameter update efficiency of deep learning in this mode [21].

Under different mechanisms, the relationship between the average delay of IoT devices and bandwidth is complicated. In the same area, when random allocation mechanism, weighted Tyson polygon division mechanism, and delay optimization allocation mechanism are used, the average delay of IoT devices varies with bandwidth increase and decrease [22, 23]. In the case of a certain bandwidth, the delay optimization segmentation mechanism obtained by the optimal transportation theory is smaller than the average delay achieved by the random allocation mechanism, and the average delay can be reduced [24]. In the case of a certain bandwidth, the delay optimization segmentation mechanism obtained by the optimal transportation theory is smaller than the average delay achieved by the weighted Tyson polygon division mechanism, and the average delay can be reduced [25]. In the same area, the higher the density of IoT devices at hotspots, the more users will concentrate near the hotspots and offload computing tasks onto the same edge computing service that is closest to each other. When using the weighted Tyson polygon division mechanism and the delay optimization allocation mechanism, the average latency of IoT devices increases with the increase in the density of hotspots of IoT devices. In the case of a certain density of hotspots of the Internet of Things equipment, the delay optimization segmentation mechanism obtained by the optimal transportation theory is smaller than the average delay achieved by the random allocation mechanism [26]. For constant hotspot densities of Internet of Things devices, the delay optimization segmentation mechanism obtained using optimal transport theory is less than the average delay achieved by the weighted Tyson polygon splitting mechanism. In the same area, the denser the IoT devices in the hotspot are, the more the users will be concentrated near the hotspot. Due to the constraints of the same computing power of edge computing servers, some IoT devices need to offload computing tasks onto remote edge computing servers. When the delay optimization allocation mechanism is adopted, the average power of the Internet of Things devices increases with the increase of the density of the hotspots of the Internet of Things devices [27]. In the case of a certain density of hotspots of the Internet of Things equipment, the time delay optimized segmentation mechanism obtained by the optimal transportation theory is smaller than the average power achieved by the random allocation mechanism and the average power achieved by the weighted Tyson polygon partitioning mechanism [28].

2.2. Check-In Information Management Strategy. This research believes that the check-in management platform is a completely trustworthy platform. Other devices, including all terminals and edge servers, will execute programs in accordance with the deep learning protocol, but this does not rule out the fact that they try to obtain other students' check-in information data. This setting is also widely used as the basis for studying the protection of check-in information in deep learning. Therefore, in order to prevent other devices from inverting local data according to the model parameters uploaded by the terminal, the terminal cannot directly upload the locally updated parameters to the edge server. In other words, in order to protect student punch-in information, it should be ensured that, except for the terminal itself, all devices are unaware of the local data of the terminal and the actual parameters after the local update. At the same time, the clock-in information protection mechanism also needs to ensure that the edge servers can aggregate to get the real model. This section describes how the terminal encrypts the uploaded data according to the key. In the n round of global update, suppose that the data to be uploaded after the local update of the m terminal is Rn. In deep learning, the server's global aggregation algorithm is

$$S = \frac{\sum_{m=1}^{k}\sum_{m=1}^{k}\sum_{m=1}^{n_j}\sum_{r=1}^{n_h}\left|y_{ij} - y_{hr}\right|}{2n^2u}|\mathrm{Rn}|. \tag{1}$$

That is, the aggregation algorithm is a function S of the sum of the data uploaded by all participants. After receiving the participant information transmitted by the edge server, the key generation platform generates $|\mathrm{Rn}|$ ($|\mathrm{Rn}| > 1$) random numbers, so that each participant corresponds to a random number. Then, in the key transfer phase, in addition to the random number corresponding to the participant, the platform still needs to transfer the random number corresponding to any other participant. At this time, it is necessary to ensure that all random numbers are selected and only selected once. These two random numbers are used as the key generated by the key generation platform for the terminal. On the basis of S, the terminal adds the corresponding random number and subtracts another random number to obtain the data actually transmitted to the edge server. Therefore, the edge server will not obtain the updated direct data of the students while getting the correct aggregation results. The whole process of punching card information protection mechanism is shown in Figure 1.

2.3. Edge Computing Encryption Algorithm. This article takes students using a mobile phone terminal as an example to describe the encryption process of edge computing students' check-in. The edge server first selects the participants in each round and then passes the model parameters to the participants, and the participants update the parameters locally. During the local update process, if the learning accuracy is greater than the target accuracy, the local update is stopped. Otherwise, the data needed for aggregation is passed to the edge server. After the local update of all participants ends, the edge server aggregates and generates the parameters for the next round of global update. The algorithm continues until the maximum number of global update rounds is reached or all terminals have reached the target learning accuracy. There are a total of M students using a mobile terminal each in the coverage of an edge computing server. $D = \{D1, D2, \ldots, Dm\}$ represents the set of these M mobile terminals, where Dm

