Received: September 20, 2023. Revised: October 25, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Hybrid Fog-Edge-IoT Architecture for Real-time Data Monitoring

Ramya R[1] **Ramamoorthy S[1]***

[1]*Department of Computing Technologies, SRM Institute of Science and Technology, Kattankulathur, Tamilnadu, India*
* Corresponding author's Email: ramamoos@srmist.edu.in

Abstract: The Internet of Things (IoT) has recently transformed many lives into comfort zones. The IoT concepts and their exponential growth are burning research issues that need more space for processing and monitoring. To meet the explosive growth of IoT data, edge and fog computing is being deployed for better data analysis with minimum computational complexity. The data can be gathered and analyzed at the fog or edge layer to maximize data utilization. This paper presents a novel prediction and resource allocation using Q-based deep extreme neural network learning algorithms suitable for smart healthcare applications. Resource allocation and effective prediction represent challenging missions involving various resources and IoT nodes to achieve effective computations, leading to real-time computational complexity. The proposed system is composed of four modules: (1) the data collection unit (DCU), (2) the data processing unit (DPU), (3) the intelligent prediction module (IPM), and (4) the adaptive resource allocation module (ARAM). The proposed algorithm is trained using the medical training data, which consists of different heart arrhythmias, specifically heart attacks. The proposed algorithm will be tested using the user's sensing data from the IoT layer to predict the probability of a heart attack and then decide accordingly. The system aims to achieve an improved quality of service (QoS) with less latency. Extensive experimentation results demonstrate that the proposed algorithm has outperformed real-time data monitoring with prediction and resource allocation.

Keywords: Internet of things; fog; edge layers; Q-based deep extreme learning machines; quality of services.

1. Introduction

The IoT is made up of devices such as sensors, transducers, micron rollers, and transceivers that are able to communicate with one another in order to achieve that same task [1]. The IoT is a relatively recent invention in technology that offers a unified platform for the integration of a wide range of technologies in order to enable intelligent performance across all domains of activity. As a direct result of this, there has recently been a rise in the number of homes, mobile phones, and other embedded applications that are connected to the internet. Clouds are considered the standard infrastructure to gather data from IoT devices normally used for monitoring and analyzing [2]. This remote location has a negative impact on the response time, particularly for apps dealing with real-time healthcare information. In addition, IoT sources may be spread out over a wide geographic area and create a significant quantity of data that is then transferred to the cloud for processing, which can result in the cloud being overloaded. The computing resources located at the edge of an IoT system are able to address the issues that were discussed before [3, 4].

Fog computing, also known as edge computing, is being implemented as a solution to this problem in the hopes of successfully implementing services at the network's edge. The FC is able to accomplish location awareness and significantly cuts down on latency [5]. The gadgets that create fog are known as fog nodes (FN). These fog nodes enable a widespread dispersion of services that can analyze data from Internet of Things devices close to their points of origin. To satisfy the QoS and resource needs of wide-standard IoT systems [6], fogs and clouds often cooperate with one another in an integrated way, as seen in Fig. 1. In FC, Resource Allocation, also known as RA, is a challenging job since it requires a number of different resources and fog nodes to do the calculations that are necessary [7].

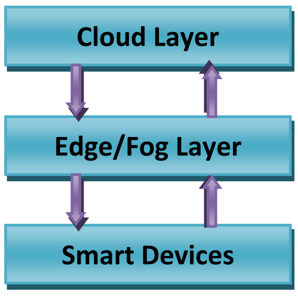

Figure. 1 General architecture for the fog-edge –IoT architecture

In FC, resource allocation (RA), is a challenging task since it requires a number of different resources and fog nodes in order to complete the calculations that are necessary for IoT systems. In the past, relatively a few additional RA techniques, such as least connection (LC), round robin (RR), weighted round robin (WRR), and adaptive weighted round robin (AWRR). On the other hand, they have a number of drawbacks, the most notable of which are as follows: (i)they do not take into account the heterogeneity of the computing resources; and (ii) their poor performance is attributable to the migration of processes. Both of these drawbacks are highlighted in the following sentence: (iii) A much prolonged delay (iv) the lack of one universally accepted standard for FC [8] It is anticipated that the combination of fog, cloud, and a diverse collection of IoT devices would result in an increase complexity of RA concerns [9].

Recently, the advent of machine learning and deep learning algorithms has provided better insights to achieve better QoS between fog devices and cloud layers. Many algorithms exist, such as deep neural networks (DNN) [10], deep reinforcement networks (DRN) [11], probabilistic neural networks (PNN) [12], and deep belief networks (DBN) [13]. But these algorithms are designed to achieve high QoS performance but are not provided with computational complexity. The computational complexity of these algorithms will increase as the data grows that again affecting the performance of Fog devices in predicting the particular disease with high latency allocation of resources.

Motivated by this above drawback, this paper proposes the Latency Aware Intelligent prediction and adaptive resource allocation method for an effective diagnosis of diseases and high-speed allocation. The form employs the novel Q-deep extreme learning machines (Q-DEM) to achieve a high prediction ratio with less time for computation. To the best of our knowledge, the proposed ensemble of Q-learning and deep-learning machines are the first to be deployed for fog devices that control IoT nodes and clouds.

Contribution of the paper: The contribution of the paper is tri-folded which are as follows. The paper proposes the ensembled combination of Q-reinforcement learning and deep ELM to achieve high performance and less latency to handle the large IoT and fog nodes. This paper proposes a new dataset collection unit (DCU) to generate larger IoT nodes that aid innovative healthcare applications. The generated datasets aim to predict heart diseases, specifically heart attacks. Finally, the paper proposes extensive evaluation methods in which various metrics such as latency, prediction accuracy, and mask pain analysis are calculated and compared with the other state-of-art algorithms.

