Int. j. inf. tecnol. (June 2022) 14(4):2093–2103
https://doi.org/10.1007/s41870-022-00922-z

ORIGINAL RESEARCH

Novel DLSNNC and SBS based framework for improving QoS in healthcare-IoT applications

Jyotsna[1] · **Parma Nand**[1]

Received: 9 December 2021 / Accepted: 30 March 2022 / Published online: 20 April 2022

Abstract Health care system is intended to enhance one's health and as a result, one's quality of life. In order to fulfil its social commitment, health care must focus on producing social profit to sustain itself. Also, due to ever increasing demand of healthcare sector, there is drastic rise in the amount of patient data that is produced and needs to be stored for long duration for clinical reference. The risk of patient data being lost due to a data centre failure can be minimized by including a fog layer into the cloud computing architecture. Furthermore, the burden of such data produced is stored on the cloud. In order to increase service quality, we introduce fog computing based on deep learning sigmoid-based neural network clustering (DLSNNC) and score-based scheduling (SBS). Fog computing begins by collecting and storing healthcare data on the cloud layer, using data collected through sensors. Deep learning sigmoid based neural network clustering and score based Scheduling approaches are used to determine entropy for each fog node in the fog layer. Sensors collect data and send it to the fog layer, while the cloud computing tier is responsible for monitoring the healthcare system. The exploratory findings show promising results in terms of end-to-end latency and network utilization. Also, the proposed system outperforms the existing techniques in terms of average delay.

Keywords Entropy · Clustering · Neural network · Fog computing · Score-based scheduling

✉ Jyotsna
jyotsna.seth@gmail.com

[1] Department of Computer Science, Sharda University, Greater Noida, Uttar Pradesh, India

1 Introduction

Internet of Things principles can improve patients' health and welfare by increasing the availability and quality of care, as well as significantly reducing treatment expenses and frequent travel [1]. The Internet of Medical Things (IoMT) is a digital healthcare system that connects patients to medical resources and services [2]. Wireless sensor networks are becoming more pervasive and easy-to-use enabling technology for structural health monitoring than current wired systems [3]. Patients can use smart wearable devices with sensors that come with smartphones to gather data about their health status such as heart rate, glucose level and blood pressure [4]. The analysis and processing of data is done by cloud servers. Moreover, cloud computing is the most likely practical approach for connecting IoT with healthcare [5]. Patient data may be used not only to monitor a patient's present health, but also to forecast future medical concerns using cloud big data storage and machine learning techniques [6]. But patient's physical condition changes over time, demanding quick action to monitor remote patients. And cloud mechanism lacks to handle the real time application and cannot meet the requirements of quality-of-service (QoS). There is need of system that can continually and quickly monitor the report on the patient's condition [7].

The introduction of fog computing in healthcare applications is to bridge the gap between IoT devices and analytics [8]. Fog computing is a distributed computing platform for managing applications and services at the network edge [9]. The probability of a mistake and the delay increases as the volume of data transmitted over the network grows. Data packet loss and transmission latency are directly proportional to the amount of data transported by IoT devices to the cloud. The Edge or Fog paradigm

overcomes problems like latency by placing small servers known as edge servers in close proximity to end user devices [10]. A fog-based IoT system comprises of three layers: device, fog, and cloud. It has been hailed as a promising paradigm for lowering networking infrastructure and processor energy consumption while offering cloud-like health monitoring services [11]. The number of fog-based applications is expanding and is thought to outweigh IoT apps in near future [12]. IoT technology in healthcare can enhance the quality as well as affordability of medical treatment by automating formerly manual activities [13].

Fog provides storage and processing capabilities more accessible to end-users. Fog can capture, analyse, and store massive amounts of data in real-time [14]. Because medical sensors collect data on a frequent basis, real-time analysis performance might be enhanced, enabling intelligent data analysis and decision-making based on end-user rules and network resources. [15, 16].

The following are the main contributions of this work:

Fog computing uses deep learning sigmoid based neural network clustering and score based scheduling to calculate entropy values for each fog node and thus to improve the quality of service in fog based architecture.

The manuscript's structure is organized as follows: Sect. 2 examines the existing literature on the proposed strategy. Section 3 provides a brief overview of the proposed system, Sect. 4 explores the exploratory findings, and Sect. 5 concludes the article.

2 Related works

The quality of service is determined by resource allocation and load balancing in cloud/fog computing. Fog-based architectures have been proposed by many researchers for a variety of applications. Table 1 presents an overview of existing Fog literature surveys relevant to our work.

The support for real-time applications is a major reason for the fog computing architecture emergence. There are several QoS metrics to consider, including latency, bandwidth, energy consumption reduction, and cost minimization for the successful development of fog-based system.

3 Proposed methodology

The three tiers of computing is cloud computing, fog computing, and sensors, which all communicate with one another. The primary purpose of the proposed technique is to present three-tier architecture for context and latency-sensitive monitoring systems. In this paper, we propose that fog computing can be utilized to assist in the monitoring of patients' healthcare data, ensuring that data is gathered and evaluated efficiently. Sensors are used to collect data from patients at first. Both external and internal data are recorded by these sensors. The role of sensors is to gather and transmit all data to the fog computing layer. Fog computing then uses Deep Learning Sigmoid based Neural Network Clustering and Score based Scheduling to get the entropy value for each fog node. This layer analyses the data and information collected by the edge devices. The layer functions similarly to the server. In addition, the cloud-computing tier constantly checks the health monitoring system as shown in Fig. 1.

To resolve jobs in a more qualified manner or to implement a range of strategies in order to reach a better result, the neural network must always learn. When it receives new information from the system, it learns how to respond to a new circumstance. A deep neural network is a sort of machine learning in which the system uses numerous layers of nodes to extract high-level functions from input data. It requires converting numerical data into a more abstract and artistic form. Convolution, Sigmoid-based normalisation, pooling, and a fully connected layer are among the suggested DLSNN layers that solve CNN problems. Figure 2 depicts the deep learning sigmoid neural network clustering topology.