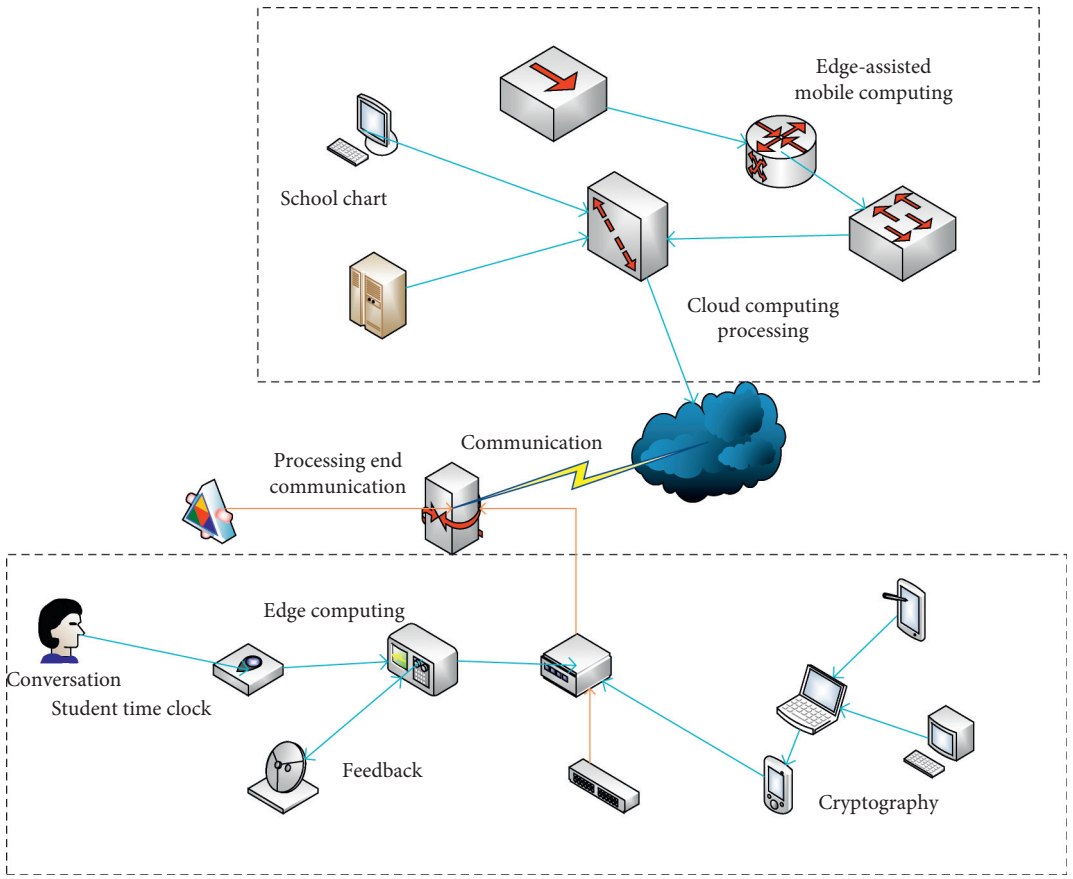

Figure 1: Frame diagram of edge computing for graduate students entering and exiting the bedroom.

represents the m mobile terminal. A total of N rounds of global updates are required to meet the deep learning requirements or the model converges to stop learning:

$$\begin{aligned} \mathrm{Dm} &= -\frac{1}{T}\ln(1+\beta), \\ \ln\left(\frac{N_{it}}{N_{it}-1}\right) &= \alpha+\beta\ln N_{it}-1+\phi X_{it}+\tau_t, \end{aligned} \tag{2}$$

where β represents the random number generated by the key generation platform relative to Dm when $\mathrm{Dm}\in \mathrm{Rn}$, N represents the random number of other participants selected by the key generation platform for Dm, and α is the key of Dm. Dm is the maximum number of iterations to run the learning model locally. Obtain the local data download model, pass the key, upload the participant U, update the parameter, and upload the data S:

$$s_j = \sum_i c_{ij}u_{(j|i)}. \tag{3}$$

At the beginning of each round of global update, the edge server determines the participating terminals of the current round, transmits model parameters to these terminals, and transmits the participating terminal information to the third-party key generation platform. The key generation platform generates the keys of these participants and transmits them to the corresponding participants. When updating locally, each participant uses local data to learn. After the learning is over, the locally updated model parameters are obtained, and the data is encrypted with a key and uploaded to the edge server. The server aggregates the uploaded data of all participants and generates the model parameters for each participant's next learning. The above process continues until the maximum number of global aggregation rounds is reached or the entire deep learning model converges. The convergence function is

$$\begin{aligned} h_t &= z_t\Theta h_{t-1}+(1-z_t)\Theta h_t, \\ P &= \sigma t = \frac{\sqrt{1/n\sum_{i=1}^{n}\left(N_{it}-N_{it}\right)^2}}{N_{it}}. \end{aligned} \tag{4}$$

Processes that have a significant impact on deep learning performance and learning efficiency include participant selection, local updates, and global aggregation. In them, the participant decides which device to choose and whether the device will participate in this round of deep learning. Local updates determine how the device runs the learning model. This directly affects the accuracy of the analysis. Global aggregation determines in the next round of a participant's learning model that the strengths and weaknesses of the aggregation method directly affect the participant's time performance, energy consumption, and learning accuracy. Therefore, this part proposes improved algorithms for the

above three processes and finally summarizes the entire deep learning algorithm process. Therefore, the goal of each local update of Dm is

$$\mathrm{Dm}_{jh} = \frac{\sum_{Z=1}^{h_j} \sum_{r=1}^{n_h} \left| y_{ji} - y_{hr} \right|}{\mathrm{Wm}_j n_h \left(\mathrm{Bm}_j + \mathrm{Dm}_h \right)}. \tag{5}$$

Among them, Wm and Bm represent the current model parameters of Dm, and y is the introduced slack variable. The above formula is further transformed into

$$\begin{aligned} \mathrm{Dm}_{nb} &= \sum_{j=2}^{k} \sum_{h=1}^{j-1} G_{jh} \left(p_j s_h + p_h s_j \right) D_{jh}, \\ \mathrm{Bm}_t &= \sum_{j=2}^{k} \sum_{h=1}^{j-1} G_{jh} \left(p_j s_h + p_h s_j \right) D_{jh} \left(1 - D_{jh} \right). \end{aligned} \tag{6}$$