The remaining portions of the paper are structured as follows: section 2 presents a proposed methodology for the fog environment using real-time resource allocation and QDELM, with more details about each contribution. Additionally, the data collection unit and pre-processing data unit are detailed in section 2. Section 3 introduces the evaluation results and discussion. The conclusion with the future scope is discussed in Section 4.

Pareek et al. [14] studied various computing systems, such as cloud, edge, and fog systems targeted for the healthcare industry. The significance of fog computing is reviewed with regard to its architecture, performance, and efficiency in healthcare monitoring, remote assistance, and tracking purposes over cloud computing. Fog architecture is positioned with a minimum number of nodes in between the cloud server and end-user devices, which extremely reduces the overall delay and response time and improves the transmission speed during data transfer with more protection. The fog nodes reduce bandwidth, latency, and power consumption, which is essential in the healthcare industry. The author concluded that the fog layer with minimal nodes suites better for healthcare functions compared to cloud and edge computing data centers in terms of processing, fast transmission, and security perspectives.

Bhatia et al. [15] employed IoT technology to evaluate a person's real medical tracking during a warm-up exercise and evaluate them for illness intensity assessment. In this research, the Bayesian belief networks (BBN) technique is utilized to calculate the severity of illness as a probabilistic metric. Gia et al. [16] proposed a fog layer and resource edge devices for cost-effective patient monitoring. This strategy cuts healthcare costs while also improving the quality of life. The nRF protocol serves as the foundation for these energy-efficient sensor nodes. The system makes quick decisions and provides services that are needed right now.

Elhadad et al. [17] developed a five-layer smart monitoring system with different fog layers to maintain the tracking efficiency of patients in real time. The proposed framework includes the "cloud, network, and edge computing layers and a sensor layer" in each Fog system. The sensor layer consists of many wearable devices to observe the temperature, systolic and diastolic pressure values, and heart rate from the sensors connected to patients, which have been transmitted to the fog layer in parallel processing. The fog layer performs the data transmission in a distributed format with security, and the analyzed data is stored on a cloud server with notifications. The proposed 5-layer healthcare monitor achieved better results in throughput and overall response time.

Kumari et al. [18] reviewed the real-time challenges of healthcare monitoring, tracking, and the difficulties of huge data transmission. The significance of fog computing in medical care 4.0 is detailed. The author detailed the healthcare services and challenges regarding delay, response time, data management, security and privacy, scalability, human interfaces, and interoperability. To address these issues, researchers are adopting the fog computing layers in monitoring, tracking, and improving remote assistance, especially for disabled patients in healthcare environment 4.0. The taxonomy of fog system combines the data collection, analysis, and network configurations (protocols and routers) which is finally communicated to end-users in a distributed format that enables the efficiency of remote monitoring and extends the lifetime of patients.

The outcome of these papers clearly states that fog-computing systems achieve the best results in providing healthcare services compared to a centralized cloud architecture from many perspectives, such as "security, power consumption, data transmission efficiency, reduced delay, latency, and high confidentiality," and so on. Though there are many benefits in adopting the fog layers in the healthcare industry, there are still a few challenges called resource management, effective protocol selection, routers design, reducing the number of gateways, and real-time decision-making are need to be considered. The following section illustrates the various resource management approaches for fog computing systems.

HeenaWadhwa et al. [19] created a new resource assignment model called TRAM, which includes different scheduling heuristics to improve the utilization of fog resources. Using the expectation maximization (EM) technique, this strategy is utilized to track the intensity level of existing workloads and assess the current resource status. A wireless system is used to handle all of the available resources. This work presents a resource grading scheduling technique for a fog computing environment. The performance of this technique was evaluated using the iFogSim simulator, and the results were compared to those of "SJF, FCFS, and MPSO." According to the simulated findings, TRAM analyzed the completion time, network and energy usage, as well as the average loop delay. The proposed TRAM approach is not effective for large structures and in-efficient for massive data transmission due to the sequential processing method.

Basset et al. [20] suggested an energy-aware work scheduling method for marine predators. Using IoT devices, this technique was deployed in fog applications. In the pervasive computing environment, this solution tackled the issue of bandwidth utilization overhead. By adopting a modified marine predator's method, the hurdles involved in fog computing work scheduling were removed. The ranking approach determined the consecutive iterations required to achieve a better position than the swarm-based optimizers. The limitation of this proposed model is high computational complexity due to finding global solutions with a high number of search iterations. An optimization problem was devised by Shudong et al. [21] with the goals of increasing resource utilization and maintaining load equilibrium on edge nodes in 5G networks. They came up with the BSOM service offloading method so that they could make available computer resources on the network. They defend the privacy of individuals' sensitive data, and the privacy protection model that they use was developed by monitoring a variety of hostile relationships.

Under the framework of the fog computing methodology, Wang et al. [22] proposed a task scheduling algorithm that was derived from a modified version of the firework algorithm. The roles of the devices that make up the Internet of Things were broken down into three categories: requests that are sensitive to the duration of time, storage needs, and bandwidth requirements. Fog devices and fog processors were thought to make up the fog layer. The tasks were grouped according to the classification kinds, and the fog's resources were also categorized. This method employed an updated fireworks algorithm to schedule tasks in the most efficient way possible. This model has been tested, and the findings show that scheduling using I-FASC leads to better-balanced execution time and device resource allocation.

Received: September 20, 2023. Revised: October 25, 2023.