3.1 Deep learning sigmoid neural network clustering (DLSNNC)

A sigmoid function is a mathematical function with a distinctive "S"-shaped curve, sometimes known as a sigmoid curve. Equation (1) represents the sigmoid function,

$$f(sig) = \frac{1}{1 + E_y^i}, \tag{1}$$

where (sig) is the input and f is the output. The output of a sigmoid function is used in DLSNN normalisation. The measurement of haphazardness used to describe the texture of the input fog node data is entropy (E). The entropy of the ith data E_y^i is calculated by the condition (2)

$$E_y^i = \sum_{u=0}^{m-1} \sum_{v=0}^{m-1} P(u,v)(-\log_2(P(u,v))), \tag{2}$$

where, u and v are the coefficients of co-occurrence matrix of enhanced node, $P(u,v)$ is the component in the co-occurrence matrix at the coordinates u,vm and is the dimension of the co-occurrence matrix.

Table 1 Summary of existing techniques in Fog computing

Existing work	Quality attributes	Technique	Features	Tool used	Findings
Hussein et al. [17]	Communication cost, response time	Ant colony optimization (ACO) and particle swarm optimization (PSO)	Formal model for task offloading is provided Two nature inspired metaheuristic optimization algorithms named ACO and PSO are used based on formal model Comparison is done between both algorithms	MATLAB	Results of the experiments reveal that the proposed ACO task offloading method improves IoT application response times and successfully balances workloads among fog nodes
Gavaber et al. [18]	Bandwidth, delay	Bandwidth and delay efficient placement (BADEP), uncritical module placement (UMP)	Division of modules into two types: critical and non-critical Implementation of BADEP algorithm for critical type and UMP for non-critical type	iFogSim	Simulation results suggest that the proposed strategy reduces delay by 9 and network usage by 13 percentage
Vedaei et al. [19]	Energy consumption, bandwidth	Adaptive-network-based fuzzy inference system (ANFIS)	Real-time monitoring and notifying patient's health using ML algorithm Utilization of a fuzzy decision-making system on the fog server Training memberships and rule defining using ANFIS	Raspberry Pi Zero	The system might aid users in keeping track of their daily activities and lowering their risk of contracting the Coronavirus
Saidi et al. [20]	Latency, energy consumption	Particle swarm optimization (PSO) algorithm	Comparison of task generated in fog and cloud computing Comparison based on performance metrics, Workflow scheduling and comparison is done using PSO algorithm	FogWorkflowSim toolkit	Design of efficient fog based framework for remote monitoring system of elderly people
Li et al. [21]	Resource utlization and user satisfaction	Fuzzy c-means algorithm, particle swarm optimization (PSO)	Standardization and normalization of resource attributes, clustering of fog resources is done using fuzzy clustering with PSO Matching of classified resource with that of user request done using weighted matching method	MATLAB	The suggested approach may more quickly match user requests with relevant resource categories, resulting in higher user satisfaction
Aburukba et al. [22]	Latency	Genetic algorithm	Scheduling of IoT request using customized genetic algorithm Two important parameters: population size (POP) and the maximum number of iterations (MAX) impacts directly on the quality of solution	Discrete event simulator	Suggested technique has a 21.9 to 46.6 per cent lower total latency than other algorithms

Table 1 continued

Existing work	Quality attributes	Technique	Features	Tool used	Findings
Kishor et al. [23]	Latency	Intelligent multimedia data segregation (IMDS) scheme using machine learning	Data is divided into k-chunks using k-fold random forest technique k−1 chunks are used for training purpose and left for testing the model	Python 3.7	By reducing latency and network utilization, the suggested approach improves the quality of service in e-healthcare
Akintoye et al. [24]	Cost, latency, energy consumption	Hungarian algorithm based binding policy (HABBP), genetic algorithm based virtual machine placement (GABVMP)	Linear-programming problem of task allocation done using HABBP algorithm Load balancing is done using GABVMP algorithm	CloudSim	According to the simulation findings, the GABVMP outperformed the two greedy heuristics
Shahid et al. [25]	Energy consumption, latency	Popularity-based cache mechanism	Random distribution for the popular content Filtration method id used for caching of popular content on active node Load-balancing algorithm to increase efficiency of overall system	python	In terms of energy usage and delay, the suggested approach's evaluation shows better results
Mahmud et al. [26]	Latency	Module placement and module forwarding algorithm	Prioritization of application for allocating fog nodes is done using module placement algorithm Optimization of resources is done using module forwarding technique	iFogSim	For applications with tight deadlines, the suggested management approach overcomes latency in service delivery