Before each selection, the edge server compares the analysis accuracy and target accuracy of each terminal and selects only those terminals that do not meet the accuracy requirements. When the analysis accuracy of a terminal reaches the target accuracy, the terminal will not be selected in subsequent global updates. In this way, the accuracy of each terminal is as large as possible, and the interference with other terminals is reduced. For Dm, if the accuracy λm does not exceed the target accuracy after each learning progress and the maximum number of iterations is not reached, Dm needs to perform the next learning. Suppose that, in the n round, the model parameter of Dm is Wm and the determination of n is related to the current learning parameters and learning accuracy, as follows:

$$f(\nabla) = \frac{1}{(N/n) + \mathrm{Dm}} \sum_{i=1}^{N} k \left(\frac{X_i - x}{\mathrm{Wm}} \right). \tag{7}$$

Among them, ∇ represents the descending gradient of the loss function when the parameter is Wm. Dm represents the learning rate of Dm in this learning, and the calculation method of ηm is as follows:

$$\begin{aligned} \eta m &= \sum_{T} = \mathrm{Dm} \left(\max \left(\sigma_i - \upsilon, \theta \right) \right), \\ \mathrm{Dm}_{t1}[i] &= \sum_{j} \cos \left(w_i^1, w_j^2 \right). \end{aligned} \tag{8}$$

Among them, θ is a constant and max represents the maximum learning rate of Dm. This calculation method enables the learning rate to adapt to the current learning accuracy, avoiding the negative impact caused by the learning rate being too large or too small. The FedAvg aggregation algorithm averages the weights of all participants. The calculation expression of the model parameter W_{n+1} in the $n+1$ round is as follows:

$$\begin{aligned} W_{n+1} &= \frac{d_{jh} - n_{jh}}{d_{jh} + n_{jh}}, \\ d_{n+1} &= \int_0^{\infty} \mathrm{d}F_j(y) \int_0^{y} (\mathrm{FedAvg} - x) \mathrm{d}F_h(x). \end{aligned} \tag{9}$$

This method of parameter aggregation may cause some participants to have a lower accuracy in the next round than in the previous round, because the server considers all participants on an average during aggregation, ignoring the impact between participants. Therefore, this study uses the parameter aggregation mechanism in the q-FedSGD algorithm. The formula for calculating model parameters in the $n+1$ round is as follows:

$$\begin{aligned} D_{n+1} &= \frac{\left(1/2u_j \right) \sum_{i=1}^{n_j} \sum_{r=1}^{n_j} \left| y_{jn} - y_{jn} \right|}{n_j^2}, \\ \mathrm{Dw} &= \sum_{j=1}^{k} G_{jj} p_j \left(\frac{\mathrm{Bw}_j}{Wn} \right). \end{aligned} \tag{10}$$

Among them, Wn represents the model parameters of the n round, and the calculation expressions of K are as follows:

$$\begin{aligned} k(x) &= \frac{1}{\sqrt{2\pi}} f \left(-\frac{x^2}{2} \right), \\ Wn_t &= \tanh \left(w_c L_t + u_c \left(r_t \Theta h_{t-1} \right) + Q_c \right). \end{aligned} \tag{11}$$

Among them, Q and L are both constants. This approach makes parameter aggregation more effective. This can greatly reduce the terminal's deep learning time and energy consumption. On the other hand, considering the limitations of terminal computing resources and storage resources, in future related work, deep learning will be further combined with edge computing to minimize terminal overhead. At the same time, more datasets will be used to train FLBEC methods and verify their efficiency. Simultaneously, we will further optimize the check-in information protection mechanism proposed in this study and design a more complete mechanism based on the deep learning framework that protects student check-in information in edge computing.

3. Experimental Design of Daily Information Management Model Based on Mobile Edge Computing

3.1. Experimental Objects and Simulation Indicators

3.1.1. Subject. This experiment considers a multiuser single-task processing scenario. Tasks can be processed locally or reprinted to SBS or MBS for processing. The Poisson point distribution is used to describe the geographical distribution of users, SBS, and MBS. Four SBSs are set at a radius of 500 m. The 500 m circle is located on the 4 arcs in the north, south, west, and east directions, and users are distributed in a band-shaped arc composed of a large circle with a radius of 800 m and a small circle with a radius of 500 m. N is set in this article to 100 graduate students, and all users are within the coverage of MBS and all SBS. The three-tier processing architecture of the mobile edge network used in this article has more advantages than the two-tier processing architecture. MATLAB simulation is used

to compare the three-tier processing architecture and the two-tier processing architecture under the average resource allocation scheme in terms of model utility and model power consumption. The index varies with the model's power consumption weight, task delay constraint, and the number of users.

3.1.2. Simulation Index. Considering the two simulation indicators, the utility of the model and the power consumption of the model the model utility is the value of the objective function. The objective function also shows that the utility of the model consists of two parts: the total number of tasks completed by the model within a particular constraint range and the total power consumption of the weighted model. The model power consumption metric shows the total power consumption of the entire model from the start of task processing, including transmission, to the completion of task processing.

3.2. Architecture Used by the Model. According to the research significance and demand analysis of this article, the daily information management model for graduate students needs to meet the characteristic of not limited to the operation model, and the model can be accessed anytime and anywhere, to ensure the security, unified management, and easy scalability of the model. Considering the characteristics of the C/S structure and the B/S structure, it is finally determined that the B/S structure is the best choice for the model. The model framework is a three-tier system structure, namely, client, application server, and database server. The model system structure diagram is shown in Figure 2.

3.3. Model Network Topology. This model is based on the existing campus LAN, mobile device wireless network, and other resources; serves relatively few and scattered objects; and requires fast data access functions. Therefore, this model uses a distributed network topology, as shown in Figure 3.

3.4. Model Module Division. According to the demand analysis of the model, the functional division of the model is one of the important methods of software design. The functional module division of this model is shown in Figure 4.

The model can be divided into student card access control module, model management module, client module, and attendance data storage module. Among them, the student card access control module is a mature technology, and the client module is further divided into a tutor function module and a student function module.