Talaat et al. [23] created a novel prediction-based resource allocation technique for fog computing systems. EPRAM uses a real-time resource allocation and prediction method to accomplish effective resource management in a fog environment. The PNN and one or more predictors are used in conjunction with the EPM so that an accurate forecast of a target field may be made by the EPM. PNN is trained to detect the probability of having a heart attack by using the dataset that it was trained on as its main source of information. PNN will then be analyzed to forecast the potential of a heart attack using the user's sensory data acquired from the IoT layer, and in response to this prediction, the most proper measures will be done. EPRAM, unlike existing RA approaches, uses a novel deep reinforcement learning (RL) algorithm. The prediction algorithm is likewise based on the probabilistic neural network (PNN). Because deep RL and PNN were used, it was able to reach such acceptable results.

2. Proposed method

2.1 System overview

The most important application that is directly connected to the IoT is the creation of an effective healthcare system. When it comes to healthcare systems, there are numerous considerations that need to be made, such as the passage of time, the protection of personal information, and the precision of diagnoses. It is important to take into consideration the dependability and credibility of the various healthcare systems. The system model that is presented in this research and is shown in Fig. 2 is derived from the parameters that were discussed previously. The suggested system is made up of three distinct levels, which include (1) Internet of Things layers, (2) fog-edge layers, and (3) cloud layers. In the first layer, IoT devices are equipped with healthcare sensors such as pulse sensors, ECG sensors, and positional sensors. The data from IoT layers is collected and preprocessed in fog-edge layers. The fog-edge layer is also considered for handling the request from the IoT devices and forwarding it to the suitable cloud layers. The cloud layers analyze the data and send it to the suitable healthcare systems.

2.2 IoT layers

This section introduces the IoT layers for collecting healthcare data from medical sensors. To collect the experimental data, the IFogSim simulator is used to deploy the nodes in the IFogSim simulator, as shown in Fig. 3. Additionally, the proposed research employs the mHealth datasets for training, testing, and validation. Using the iFogSim simulator, 50 nodes are created in which the 15000 medical data with five attributes are collected for 45 days. Each IoT node has a pulse sensor, an ECG sensor, and a motion detector. The sample datasets collected from the IFogSim are presented in Table 1.

In the meantime, research uses the mHealth (Mobile Health) dataset, which comprises body motion and vital heart sign recordings for ten volunteers of diverse profiles while they were

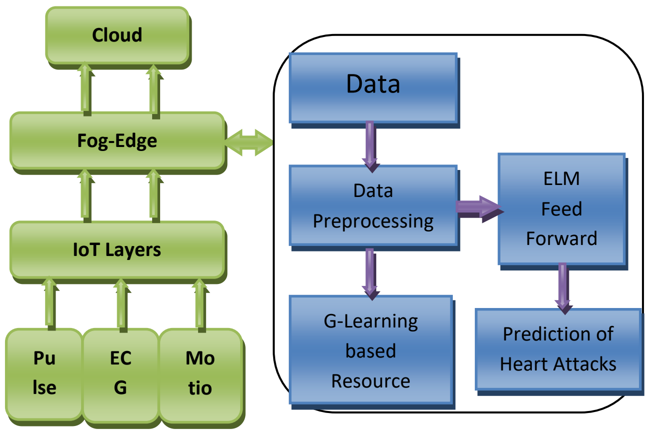

Figure. 2 Overall architecture for the proposed resource allocation and effective prediction

Table 1. Sample datasets collected from the iFogSim environment

Patient Data	Pulse rate	ECG_Data	Acceleration Data (Motion detection)	Location ID
Patient _1	72	88	167	36.56, 144.29
Patient _2	65	92	145	36.78,135.89
Patient _3	76	88	167	35.90,125.90
Patient _4	74	102	182	35.80,115.90
Patient _5	68	90	190	36.90.129.90
Patient _6	70	104	120	37.89,113,40

Table 2. Sample mHealth dataset used for training, testing and validation

Patient Data	Pulse rate	ECG_Data	Acceleration Data(Motion detection)	Location ID
Patient_1	72	88	167	36.56,144.29
Patient_2	65	92	145	36.78,135.89
Patient_3	76	88	167	35.90,125.90
Patient_4	74	102	182	35.80,115.90
Patient_5	68	90	190	36.90.129.90
Patient_6	70	104	120	37.89,113,40

engaging in physical activities. This was done in order to demonstrate that the proposed algorithm is superior to existing alternatives. The sensors that are placed on the chest are based on readings from a two-lead electrocardiogram, which may be utilized for monitoring the patient's heart rate as well as testing for various of arrhythmias. These datasets were collected from three sensor nodes and ten volunteers. Table 2 presents the sample datasets used for validating the proposed system.

2.3. Fog-edge layers

2.3.1. Data Collection and Pre-Processing Unit:

The information that is gathered from Internet of Things devices is saved in fog edge nodes, which are also the locations where the data is preprocessed for the purposes of resource allocation and the forecasting of heart attacks. Nearly 15000 data are collected from the IFogSim simulators, and data are preprocessed for further applications. The data preprocessing is carried out in three stages: the sampling stage, the splitting stage, and the balancing stage. During the sampling stage (SS), data are selected for sampling based on the location and nature of the data. The data can be classified as critical and non-critical data, as shown in Table 3. The critical data demands a higher response time which needs the feedback and control system to save

Table 3. Data splitting process deployed for training and testing

Patient Data	Pulse rate	ECG_Data	Accelaration Data (Motion detection)	Location ID	Labeled Data
Patient_1	72	88	167	36.56,144.29	0 (Not critical)
Patient_2	101	120	145	36.78,135.89	1 (critical)
Patient_3	76	88	167	35.90,125.90	0
Patient_4	102	112	182	35.80,115.90	1
Patient_5	68	90	190	36.90.129.90	0
Patient_6	105	194	120	37.89,113,40	1

the life of the patients. After classifying the data, collected data are split into three categories such as training data, testing, and validating data. However, the testing data can refine the model's performance, which the model can rebuild using a training sample regarding the testing performance. The validation data, which, unlike the training and testing sets, did not contribute in any way to the development of the final model, are used to evaluate the performance of the model in comparison to data that it has not before seen. This is in comparison to the testing sets, which did contribute in some way to the development of the final model. Table 3 presents the splitting of the data structure.