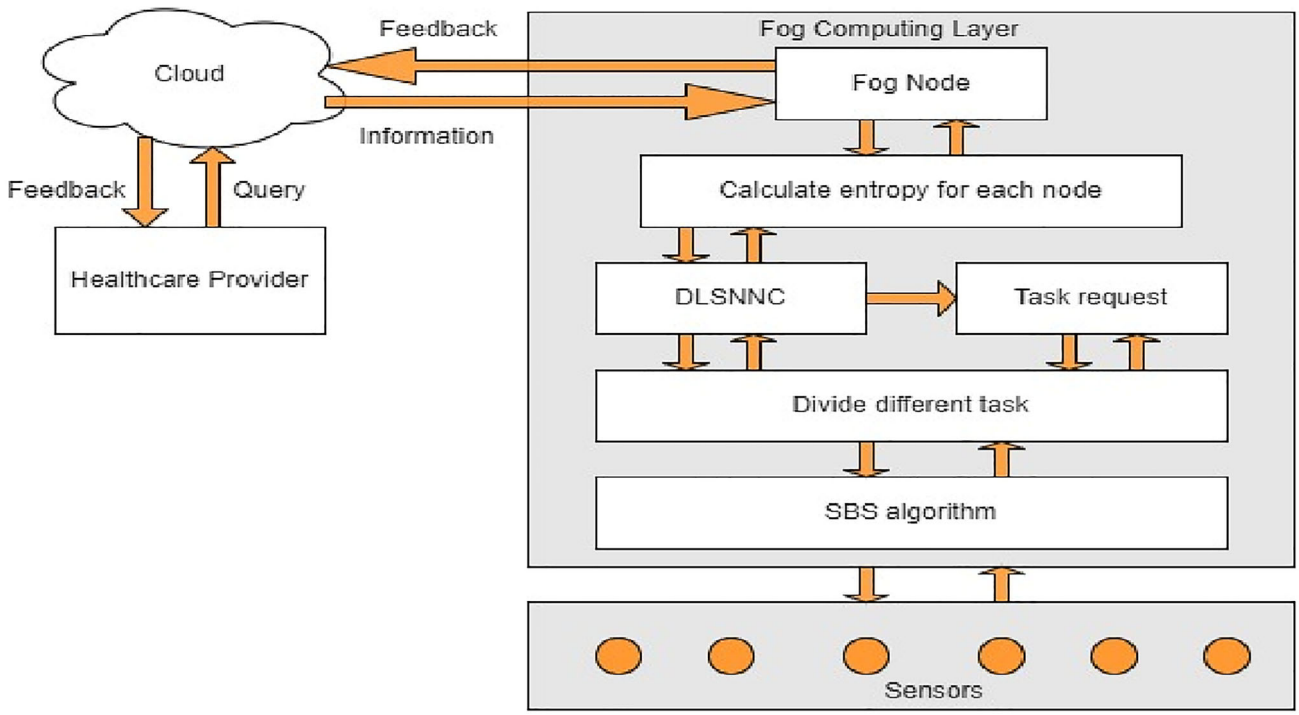

Fig. 1 Proposed methodology to improve quality of service in healthcare system

Algorithm 1 Pseudo code of the proposed DLSNNC

Input: sensor data
Output: data clustering

Begin:
Set all weights and biases using (3), (4).
For all input data I_f **do**
//Convolutional layer
For $k = 1$ to n do
For layers =1 to $L' - 1$ **do**

$$C_k = \sum_{m=0}^{M-1} y_n \hat{h}_{k-n}$$

End for
End for
//sigmoid-based normalization layer equation (6),
For $k = 1$ to n **do**
For layers =1 to $L' - 1$ **do**
// sigmoid

$$Z_{norm} = \frac{f - \mu}{\sigma}$$

End for
End for
// Weights of the previous neural network should be upgraded.
For $i = k$ to 1 **do**
For $i = 1$ to L' **do**
If the module i isn't a maximum-pooling layer,
Upgrade the module's weights and biases. i
End if
Increase the module's thresholds. i for classification
End for
End for
End

The weight and biases of the preceding layers in the structure design are used by the DLSNNC classifier to reach a conclusion. The model is then improved with conditions (3) and (4) for each layer independently.

$$\Delta W_l = -\frac{x\lambda}{r} W_n - \frac{x}{N_t} \frac{\partial C}{\partial W_n} + m\Delta W_n(t), \tag{3}$$

$$\Delta B_n = -\frac{x}{n} \frac{\partial C}{\partial B_n} + m\Delta B_n(t), \tag{4}$$

where W_n means the weight, B_n means the bias,n means the layer number, λ means the regularization parameter, x means the learning rate, N_t means the total amount of sensor data sets,m means a momentum, t means the upgrading phase, and C means the cost function.

The DLSNN Cluster contains various kinds of layers are according to the subsequent,

Step 1: Convolutional layer: Using a condition, this layer completes the convolution of the input data with the kernel (5).

$$C_k = \sum_{m=0}^{M-1} y_n \hat{h}_{k-n}, \tag{5}$$

where y_n represents reproduced segmented data, $\hat{h}$ represents the filter, and M represents the number of components in y and the output vector is C_k.

Step 2: Sigmoid-based normalization layer: The technique of linearly modifying data to fit it within a given range is known as normalisation. The Z-score normalisation method is used to standardise data by changing it linearly. The formula for Z-score normalisation is shown in Eq. (6):