3.5. Work Flow Analysis. The software part of this model is a Web service model including the attendance data operation part and the B/S architecture. The hardware part specifically refers to the student card access control multifunction machine. The model composition diagram is shown in Figure 5.

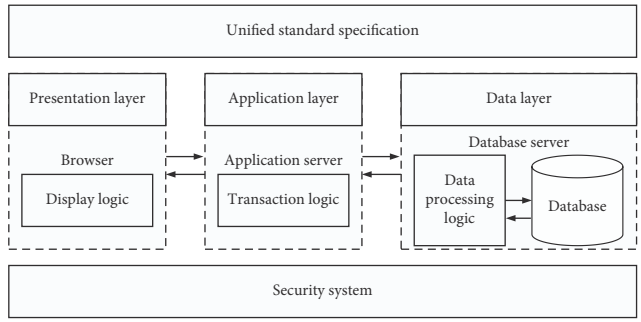

Figure 2: System architecture diagram.

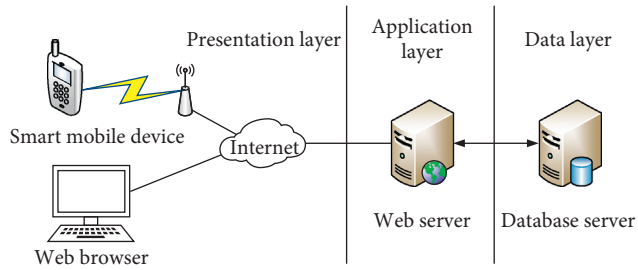

Figure 3: System network structure diagram.

3.5.1. Student Card Access Control Module. The student card access control module is composed of a student card access control machine. It can collect the attendance information of all users through verification methods such as student cards, implement the entrance student card and card management mode, record the user sign-in time, and then store the collected attendance information in the student card access control machine.

3.5.2. Attendance Data Operation Module. This module reads the attendance record from the access control part of the student card and transfers it to the database model.

3.5.3. Web Service Model Module. The functional design of this part is mainly considered at two levels: one is divided into external and internal functions, and the other is that users have tutors, students, and administrators. For all users, the internal function is the same, that is, interacting with the database through the filtering of the search conditions, and extracting the corresponding data information from the database. External functions reflect the permissions of various users.

This module can be further divided into client and management. The client includes two types of users: tutors and students. The management is designed for model managers. The external functions of the tutor include issuing training programs or notices, processing student leave information, and executing training progress; the external functions of students include submitting study progress, applying for holidays, etc.; the external function of the model administrator is to add new users, model settings, etc.

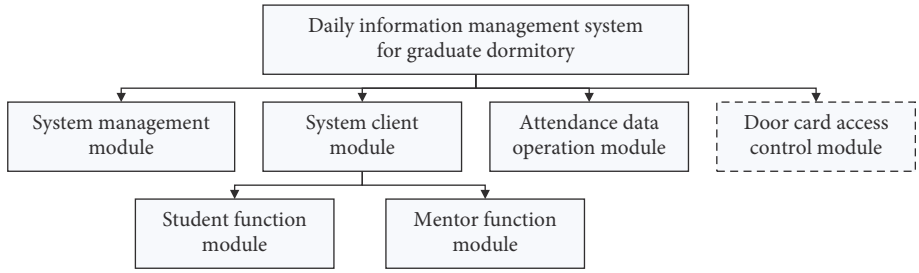

Figure 4: System overall module diagram.

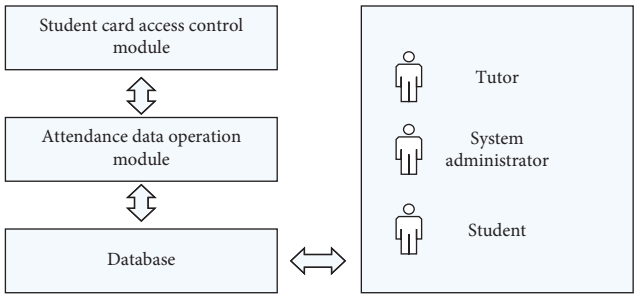

Figure 5: System structure diagram.

3.6. Statistical Processing. Statistical analysis was performed with SPSS 13.0 statistical software. One-way analysis of variance was used to test the significance of differences, LSD-t test was used to test the difference between the two groups, and the daily information results statistics of graduate students entering and exiting the bedroom were statistically performed by group t test. $P < 0.05$ is considered to be statistically significant.

4. Experimental Daily Information Management Model Based on Mobile Edge Computing

4.1. Simulation Performance Analysis. Figure 6 shows the trend curve of the utility of the three-tier processing architecture and the two-tier processing architecture model with the power consumption weight using the average resource allocation scheme.

From Figure 6, the model utilities for the 3-layer task processing architecture consisting of local & SBS & MBS and the 2-layer task processing architecture consisting of local & SBS and local & MBS both increase the power consumption weight of the model. This is due to the power consumption of the model. The weight represents the proportion of the model's power consumption in the model's utility. An increase in the weight means an increase in the model's power consumption, which leads to a decrease in the corresponding model's utility. Moreover, we can also see from the figure that the model of local & SBS & MBS is the most useful, followed by local & SBS, and finally local & MBS. This fully shows that the three-tier task processing architecture has more advantages than the two-tier task processing architecture in terms of model utility indicators.