In the third stage, classified and labeled data are balanced by discarding (reducing) highly frequent records in the data, which boosts the performance of the trained model.

2.3.2. Q- based resource allocation module (Q-ram)

The Q-learning method is used significantly in the construction of the resource allocation module. Q-based reinforcement learning is selected despite the many limitations of the other algorithms that are now available. With the assistance of the Q-learning model, it is much simpler to learn traits that connect to various changes in the environment. As a direct result of this, the Markov decision processor (MDP) may now be seen as Q-learning, which is an approach to reinforcement learning. With the purpose of amassing rewards for it, the reinforcement agent engages in activities inside an environment. The agent performs action after receiving the state of the present environment and doing so in accordance with the state. The action causes a shift in the state of the environment, which is followed by the agent receiving feedback on the shift in the form of a reward. This MDP determines the parameters, such as the state action, the reward, and the chance that the event will take place. Let the current state be represented by, and the future state with an action value be represented by Eq. (1). The reward function for state z and z' is given as $R^{a}_{z_t z_{t+1}}$. t.and the existing state's total reward function may be described as Eq. (2), Let $Q_{\omega}(z, a)$ be the utility function with a policy variable ω that is diplayed in Eq. (3). Here, γ denotes the reduction factor that ranging from [0,1] and practically γ values are mostly [0.5,0.99] and it is displayed in Eq. (4) below.

$$P^{a}_{zz^1} = Pr\,o\,b\{z_{i+1} = z^1 \,| z_t = z, a_t = a\} \quad (1)$$

$$R_t = \sum_{z_{t+1} \in Z} P^{a_t}_{s_t s_{t+1}} \, R^{a_t}_{z_t z_{t+1} | z_t = z, a_t = a} \quad (2)$$

$$Q_{\omega}(z, a) = \{R_t + \gamma \sum z_{t+1} \in z P^{a_t}_{zz_{t+1}} Q_{\omega}(z_t\,, a)\} \quad (3)$$

$$\begin{aligned} Q * (z, a) &= max\{\, Q_{\omega}(z, a)\} \\ &= \left\{R_t + \gamma \sum_{z_{t+1} \in Z} P^{a_t}_{z_t z_{t+1}} \, max\{\, Q\omega(z_{t+1}, a)\}\right\} | Z_t = z, a_t = a \\ &= \{R_t + \gamma \sum z_{t+1} \in Z \, P^{a_t}_{z_t z_{t+1}} Q * (z_{t+1}, a)\} | z_t = z, a_t = a \end{aligned} \quad (4)$$

The agent will be notified of the change in the state of the environment via a reward once the reward function has been calculated using Eqs. (3) and (4). The calculation of the reward function relies on the change in the state of the environment. Algorithm 1 provides a visual representation of the operational mechanism of the Q-based RAM. Table 4 explains an algorithm-1for Proposed Q- based RAM

In the Algorithm-1, the reward function presents in reinforcement learning algorithm. Learning is set to low latency, which produces a faster response for allocating the data to the different resources. The Random Access Memory (RAM) is trained to identify the most appropriate nodes to carry out the incoming request.

Table 4. Algorithm-1for Proposed Q- based RAM

	Algorithm-1 // Working Mechanism of the Proposed Q- based RAM
01	Input : No of Input data
02	Output :Balanced Resource with Low Latency and High performance
03	For n=0 to Max_iteration
04	Create the Q-table Containing the States and Actions
05	Agent interacts with environment and updates the State-Action tables
06	Agent uses the Q-tables and views all the possible solutions, finds the best resource with the less latency using the equation(3) and (4)
07	Updates the values in Q-tables using (1) and (2)
08	Go to Step 03

2.3.3. ELM prediction mechanism

The proposed method makes use of ELMs, which are more often referred to as ELM, in order to provide predictions for a target field by using one or more predictors. In order for ELM to learn how to accurately forecast the chance of a heart attack occurring, it is trained using the training dataset. Following that, ELM will be analysed based on the sensory data supplied by the user through the IoT layer in order to calculate the likelihood of the user having a heart attack and choose the most appropriate next step. The next step is to create normalised versions of the sampled data, which are then input into extreme learning machines created by G.B. Huang [24]. In addition to their speed, velocity, and precision, these machines also have great degrees of speculation and exactness, and they have the ability to approximate universal functions. This method states that the "L" neurons of the hidden layer are connected with a constant activation function, despite the fact that the output layer is equated with being in line (for example, the sigmoid function). There is no need in ELMs for the individual tuning of the concealed layer. The weights of the concealed layer are determined by chance instead than being fixed (counting the bias loads).