$$Z_{norm} = \frac{f - \mu}{\sigma}. \tag{6}$$

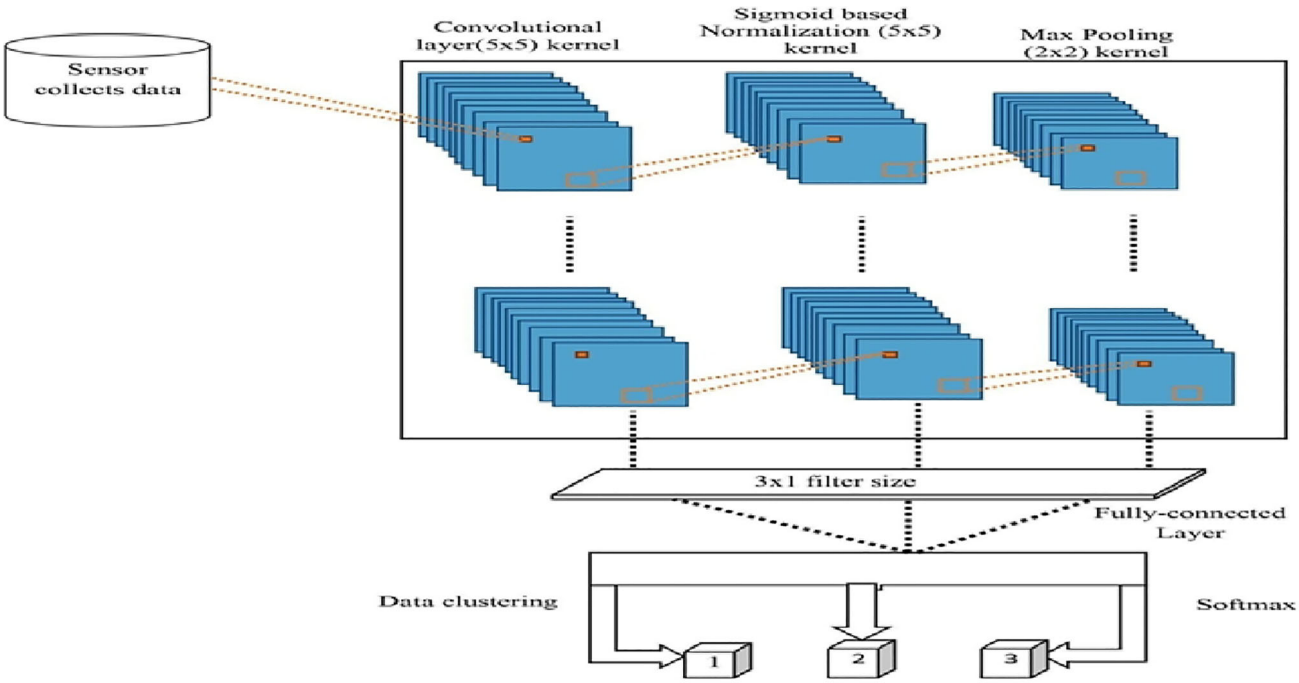

Fig. 2 Architecture of the DLSNN clustering

Here, Z_{norm} is the normalized output, f is the sigmoid function value, μ is the mean value of the convolutional layer output data, and σ is the standard deviation of the values in the convolutional layer output. The convolutional layer output is normalized using the sigmoid function by using the Eq. (6). The Sigmoid-based normalised output from this layer is sent into the pooling layer. This layer is expected to contribute to the pooling layer by providing value-based normalised data support.

Step 3: Pooling layer: The down-sampling layer is another name for this layer. To save computing effort and minimise overfitting, the pooling stage reduces the size of output neurons from the convolution layer. The max-pooling algorithm selects only the highest value in each data map, resulting in fewer output neurons. Pooling layers are typically used after convolution layers to assist simplify the information in the convolution layer's output.

Step 4: A fully connected layer: The actuation work computes a probability distribution of the classes. Thus, the output layer uses the softmax function to find a preceding layer outcome that fits the most clustered data.

$$p_i = \frac{e^{y_i}}{\sum_1^k e^{y_i}}, \tag{7}$$

where y, which represents the resultant cluster. Here, the DLSNNC is adapted with the sigmoid function-based normalization to direct the over-fitting in layers and conclusions in the important clustering of sensor data to the fog-cloud computing layers.

3.2 Score based scheduling algorithm

Our major purpose is to design workflow tasks that contain patients' health-care data. Initially, the task request is produced and separated into numerous task requests so that execution durations may be reduced at a reasonable cost while staying within the user-specified deadline. The Score based workflow task scheduling algorithm selects only those task requests that match the minimal threshold of workflow tasks for scheduling. In [27], there is an existing scheduling algorithm. The flow chart in Fig. 3 describes our proposed Score based workflow task scheduling system.

The steps of the Score based Scheduling algorithm are described below:

The following are the steps of the score-based scheduling algorithm:

Step 1: Submit the workflow task list, which includes patient healthcare information. (T = T1, T2, T3, ….,T_n).

Step 2: Contact the data centre to learn about the virtual resources that are available. VM = VM1, VM2, VM3, and so on, up to VMn.

Step 3: Assign a user-defined deadline constraint D in the form of sub-deadlines for various task requests to the whole workflow application.