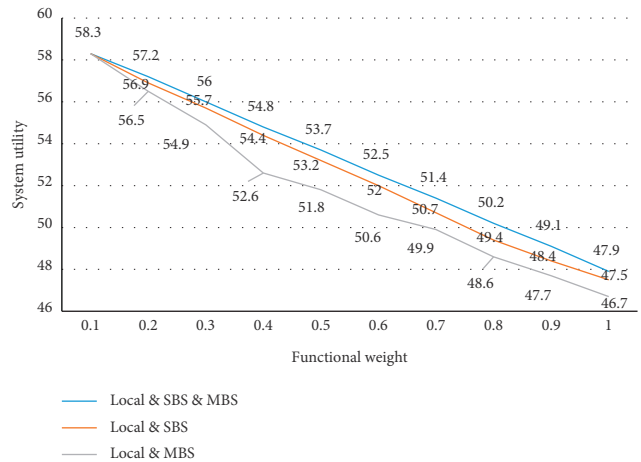

Figure 6: Graph of utility versus power consumption weight.

It can be seen from Figure 7 that the model power consumption of the three-tier task processing architecture and the two-tier task processing architecture decreases as the power consumption weight increases. This is because the larger the power consumption weight, the greater the value of the model power consumption. From the overall trend of the curve, we can also see that the power consumption curve of the local & SBS & MBS model is below the power consumption curve of the local & SBS and local & MBS models, which shows that the power consumed by the three-tier processing architecture model is higher than that of the two-tier architecture. Among them, in the two-layer processing architecture, local & SBS consumes less model power than local & MBS, because MBS is farther away from users, which tends to cause larger transmission power consumption.

Figure 8 shows the utility of the three-tier and two-tier processing architecture model using the average resource allocation scheme with the change trend curve of the task delay constraint. In order to fully illustrate the difference between the two architectures, the time delay range selected in the figure exceeds the time required to complete the task, so the model utility of the two frameworks is relatively constant over the entire time delay range.

It can be seen from Figure 8 that the three-layer task processing architecture composed of local & SBS & MBS has a model utility of about 47.5 within the entire task delay constraint range (5 seconds to 15 seconds), while the model utility of local & SBS is about 47, and the model utility of local & MBS is about 47. The model utility is about 46.0. This

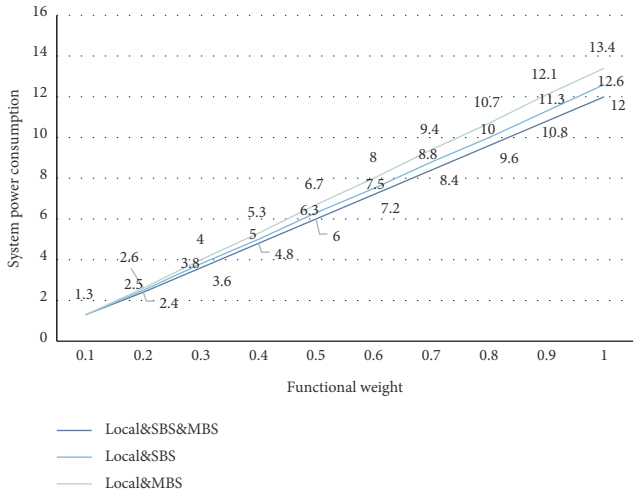

Figure 7: Graph of system power consumption varying with energy consumption weight.

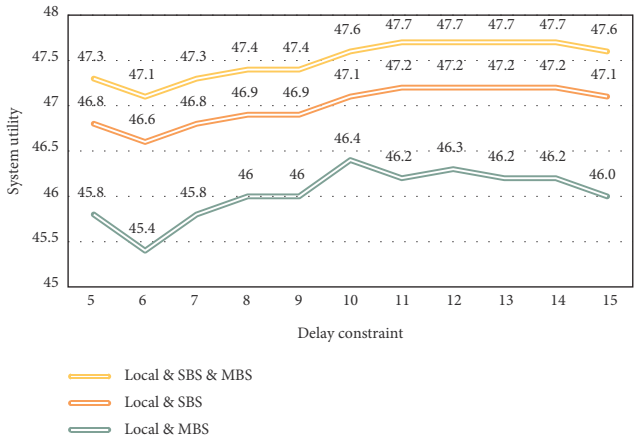

Figure 8: Graph of the system utility changing with the task delay constraint.

result fully demonstrates that the three-tier task processing architecture used in this paper has higher model utility than the two-tier processing architecture for tasks under different delay constraints. This is because the MEC Server is deployed at the SBS and MBS for the three-tier tasks, which provides users with more computing resources and can also meet the needs of tasks with different delay constraints.

Figure 9 shows the trend curve of the power consumption of the three-tier and two-tier processing architecture models using the average resource allocation scheme as a function of the task delay constraint. The time value of the abscissa in the figure has met the completion time of the task in the model, so even if the delay constraint increases, the model power consumption of the two processing frameworks is still relatively stable.

It can be seen from Figure 9 that the model of the local & SBS & MBS processing architecture used in this article has the smallest power consumption within the entire time delay constraint, fluctuating around 12.5 joules, and the model power consumption of the local & SBS processing

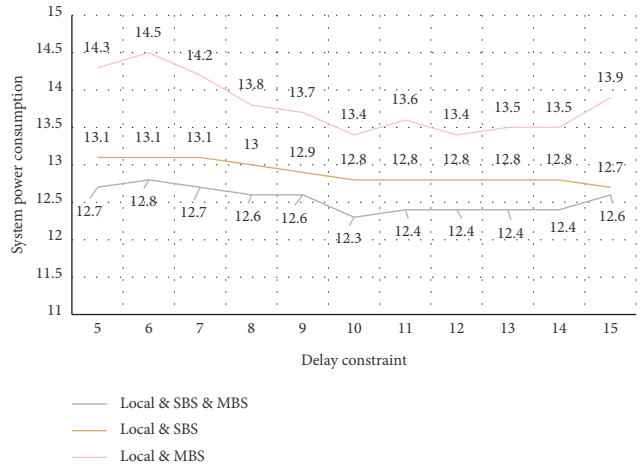

Figure 9: Graph of system power consumption varying with task delay constraints.

architecture is the second, fluctuating around 13 joules. The local & MBS model has the largest power consumption, fluctuating around 14 joules.