The model is equated before any of the training data is taken into account. The equation that describes the system yield for an ELM with a single hidden layer may be found here by Eq. (5). Where x

Table 5. Hyper parameters used for Building the ELM network model

Sl.no	Hyper parameters	Specification
1	No of Hidden layers	200
2	No of Epochs	100
3	Learning Rate	0.001
4	Batch Size	20

represents an input, β denotes the Output weight vector, which is expressed as Eq. (6) and H(x) describe hidden layer output is defined as Eq. (7).

$$f_L(x) = \sum_{i=1}^{L} \beta_i h_i(x) = h(x)\beta \quad (5)$$

$$\beta = [\beta_1, \beta_2, \ldots\ldots\ldots \beta_L]^T \quad (6)$$

$$h(x) = [h_1(x), h_2(x), \ldots\ldots\ldots h_L(x) \quad (7)$$

The hidden layers are described by the Eq. (5), which is used for the estimate the O, which is the target vector. The ELM is built using the marginal non-linear property of the least square model, and its computes an Eq. (8). Where H∗ denotes the inverse of H is known as the Moore–Penrose generalized inverse can be reformulated as Eq. (9) and finally output function is estimated as Eq. (10).

$$\beta' = H * 0 = H^t\ (HH^T)^{-1}0 \quad (8)$$

$$\beta' = \mathrm{H}^{\mathrm{T}}\,(\tfrac{1}{\mathrm{C}}\mathrm{HH}^{\mathrm{T}})^{-1}0 \quad (9)$$

$$f_L(x) = h(x)\beta = h(x)H^T\,(\tfrac{1}{C}HH^T)^{-1}O \quad (10)$$

ELM makes use of the kernel function to achieve high levels of accuracy, which results in improved speed. The ELM has a number of benefits, the most important of which are reduced training errors and improved approximation. Applications in classification and prediction value analysis may be found for ELM due to its use of auto-tuning of the weight biases and non-zero activation functions. The training hyper parameters deployed for ELM prediction network is presented in the Table 5.

3. Implementation and evaluation

Applications are executed using the proposed technique in fog devices that are located between the cloud and end devices. In order to investigate the advantages of using cloud computing and edge computing in terms of the distribution of data and the reduction of latency, an experimental setting has been established. The IoTs sensors are located at a lower layer in the paradigm, and it is their job to receive and send data to higher levels through gateways. This is performed in line with the paradigm. Each network application has its topology, which provides the filtering and analysis of medical data collected from the lower IoT layers. Table 6 expalins a Software used for experimentation. Table 8. Explains the mathematical expression for the calculating time based performance metrics.

Table 6. Software used for experimentation

Sl.no	Software used	**Source**
1	IFogSim	https://github.com/harshitgupta1337/fogsim
2	Eclipse IDE for JAVA Developers	http://www. eclipse.org/downloads/packages/release/Mars/2
3	Anaconda v 3.2	http://www.anaconda.org/distribution/downloads
4	TensorFlowv 2.6.0	https://www.tensorflow.org
5	Keras v2.56	

The simulation and modeling of FC environments is achieved through the use of IFogSim. It analyses the efficiency of different scheduling algorithms and methods of resource management inside the FC context. Its primary use is in the process of calculating a variety of variables, including operating cost, power consumption, delay, and network congestion. Additionally, Eclipse Modeling tools, Python 3.9 with Tensorflow, and Keras are used for modeling the proposed network. The complete algorithm is implemented in a PC workstation with an Intel I9 CPU, a 256GB SSD, 8GB RAM, and a 16GB NVIDIA Tesla GPU. The data are recorded in CSV format and used for training and testing. The performance of the proposed model is calculated and analyzed using mathematical expressions listed in Table 7.

In addition, the effectiveness of the suggested algorithm was evaluated and compared with that of other algorithms already in existing in terms of time-related metrics such as TSR (Time Service Request), WT (Waiting Time), and ARU (Average Resource Utilization). Table 7 presents the mathematical phrase that was used for the purpose of computing

Table 7. Mathematical expression for performance metrics calculation

SL.NO	Performance Metrics	Mathematical Expression
01	Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
02	Sensitivity or recall	$\frac{TP}{TP+FN} \times 100$
03	Precision	$\frac{TN}{TP+FP}$

Table 8. Mathematical expression for the calculating time based performance metrics

Sl.No	Performance Metrics	Mathematical Expression
1	TSR(Time Service Request)	CT-AT where CT is Completion time and AT is Arrival time
2	WT(Waiting Time)	TSR- BT where BT is Burst time for which the process requires for execution.
3	Average Resource utilization	LBL= BS*OVL/BS where BS is Number of Balanced Nodes and OVL is total overloaded number of nodes in network

each metric. In order to demonstrate that the ELM prediction algorithm that has been suggested is superior, performance metrics are computed using the number of datasets and tasks that are present in the network. The effectiveness of the enhanced learning machine that was developed is evaluated and compared with that of other algorithms already in use, including PNN, support vector machines (SVM), decision trees (DT), random forest (RF), and K-nearest neighborhood (KNN).