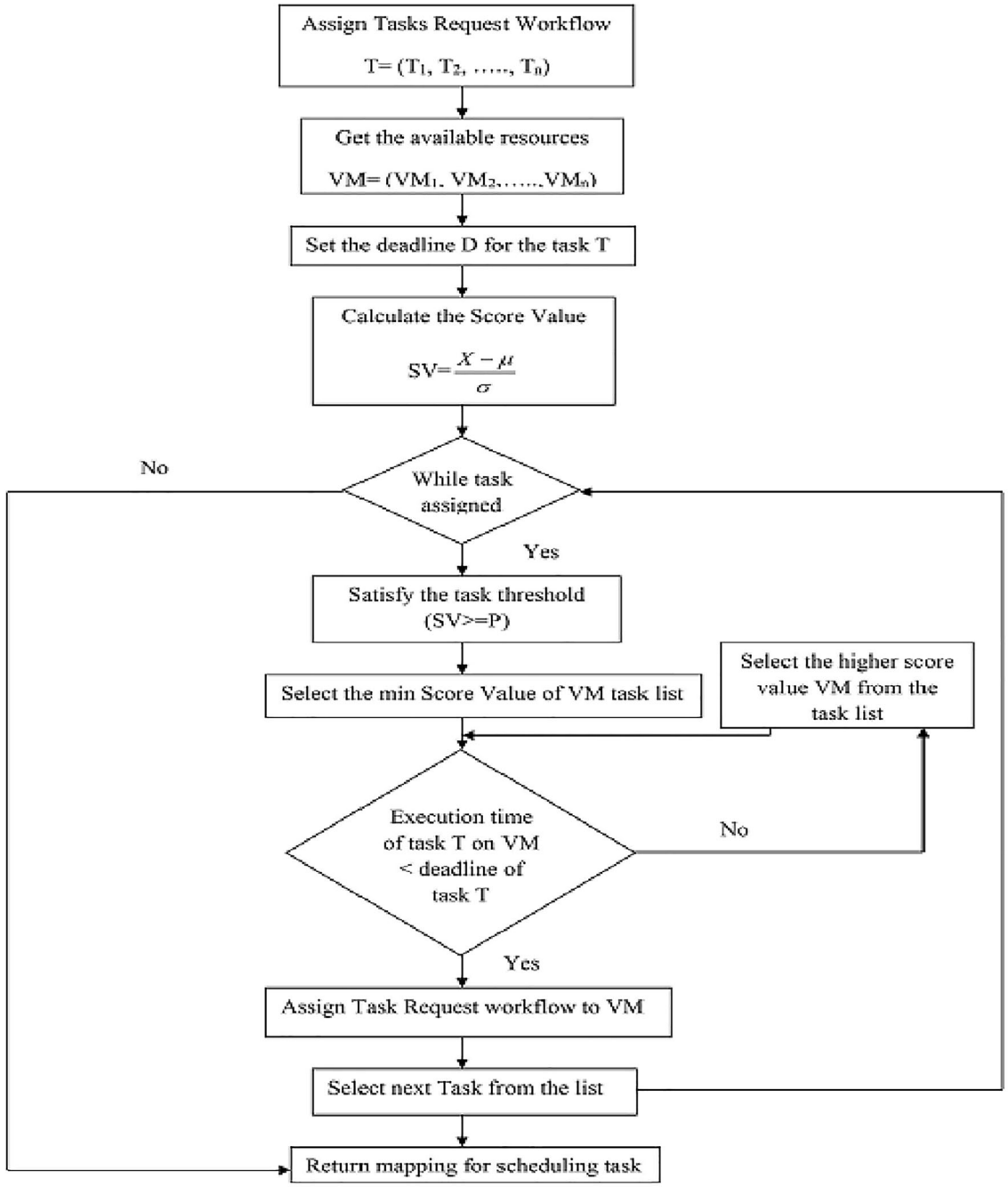

Fig. 3 Flowchart of the proposed SBS algorithm

Table 2 Latency comparison of cloud and fog and cloud

Data	Cloud	Fog and cloud
500	250	110
1000	581	138
1500	1280	134
2000	1678	295
2500	2210	482
3000	2791	479

Step 4: Using the components' minimum sub-scores, determine the VMs' scores value (SV) where X-defines the observed value, μ- mean of the sample task, σ-standard deviation task.

Step 5: Repeat steps 6, 7, and 8 if the task list contains the tasks to schedule; otherwise, return to the task mapping.

Step 6: Select the lowest-scoring VM from the VM list that meets the task's threshold. The task threshold (p) is determined by the length of the instructions.

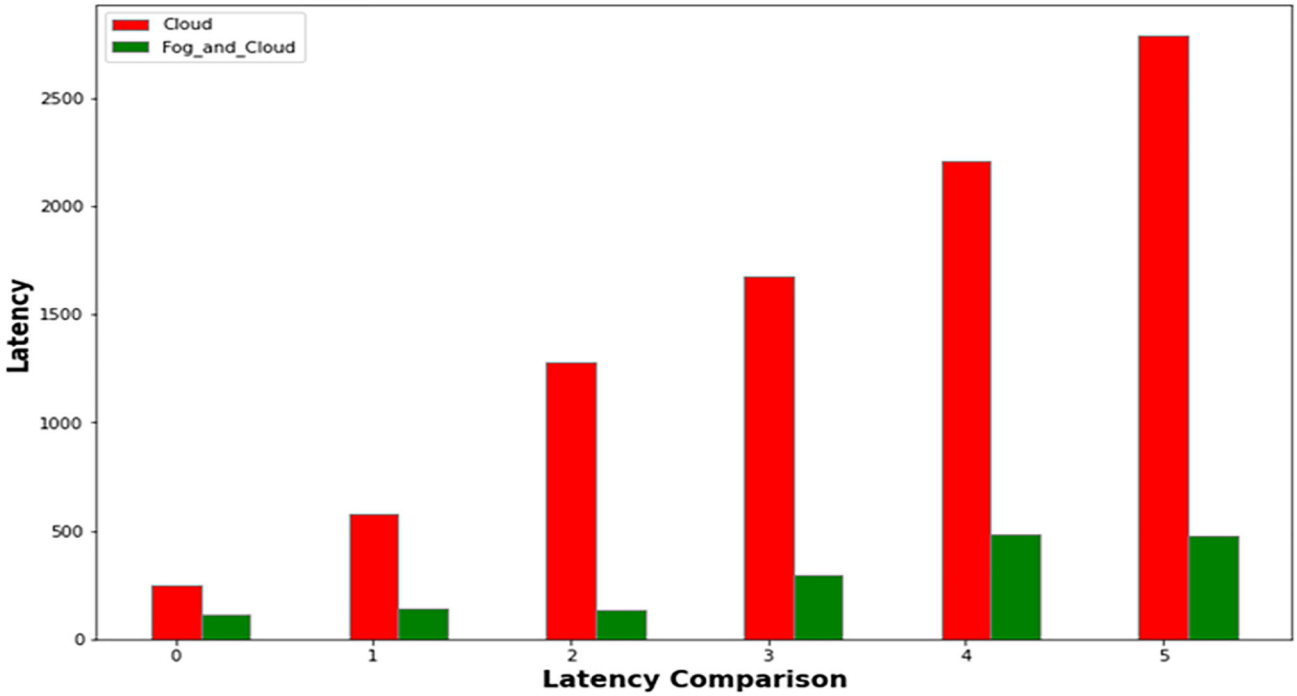

Fig. 4 Latency of fog compared to Fog + cloud system

Step 7: The job is assigned to the selected VM if it can finish the work within the specified deadline; else, the assignment is sent to the next lowest-scoring VM from the list of resources.