The utility of the three-tier and two-tier processing architecture model using the average resource allocation scheme varies with the number of users in the model as shown in Figure 10.

It can be seen from Figure 10 that when the number of users is in the range of 10 to 60, the model utility of the three-tier processing architecture (local & SBS & MBS) and the local & MBS in the two-tier processing architecture is greater than that of the local & SBS. When the number of users is 61 to 100, when the scope changes, the utility value of the three-tier task processing architecture and the local &SBS model is greater than that of the local & MBS, because from the perspective of the entire scope, the model utility curve diagram of the three-tier task processing architecture works best. The essential reason is that more MEC Servers are arranged in the three-tier task processing architecture, which has stronger computing power and can meet the computing needs of more users.

From Figure 11, we can see that when the number of users is 10, the power consumption of the 3-tier task processing architecture model is very low and close to 0. As the number of users varies from 10 to 60, the two-tier processing architecture model consisting of local & SBS consumes more power than the three-tier processing architecture. When the number of users varies from 61 to 100, the power consumption of the local & SBS & MBS model is lower than that of the local & MBS. Among them, when the users are 100, the local & SBS & MBS model power consumption is 20 joules, while the local & MBS model consumes up to 50 joules. On the whole, the local & SBS & MBS has the smallest power consumption in the entire user range.

4.2. Model Test

4.2.1. Test Environment. The test environment of this article is shown in Table 1.

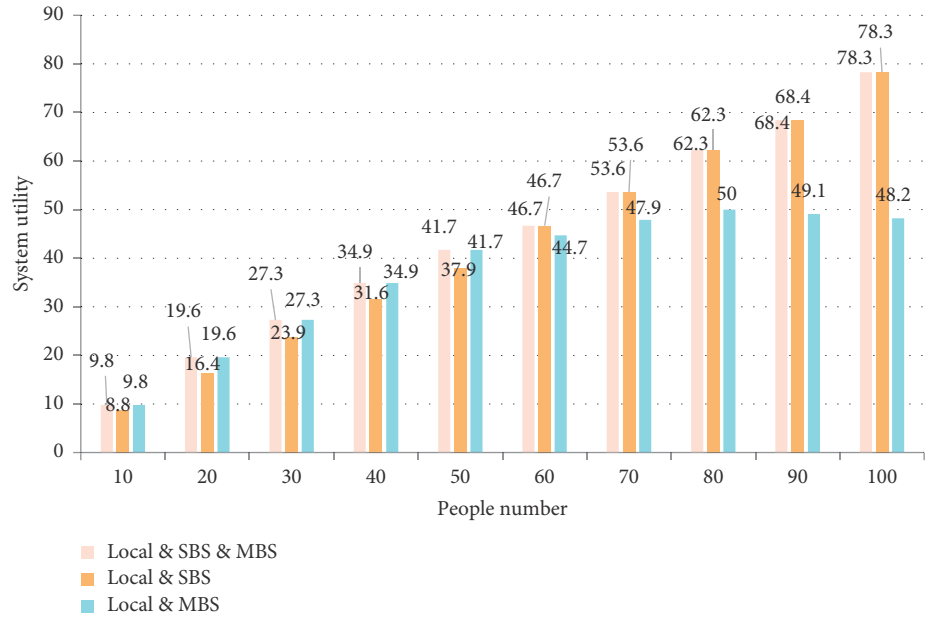

Figure 10: Graph of the system utility changing with the number of users.

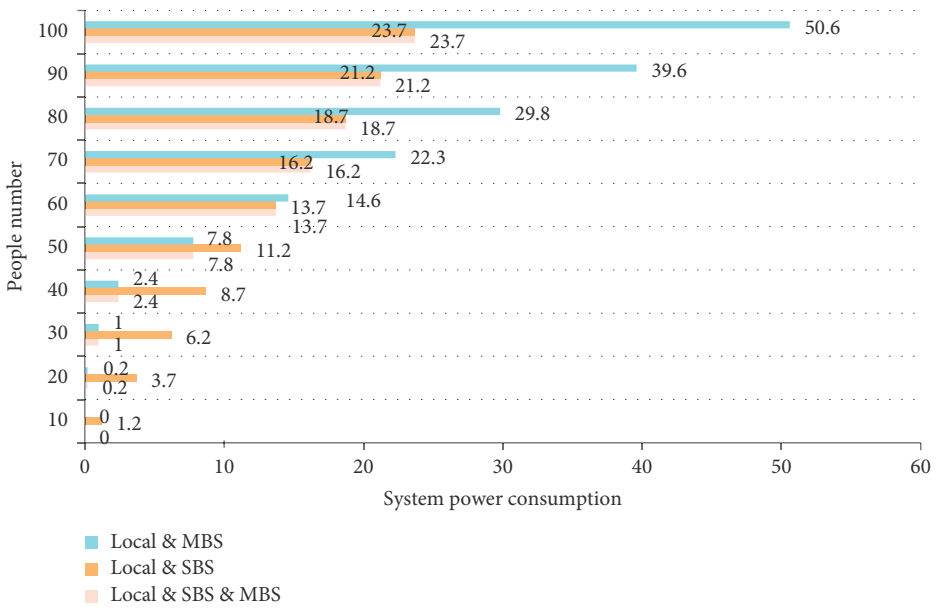

Figure 11: Graph of system power consumption varying with the number of users.

Table 1: Environmental information sheet for testing.