The prediction accuracy of the different algorithms using real-time simulated data is shown in Fig. 3, whereas using mHealth datasets to predict heart attacks is shown in Fig. 4. In both cases, as testing data increases, the performance of the existing algorithm degrades gradually[23]. In contrast, the PNN and ELM have produced better performance when handling larger datasets. But the ELM has better stability than the PNN when testing samples are given in large numbers. ELM degrades only 0.5% of performance, whereas PNN degrades 2.5% as the

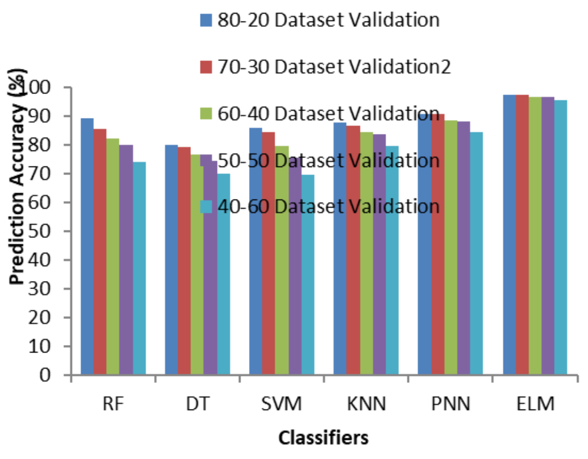

Figure. 3 Performance of different models in predicting the heart attacks using the real time datasets

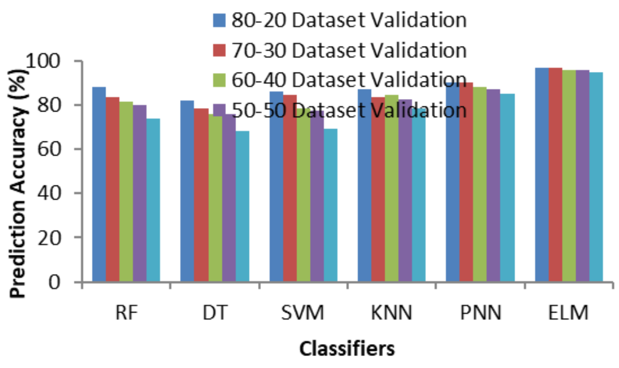

Figure. 4 Performances of the different models in predicting the heart attacks using the mHealth datasets

datasets increase. The auto-tuning property of ELM has produced a considerably better performance in handling a larger number of tasks.

The precision and recall metrics of the different algorithms are presented in tables 9, 10,11, and 12. From the tables; it is clear that the proposed ELM has shown more promising performance for the increased number of samples than other algorithms, such as PNN, SVM, DT, RF, and KNN. The stability of the ELM remains constant and degrades only 0.45% for real-time simulated data and 0.5% for mHealth datasets.

The experimentation has proven the proposed algorithm plays a vital role in predicting heart attacks, even though the number of tasks increases. Again, the performance of the proposed algorithm is compared with the other existing algorithms using the metrics mentioned in the table. In this evaluation, existing algorithms such as LC, RR, WRR, AWRR, and EPRAM are considered, as mentioned in Section-1. AUR analyses of the proposed algorithm using real-time simulated and mHealth datasets are shown in Figs. 5 and 6, respectively.

Table 9. Precision analysis for the different algorithms using real time simulated data

Data Samples	Precision Analysis (%)					
	RF	DT	SVM	KNN	PNN	ELM
80:20	89	80	86	87.5	90.5	97.3
70:30	85.4	79	84.5	86.4	90.5	97.3
60:40	82	76.5	79.5	84.5	88.4	96.4
50:50	80	74.3	75.3	83.7	88.0	96.4
40:60	74	70	69.5	79.5	84.5	95.6

Table 10. Precision analysis for the different algorithms using mHealth dataset

Data Samples	Precision Analysis (%)					
	RF	DT	SVM	KNN	PNN	ELM
80:20	88.4	82	86	87.3	90	97
70:30	83.5	78.5	84.5	83.5	90	97
60:40	81.4	75.7	78.5	84.5	88	96
50:50	80	75.9	77.3	82.5	87.4	96
40:60	74	68.5	69.5	78.4	85	95

Table 11. Recall analysis for the different algorithms using real time simulated data

Data Samples	Recall Analysis (%)					
	RF	DT	SVM	KNN	PNN	ELM
80:20	89	80	86	87.5	90.5	97.3
70:30	85.4	79	84.5	86.4	90.5	97.3
60:40	82	76.5	79.5	84.5	88.4	96.4
50:50	80	74.3	75.3	83.7	88.0	96.4
40:60	74	70	69.5	79.5	84.5	95.6

It is clear from both Fig. 5 and Fig. 6 that the remarkable performance that the suggested algorithm has obtained is a direct result of its use of deep RL and ELM. The deep learning strategy that was suggested has shown some significant improvement in terms of resource allocation. Calculations are done on the makespan for the proposed method, and it is compared to existing algorithms. The comparative results are shown in Fig. 7 (real-time simulated datasets) and Fig. 8 (mHealth datasets).

Table 12. Recall analysis for the different algorithms using mHealth dataset

Data Samples	Recall Analysis (%)					
	RF	DT	SVM	KNN	PNN	ELM
80:20	88.4	82	86	87.3	90	97
70:30	83.5	78.5	84.5	83.5	90	97
60:40	81.4	75.7	78.5	84.5	88	96
50:50	80	75.9	77.3	82.5	87.4	96
40:60	74	68.5	69.5	78.4	85	95

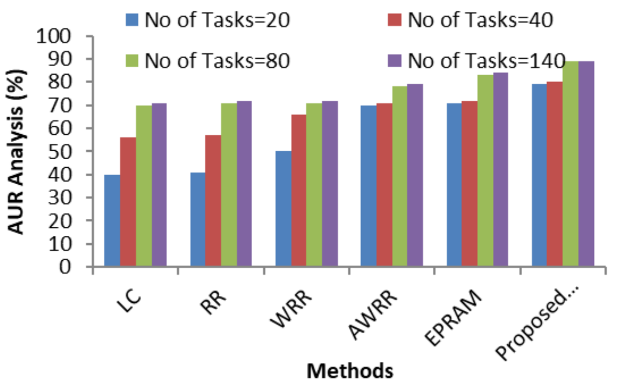

Figure. 5 AUR analysis for the different algorithms using real time simulated datasets