Step 8: From the list, choose the next assignment. If all jobs have been scheduled, their mapping to VMs will be completed.

4 Result and discussion

The implementation of our proposed effective DLSNN Clustering and Score-based scheduling for cloud IoT applications is done in PYTHON and by using an online cloud healthcare dataset. Different execution estimations such as latency and network are estimated to explore the performance of the proposed work. Finally, the average delay is estimated and compared using existing FCFS [28], SJF [28], and BMO [29] to prove the relevance of the proposed approach.

Table 3 Network usage of cloud and fog and cloud

Data	Cloud	Fog and cloud
500	284	180
1000	377	290
1500	499	321
2000	537	438
2500	578	488
3000	687	521

4.1 Latency

There will be data flow between the various tiers in our fog computing solution in health informatics. In many circumstances, the amount of information and thus the amount of time required will differ. As a result, the latency varies. As shown in equation, latency is the difference between the time of commencement and the time of completion of service (8),

$$L = ST + PT + TQT + IT. \tag{8}$$

Here, L denotes latency, ST denotes the requested task's start time, PT denotes the requested task's processing time, TQT denotes the transmission and queuing time prior to the requested task, and IT is the desired job's initiation time. In the data ranges 500, 1000, 1500, 2000, 2500, and 3000, the latency comparison between cloud and mixed cloud and fog computing layers is shown in Table 2. In addition, Fig. 4 depicts a latency comparison graph in the cloud, as well as both cloud and fog computing layers.

4.2 Network usage

The second evaluation constraint is network usage (N_{usage}). As the number of devices on the network grows, so does network usage, resulting in network congestion. As a result

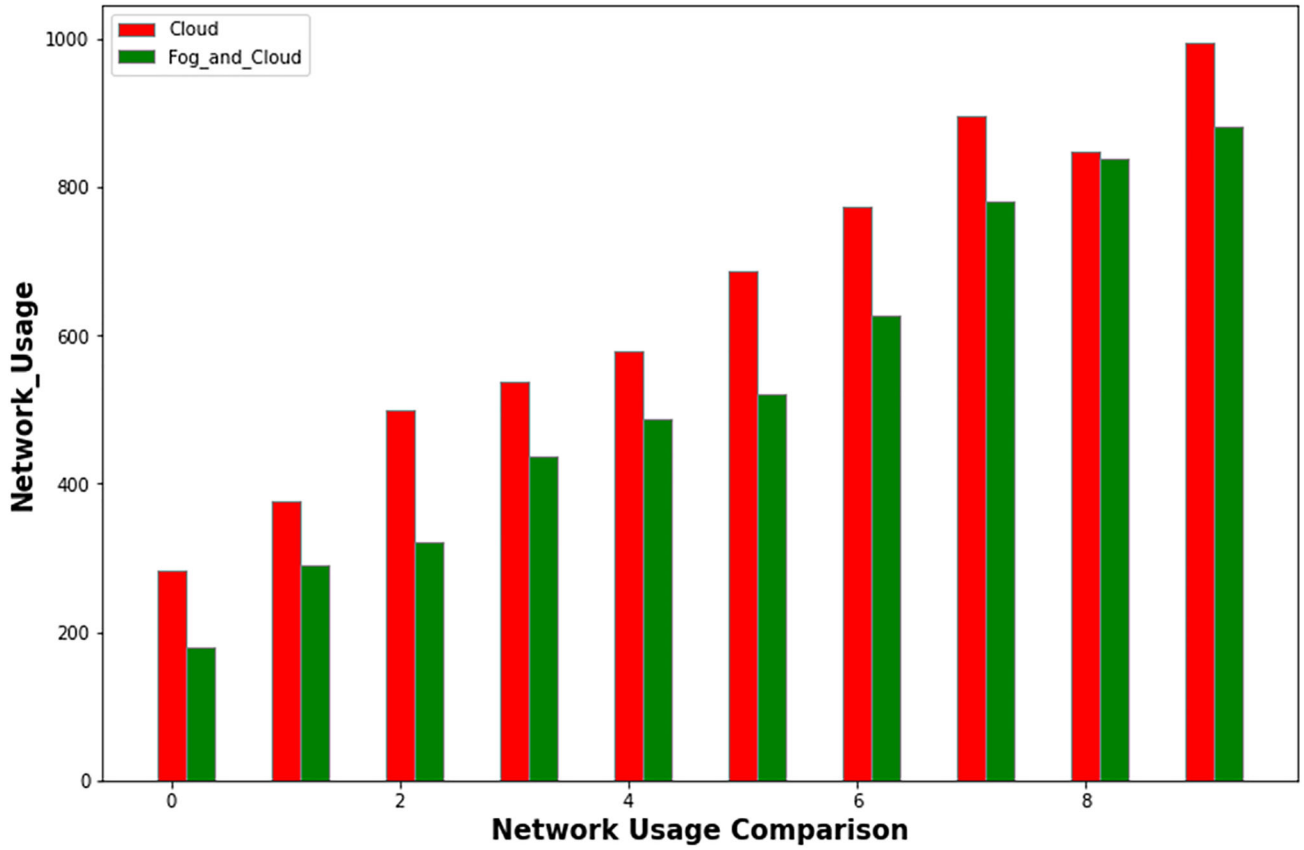

Fig. 5 Network utilization in fog compared to fog + cloud system

Table 4 Performance measure of average delay

Average waiting time in (ms)	FCFS	SJF	BMO	Proposed
500	176	129	221	110
1000	254	192	321	143
1500	296	218	378	154
2000	429	278	398	187
2500	578	398	467	231
3000	595	486	521	243

of the congestion, the application running on the Cloud network performs poorly. By dispersing the load across intermediary fog devices, fog computing aids in the reduction of network congestion. Equation is used to calculate network utilization (9),

$$N_{usage} = \sum_{i=1}^{N} L_i \times S_i, \tag{9}$$

where N is the total number of tasks, L_i is the latency, and S_i is the network size of ith task. Table 3 states the Network usage in the cloud and both cloud and fog computing layers along with the network usage in GB mentioned and the data usage 500–3000. Figure 5 stated the network usage comparison graph in the cloud and both cloud and fog computing layers.