The software and hardware configuration of the system server	Operating system	Windows Server 2003
	Database	MySQL 5.5.8
	Web server	Apache 2.2.17
	Easy language software	Easy language 5.11
Software and hardware configuration information of the client PC	Operating system	Win 7/Win XP
	Browser	IE/360/Firefox

In the testing process, design reasonable test cases based on demand analysis, repeat each test at least five times, and focus on the parts with more errors.

4.2.2. Attendance Data Operation Module Test. Participants of the attendance data operation module have software written in Yi Lang, fingerprint access control model, and MySQL database.

Table 2: Easy language partial test cases.

Operation description	Data	Desired results	Actual results
Connect equipment and database; read attendance records	Attendance data	Ability to view user attendance information	Successful
Change the time setting of timer 1	8 o'clock to 18 o'clock	Ability to view the status of Tiananmen	Successful
Change the time setting of timer 2	Operation interval is set to 5 minutes	Ability to view the actual interval of reading data and data information	Successful
Clear attendance data	Attendance records of all users	User attendance data reset	Successful

Table 3: Page content test case.

Operation description	Data	Desired results	Actual results
Mouse over every object	E-mail sending page input box	When the mouse rolls over the object, the color of the object changes	Successful
Search for a line of information	Search all student users in the table	List all student information in the system	Successful

Table 4: Link test result table.

Link test	Whether the page exists	Whether it opens smoothly
Backstage login	✓	✓
System homepage	✓	✓
View a student's timetable	✓	✓

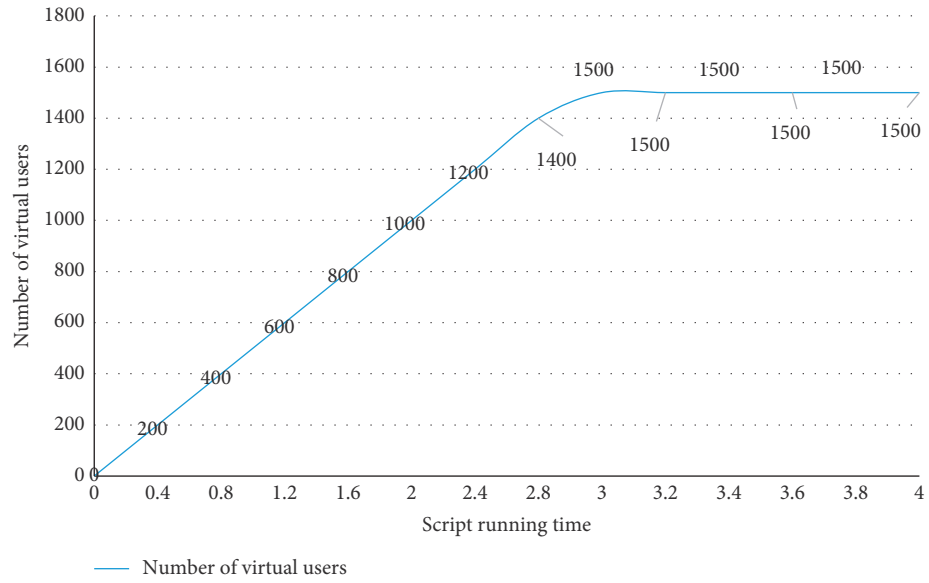

Figure 12: Software design execution effect.

The fingerprint access control model is a mature technology and does not need to be tested separately. The easy language part reads the data and stores it in the database, so it is very important that this part works normally. After the E-Language part is successfully connected to the device and database, operations such as reading attendance data are performed at regular intervals. Table 2 shows the test case of this part.

4.2.3. Functional Test. Content testing can check whether the information provided by the application model meets the correctness, accuracy, and content relevance. The test example for this part of this model is shown in Table 3.

Using links in the Web can provide users with convenient page conversion functions. The test results of this model are shown in Table 4.

In addition, after a complete page test, there are no isolated pages in this model.

4.2.4. Performance Test. The performance test can obtain the software design execution effect through the test, which reflects the stability and reliability of the model operation. This test passes the LoadRunner software test, and the experimental results are the average of multiple tests. After the analysis of the performance test results, the peak number of concurrent users of this model is 1500, as shown in Figure 12.

5. Conclusions

The mobile edge network reduces the energy consumption of mobile devices and extends the battery life of mobile devices by providing nearby computing resources for mobile devices. It has received extensive attention from academia and education. However, most research work only considers the two-tier processing architecture of tasks in mobile edge network computing scenarios and rarely considers the three-tier processing architecture of tasks. The task three-tier processing architecture adds a layer of processing architecture to the two-tier one, so it can meet the task offloading needs of more users. This design is based on the current status of graduate education and the daily information management of a college dormitory. It is proposed with the aim of sharing tutor tasks and supervising students to ensure the quality of graduate education. This model adopts modular design ideas for the needs of real users and management modes. With fingerprint access control model, campus LAN, and Apache server support, you can connect to your MySQL database using the PHP scripting language, a simple Chinese logical programming language, and complete the implementation of your application-based Web management model. This model implements the basic functions of day-to-day information management for graduate student dormitory check-in, but due to time and academic level restrictions, there are some issues that need to be resolved in the future. The model can control the mode of operation of a simple language module through a MySQL database instead of the current independent mode of operation for this part.

Data Availability

No data were used to support this study.

Conflicts of Interest

There are no potential conflicts of interest regarding the publication of this paper.

Authors' Contributions

The authors have read the manuscript and approved its submission to the journal.

Acknowledgments

This work was supported by "innovation practice research on the system and mechanism of ideological and political education of students in the new era" in the form of the project library construction system and mechanism of Guangzhou Academy of Fine Arts in 2019.

References

[1] H.-W. Kim and Y.-S. Jeong, "Secure authentication-management human-centric scheme for trusting personal resource information on mobile cloud computing with blockchain," *Human-Centric Computing and Information Sciences*, vol. 8, no. 1, pp. 11–15, 2018.