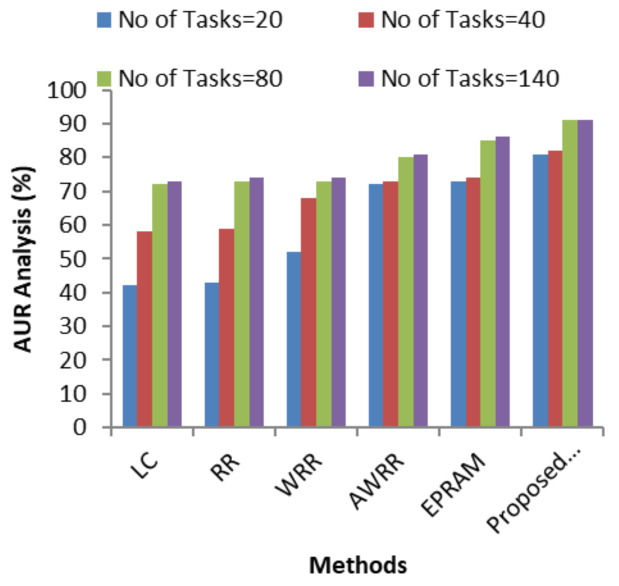

Figure. 6 AUR analysis for the different algorithms using mHealth datasets

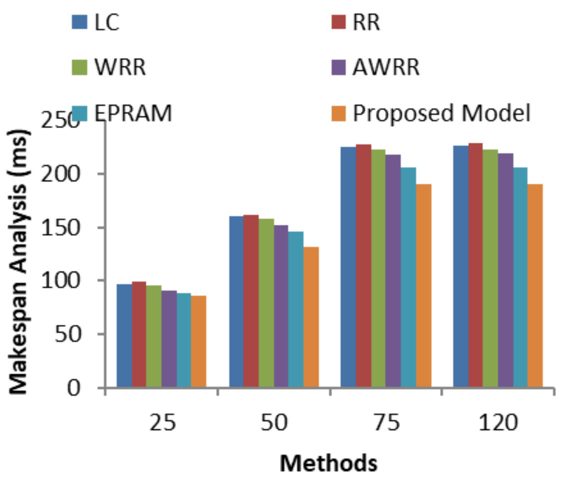

Figure. 7 Makespan analysis for the different algorithms using real-time simulated datasets

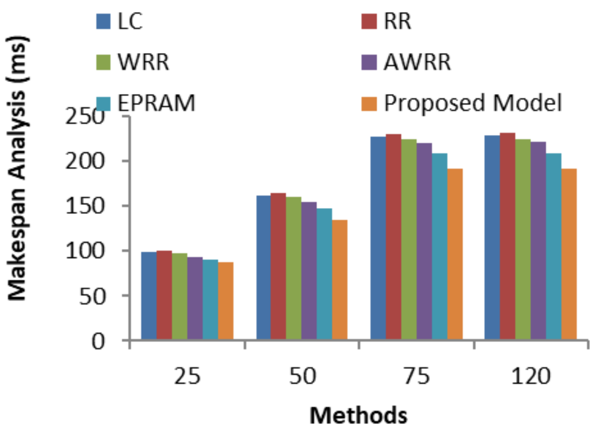

Figure. 8 MakeSpan analysis for the different algorithms using mHealth datasets

From Figs. 7 and 8, it is clear that the proposed model has shown better performance than the other existing algorithms. Since the RL deploys an agent that can allocate the task quickly and efficiently, Moreover, integration of ELM for predicting heart attacks takes lesser time than the other learning models, even for the larger datasets.

4. Conclusion and future development

This paper discusses the resource allocation technique in a fog-edge environment. The IoT layer is responsible for collecting the patients' data and forwarding it to the suitable fog-edge nodes. After collecting the data, fog-edge nodes deliver it to the cloud, which manages data transfers to appropriate resources and it provides an accurate prediction algorithm for heart diseases. In the proposed system, the DCU collects the data from the IoT layers and stores it in memory for further processing. DPU is responsible for sampling, categorizing, and balancing data into appropriate forms that can be used for additional training, testing, and validation. The intelligent prediction system uses ELMs to predict different heart attacks. Finally, ARAM comprises a deep Q-learning mechanism for reasonable resource allocation. It is estimated that around 15,000 datasets were created and used for the purposes of prediction and allocation. The proposed ELM has better stability than the PNN when testing samples are given in large numbers. ELM degrades only 0.5% of performance, whereas PNN degrades 2.5% as the datasets increase. The stability of the ELM remains constant and degrades only 0.45% for real-time simulated data and 0.5% for mHealth datasets. The deep learning strategy that was suggested has shown some significant improvement in terms of resource allocation. Calculations are done on the makespan for the proposed method, and it is compared to existing algorithms The performance of the model is compared with several performance metrics to show that the proposed algorithm is better than other algorithms. The proposed model may improve security and load-balancing techniques in future.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, Ramya R and Ramamoorthy S; Methodology, Ramya R and Ramamoorthy S; Validation, Ramya R and Ramamoorthy S; Formal analysis, Ramya R; investigation, Ramamoorthy S; resources, Ramya R and Ramamoorthy S; data curation, Ramya R and Ramamoorthy S; writing—original draft preparation, Ramya R; writing—review and editing, Ramya R and Ramamoorthy S; visualization, Ramya R and Ramamoorthy S; supervision, Ramamoorthy S;

References

[1] M. Alam and Z. A. Khan, "Issues and challenges of load balancing algorithm in cloud computing environment", *Indonesian Journal of Science and Technology*, Vol. 10, No. 25, pp. 1-12, 2017, doi: 10.17485/ijst/2017/v10i25/105688.