4.3 Average delay

The ratio of average delays state the difference between starting execution time ST and the ending execution time ET for the request tasks noted in Eq. (10),

$$Average_{Delay} = ST - ET. \tag{10}$$

Table 4 states the Average Delay in FCFS, SJF, and BMO comparing with the proposed technique with data usage 500–3000. Figure 6 stated the average delay when the average waiting time increases the average delay also increases comparison graph in FCFS, SJF, BMO and comparing with the proposed technique decreases the average delay.

5 Conclusion and future work

We propose a fog-cloud computing technique for health monitoring systems in this paper. The purpose of the study reported in this paper is to improve service quality. In this

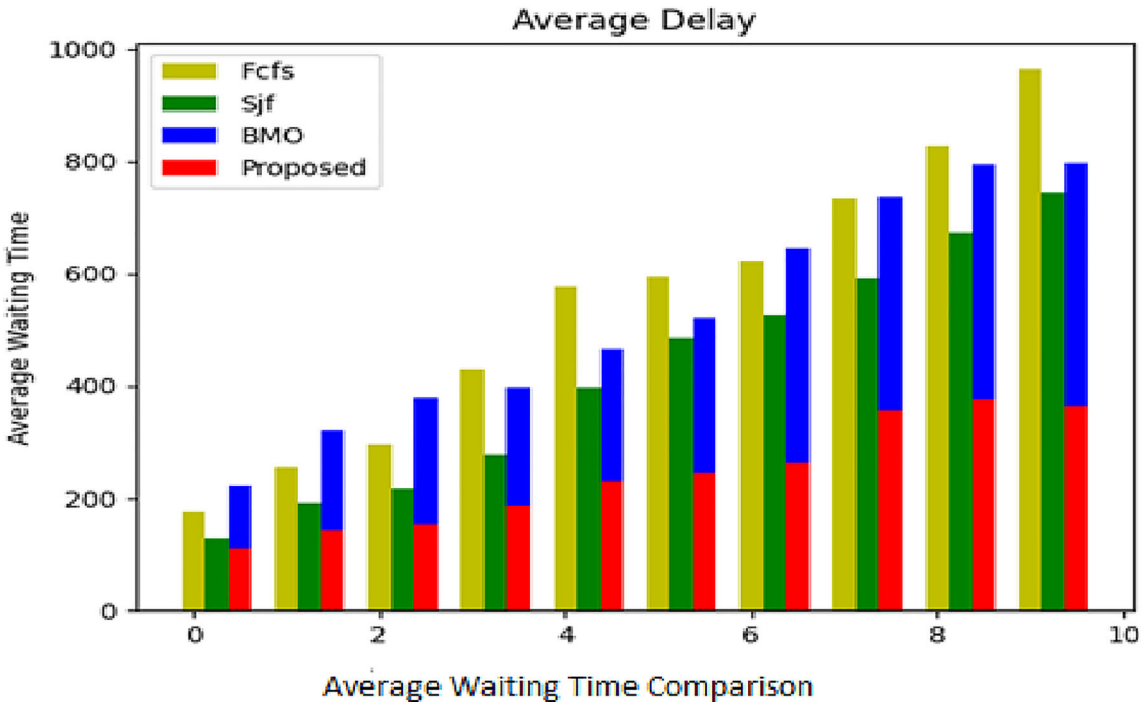

Fig. 6 Average delay of proposed approach compared to existing techniques

work, DLSNN Clustering and Score-based Scheduling are used to improve prediction. According to simulation results, proposed solution improve Quality-of-Service in the cloud/fog computing environment in terms latency and network consumption. Additionally, the proposed technique outperforms existing approach in terms of average delay. Different encryption techniques can be incorporated with the implementation of proposed architecture to improve the security of the system.