[2] W. Gaaloul, Z. Zhou, H. Panetto, and L. Zhang, "Special issue on fog and cloud computing for cooperative information system management: challenges and opportunities," *Future Generation Computer Systems*, vol. 109, pp. 704-705, 2020.

[3] B. de Bruin and L. Floridi, "The ethics of cloud computing," *Science and Engineering Ethics*, vol. 23, no. 1, pp. 21–39, 2017.

[4] A. Eivy and J. Weinman, "Be wary of the economics of "serverless" cloud computing," *IEEE Cloud Computing*, vol. 4, no. 2, pp. 6–12, 2017.

[5] S. Asadi, M. Nilashi, A. Husin et al., "Customers perspectives on adoption of cloud computing in banking sector," *Information Technology & Management*, vol. 18, no. 4, pp. 1–26, 2017.

[6] T. Almarabeh and Y. K. Majdalawi, "Cloud computing of E-learning," *Modern Applied Science*, vol. 12, no. 8, pp. 85–89, 2018.

[7] R. P. Taylor, C. Cordeiro, D. Giordano et al., "Consolidation of cloud computing in ATLAS," *Journal of Physics Conference Series*, vol. 898, no. 5, pp. 52–58, 2017.

[8] M. Abdel-Basset, M. Mohamed, and V. Chang, "NMCDA: a framework for evaluating cloud computing services," *Future Generation Computer Systems*, vol. 86, no. 9, pp. 12–29, 2018.

[9] P. Nawrocki and W. Reszelewski, "Resource usage optimization in mobile cloud computing," *Computer Communications*, vol. 99, no. 4, pp. 1–12, 2017.

[10] M. T. Baldassarre, D. Caivano, G. Dimauro, E. Gentile, and G. Visaggio, "Cloud computing for education: a systematic mapping study," *IEEE Transactions on Education*, vol. 61, no. 3, pp. 234–244, 2018.

[11] P. K. Senyo, E. Addae, and R. Boateng, "Cloud computing research: a review of research themes, frameworks, methods and future research directions," *International Journal of Information Management*, vol. 38, no. 1, pp. 128–139, 2018.

[12] Y. Wang, Z.-Y. Ru, K. Wang, and P.-Q. Huang, "Joint deployment and task scheduling optimization for large-scale mobile users in multi-UAV-enabled mobile edge computing," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3984–3997, 2020.

[13] Z. Lv, D. Chen, R. Lou, and Q. Wang, "Intelligent edge computing based on machine learning for smart city," *Future Generation Computer Systems*, vol. 115, pp. 90–99, 2021.

[14] R. Babin, "Cloud computing e-communication services in the university environment," *Journal of Information Systems Education*, vol. 15, no. 1, pp. 55–67, 2017.

[15] W. Xiong, Z. Lu, B. Li et al., "A self-adaptive approach to service deployment under mobile edge computing for autonomous driving," *Engineering Applications of Artificial Intelligence*, vol. 81, pp. 397–407, 2019.

[16] R. Varbanov, "Challenges and risks in companies' transition to cloud computing," *Economics*, vol. 1, no. 2, pp. 43–46, 2017.
[17] K. Shankar and M. Elhoseny, "Trust based cluster head election of secure message transmission in MANET using multi secure protocol with TDES," *Journal of Universal Computer Science*, vol. 25, no. 10, pp. 1221–1239, 2019.
[18] G. A. Gravvanis, J. P. Morrison, D. Petcu, T. Lynn, and C. K. Filelis-Papadopoulos, "Special issue: recent trends in cloud computing," *Future Generation Computer Systems*, vol. 79, no. 2, pp. 700–702, 2018.
[19] Z. Wang, N. Wang, X. Su, and S. Ge, "An empirical study on business analytics affordances enhancing the management of cloud computing data security," *International Journal of Information Management*, vol. 50, no. 2, pp. 387–394, 2020.
[20] Y. Liu, S. Dong, J. Wei et al., "Assessing cloud computing value in firms through socio-technical determinants," *Information & Management*, vol. 57, no. 8, pp. 368-369, 2020.
[21] L. Ding, Z. Wang, X. Wang, and D. Wu, "Security information transmission algorithms for IoT based on cloud computing," *Computer Communications*, vol. 155, no. 5, pp. 32–39, 2020.
[22] D. Schneckenberg, J. Benitez, C. Klos et al., "Value creation and appropriation of software vendors: a digital innovation model for cloud computing," *Information & Management*, vol. 8, no. 3, pp. 461–463, 2021.
[23] Y. Sun, H. Song, A. J. Jara, and R. Bie, "Internet of Things and big data analytics for smart and connected communities," *IEEE Access*, vol. 4, pp. 766–773, 2016.
[24] S. Ouf and M. Nasr, "New generation cloud computing," *Journal of Statistical Software*, vol. 66, no. 2, pp. 53–61, 2020.
[25] Z. Qian, D. Gu, C. Liang, and Y. Fang, "Impact of a firm's physical and knowledge capital intensities on its selection of a cloud computing deployment model – science direct," *Information & Management*, vol. 57, no. 7, pp. 87–89, 2019.
[26] A. Gz and B. Mnr, "Exploring vendor capabilities in the cloud environment: a case study of Alibaba cloud computing," *Information & Management*, vol. 56, no. 3, pp. 343–355, 2019.
[27] S. Tripathi, "Determinants of cloud computing intentions to use: role of firm's size, managerial structure and industrial sector," *Journal of International Technology and Information Management*, vol. 28, no. 2, p. 3, 2019.
[28] Y. Karaca, M. Moonis, Y.-D. Zhang, and C. Gezgez, "Mobile cloud computing based stroke healthcare system," *International Journal of Information Management*, vol. 45, no. 45, pp. 250–261, 2019.