[2] H. F. Atlam, R. J. Walters, and G. B. Wills, "Fog computing and the internet of things: A review", *Big Data and Cognitive Computing*, Vol. 2, No. 10, pp. 1-18, 2018, doi: 10.3390/bdcc2020010.

[3] Y. Bengio, "Learning deep architectures for AI", *Foundation and Trends in Machine Learning*, Vol. 2, No. 1, pp. 1-27, 2009, doi:

10.1561/2200000006.
[4] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 8, pp. 1798–1828, 2013, doi: 10.1109/TPAMI.2013.50.
[5] K. Chen, K. Franko, and R. Sang, "Structured model pruning of convolutional networks on tensor processing units", *Computer Science*, Vol. 2107, No. 04191, pp. 1-6, 2021, doi: 10.48550/arXiv.2107.04191.
[6] L. Deng and D. Yu, "Deep learning: Methods and applications", *Foundation and Trends in Signal Processing*, Vol. 7, Nos. 3-4, pp. 197–387, 2014, doi: 10.1561/2000000039.
[7] S. Dubey, M. Dahiya, and S. Jain, "Implementation of load balancing algorithm with cloud collaboration for logistics", *Journal of Engineering and Applied Science*, Vol. 14, No. 2, pp. 507-515, 2019,
[8] Q. Fan and N. Ansari, "Towards workload balancing in fog computing empowered IoT", *IEEE Transactions on Network Science and Engineering*, Vol. 7, No. 1, pp. 253-262, 2018, doi: 10.1109/TNSE.2018.2852762.
[9] T. N. Gia, M. Jiang, A. M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "Fog computing in healthcare Internet of Things: A case study on ECG feature extraction", In: *Proc. of IEEE International Conference on Computer and Information Technology*, pp. 356–363, 2015, doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.51.
[10] K. Gu, H. Liu, Z. Xia, J. Qiao, W. Lin, and D. Thalmann, "PM2.5 monitoring: use information abundance measurement and wide and deep learning", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 32, No. 10, pp. 4278-4290, 2021, doi: 10.1109/TNNLS.2021.3105394.
[11] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things: a vision, architectural elements, and future directions", *Future Generation Computer Systems*, Vol. 29, No. 7, pp. 1645–1660, 2013, doi: 10.1016/j.future.2013.01.010.
[12] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "iFogSim: a toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments", *Journal of Software : Practice and Experience*, Vol. 47, No. 9, pp. 1275–1296, 2017, doi: 10.1002/spe.2509.
[13] S. Gupta, S. Rani, A. Dixit, and H. Dev, "Features exploration of distinct load balancing algorithms in cloud computing environment", *International Journal of Advanced Networking and Applications*, Vol. 11, No. 1, pp. 4177–4183, 2019.
[14] K. Pareek, P. K. Tiwari and V. Bhatnagar, "Fog computing in healthcare: A review", *IOP Conference Series: Materials Science and Engineering*, Vol. 1099, No. 1, pp. 1-12, 2021, doi: 10.1088/1757-899X/1099/1/012025.
[15] M. Bhatia, and S. K. Sood, "A comprehensive health assessment framework to facilitate IoT-assisted smart workouts: A predictive healthcare perspective", *Computers in Industry*, Vols. 92-93, pp. 50-66, 2017, doi: 10.1016/j.compind.2017.06.009.
[16] T. N. Gia, M. Jiang, V. Sarker, A. M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "Low-cost fog-assisted health-care IoT system with energy-efficient sensor nodes", *International Wireless Communications and Mobile Computing Conference*, pp. 1765-1770, 2017.
[17] A. Elhadad, F. Alanazi, A. I. Taloba, and A. Abozeid, "Fog computing service in the healthcare monitoring system for managing the real-time notification", *Journal of Healthcare Engineering*, pp. 1-12, 2022.
[18] A. Kumari, S. Tanwar, S. Tyagi, and N. Kumar, "Fog computing for healthcare 4.0 environment: Opportunities and challenges", *Computers and Electrical Engineering*, Vol. 72, pp. 1-3, 2018.
[19] H. Wadhwa and R. Aron, "TRAM: Technique for resource allocation and management in fog computing environment", *Journal of Supercomputing*, Vol. 78, No. 1, pp. 667-690, 2022.
[20] M. A. Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei, and N. Kumar, "Energy-aware marine predators' algorithm for task scheduling in IoT based fog computing applications," *IEEE Transactions on Industrial Informatics*, Vol. 17, No. 8, pp. 5068–5076, 2020.
[21] S. Wang, T. Zhao, and S. Pang, "Task scheduling algorithm based on improved firework algorithm in fog computing", *IEEE Access*, Vol. 8, pp. 32385–32394, 2020.
[22] B. Wang, S. Huang, J. Qiu, Y. Liu, and G. Wang, "Parallel online sequential extreme learning machine based on MapReduce", *Neurocomputing*, Vol. 149, No. A, pp. 224-232, 2015.
[23] F. M. Talaat, "Effective prediction and resource allocation method in fog computing

environment for smart healthcare system", *Multimedia Tools and Applications*, Vol. 81, No. 6, pp. 8235–8258, 2022.

[24] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: theory and applications", *Neurocomputing*, Vol. 70, Nos. 1-3, pp. 489–501, 2006.