References

1. Satija U, Ramkumar B, Sabarimalai M (2017) Real-time signal quality-aware ECG telemetry system for IoT-based healthcare monitoring. IEEE Internet Things J 4:815–823. https://doi.org/10.1109/JIOT.2017.2670022
2. Costa CA, Pasluosta CF, Eskofier B, Silva DB, Righi RR (2018) Internet of Health Things: Toward intelligent vital signs monitoring in hospital wards. Artif Intell Med 89:61–69. https://doi.org/10.1016/j.artmed.2018.05.005
3. Mansourkiaie F, Ismail LS, Elfouly TM, Ahmed MH (2017) Maximizing lifetime in wireless sensor network for structural health monitoring with and without energy harvesting. IEEE Access 5:2383–2395. https://doi.org/10.1109/ACCESS.2017.2669020
4. Loubet G, Takacs A, Dragomirescu D (2019) Implementation of a battery-free wireless sensor for cyber-physical systems dedicated to structural health monitoring applications. IEEE Access 7:24679–24690. https://doi.org/10.1109/ACCESS.2019.2900161
5. Farahani B, Firouzi F, Chang V, Badaroglu M, Constant N, Mankodiya K (2018) Towards fog-driven IoTeHealth: promises and challenges of IoT in medicine and healthcare. Futur Gener Comput Syst 78:659–676. https://doi.org/10.1016/j.future.2017.04.036
6. Azimi I, Anzanpour A, Rahmani AM, Liljeberg P, Salakoski T (2016) Medical warning system based on Internet of Things using fog computing. In: International workshop on big data and information security (IWBIS), pp 19–24. https://doi.org/10.1109/IWBIS.2016.7872884
7. Hassen HB, Dghais W, Hamdi B (2019) An E-health system for monitoring elderly health based on Internet of Things and Fog computing. Health Inf Sci Syst 7(1):1–9. https://doi.org/10.1007/s13755-019-0087-z
8. Arora U, Singh N (2021) IoT application modules placement in heterogeneous fog–cloud infrastructure. Int J Inf Technol 13:1975–1982. https://doi.org/10.1007/s41870-021-00672-4
9. Sen AAA, Yamin M (2021) Advantages of using fog in IoT applications. Int J Inf Technol 13:829–837. https://doi.org/10.1007/s41870-020-00514-9
10. Isa IS, El-Gorashi TE, Musa MO, Elmirghani JM (2020) Energy efficient fog-based healthcare monitoring infrastructure. IEEE Access 8:197828–197852. https://doi.org/10.1109/ACCESS.2020.3033555
11. Kraemer FA, Braten AE, Tamkittikhun N, Palma D (2017) Fog computing in healthcare—a review and discussion. IEEE Access 5:9206–9222. https://doi.org/10.1109/ACCESS.2017.2704100
12. Verma P, Sood SK (2018) Fog assisted-IoT enabled patient health monitoring in smart homes. IEEE Internet Things J 5(3):1789–1796. https://doi.org/10.1109/JIOT.2018.2803201
13. Rahmani AM, Gia TN, Negash B, Anzanpour A, Azimi I, Jiang M, Liljeberg P (2018) Exploiting smart e-Health gateways at the edge of healthcare internet-of-things: a fog computing approach. Futur Gener Comput Syst 78:641–658. https://doi.org/10.1016/j.future.2017.02.014
14. Andriopoulou F, Dagiuklas T, Orphanoudakis T (2017) Integrating IoT and fog computing for healthcare service delivery. In:

Components and services for IoT platforms, Springer, Cham, pp 213–232. https://doi.org/10.1007/978-3-319-42304-3_11
15. Mutlag AA, Ghani MK, Kumar A, Mohammed MA, Mohd O (2019) Enabling technologies for fog computing in healthcare IoT systems. Future Gener Comput Syst 90:62–78. https://doi.org/10.1016/j.future.2018.07.049
16. Moghadas E, Rezazadeh J, Farahbakhsh R (2020) An IoT patient monitoring based on fog computing and data mining: cardiac arrhythmia use case. Internet Things 11(24):100251. https://doi.org/10.1016/j.iot.2020.100251
17. Hussein MK, Mousa MH (2020) Efficient task offloading for IoT-based applications in fog computing using ant colony optimization. IEEE Access 8:37191–37201. https://doi.org/10.1109/ACCESS.2020.2975741
18. Gavaber D, Rajabzadeh A (2021) BADEP: bandwidth and delay efficient application placement in fog-based IoT systems. Trans Emerg Telecommun Technol 32(8):e4136. https://doi.org/10.1002/ett.4136
19. Vedaei SS et al (2020) COVID-SAFE: an IoT-based system for automated health monitoring and surveillance in post-pandemic life. IEEE Access 8:188538–188551. https://doi.org/10.1109/ACCESS.2020.3030194
20. Saidi H, Labraoui N, Ari AA, Bouida D (2020) Remote health monitoring system of elderly based on Fog to Cloud (F2C) computing. In: 2020 international conference on intelligent systems and computer vision (ISCV), pp 1–7. https://doi.org/10.1109/ISCV49265.2020.9204096
21. Li G et al (2019) Methods of resource scheduling based on optimized fuzzy clustering in fog computing. Sensors (Basel, Switzerland) 19(9):2122. https://doi.org/10.3390/s19092122
22. Aburukba RO, Alikarrar M, Landolsi T, El-Fakih K (2020) Scheduling internet of things requests to minimize latency in hybrid fog-cloud computing. Future Gener Comput Syst 111:539–551. https://doi.org/10.1016/j.future.2019.09.039
23. Kishor A, Chakraborty C, Jeberson W (2020) A novel fog computing approach for minimization of latency in healthcare using machine learning. Int J Interact Multimed Artif Intell. https://doi.org/10.9781/ijimai.2020.12.004
24. Akintoye SB, Bagula A (2019) Improving quality-of-service in cloud/fog computing through efficient resource allocation. Sensors 19(6):1267. https://doi.org/10.3390/s19061267
25. Mahmud R, Ramamohanarao K, Buyya R (2019) Latency-aware application module management for fog computing environments. ACM Trans Internet Technol 19(1):1–21. https://doi.org/10.1145/3186592
26. Shahid MH, Hameed AR, Islam S, Khattak HA, Din IU, Rodrigues J (2020) Energy and delay efficient fog computing using caching mechanism. Comput Commun 154:534–541. https://doi.org/10.1016/j.comcom.2020.03.001
27. Singh R, Singh S (2013) Score based deadline constrained workflow scheduling algorithm for cloud systems. Int J Cloud Comput Serv Archit (IJCCSA) 3(6):31–41. https://doi.org/10.5121/ijccsa.2013.3603
28. Jamil B, Shojafar M, Ahmed I, Ullah A, Munir K, Ijaz H (2020) A job scheduling algorithm for delay and performance optimization in fog computing. Concurr Comput Pract Exp 32(7):e5581. https://doi.org/10.1002/cpe.5581
29. Dou W, Tang W, Liu B, Xu X, Ni Q (2020) Blockchain-based mobility-aware offloading mechanism for fog computing services. Comput Commun 164:261–273. https://doi.org/10.1016/j.comcom.2020.10.007