Chapter 3
Implementing and Evaluating Health-as-a-Service (HaaS) in Fog Computing

In this Chapter, the real-time ECG based analysis system is implemented which analyzes the real-time ECG signal to find abnormalities in it. This analysis is very critical in terms of time and is highly delay-sensitive, as the patient's life can be in grave danger. The ECG based Health Care system is first implemented on Cloud Computing and different QoS parameters are studied and recorded. The same system is then migrated to Fog Computing and again the QoS parameters are observed. Different QoS parameters in the context of Health Care are explained and discussed. Health-as-a-Service is introduced as a separate service, it is different from IaaS and PaaS. Because to run Health care applications, exclusive context based implementation is needed with dedicated hardware. This is to avoid the processing delays due to Cloud Computing techniques like virtual machine migration viz. To perform the Fog computation Raspberry Pi model 3 B+ is chosen, based on its characteristics. To compare these two architectures, the ECG analysis system is made to run simultaneously on both architectures. The common source of ECG sends the signal to both the Fog and the Cloud Computing nodes. Both nodes do same set of calculations of ECG intervals and analyze the signal in terms of abnormalities. And if the signal is found abnormal then the doctor, hospital staff and patient's relatives are notified by the SMS service. In case of an abnormality in the ECG, if the Short Message Service (SMS) signal reaches early then the chances of the patient getting a faster treatment increases and life can be saved. Fog Computing also helps to save energy in terms of power and processing as it supports green computation and is explained in brief in the upcoming sections. Basically, this paper focuses on the use of Fog Computing as the better and more efficient implementation of health care solutions as its use will reduces the Cloud burden in terms of redundant data upload by saving network bandwidth.

3.1 Why ECG analysis is very crucial?

ECG signal abnormalities can indicate medical emergencies like myocardial infarction and arrhythmia. As ECG signals are used to measure different heart functions, abnormality in the signals can be an indicator of the onset of cardiovascular disorders. Abnormalities signal can be caused by several issues like-

- Defects in the heart's size and/or shape: If certain regions of the heart's walls are larger as compared to others, pumping blood becomes more difficult resulting in abnormalities in the ECG readings.
- Imbalance of electrolytes: Electrolytes maintain the heart beating rhythm. An abnormal ECG signal may signify an imbalance in the natural levels. Such imbalances may cause neuropsychiatric manifestations [58].
- Heart attack: Heart attacks affect blood flow in the heart causing tissue death due to oxygen loss [59]. Dead tissues do not conduct electricity leading to an abnormal ECG signal.

Moreover, malignant spontaneous ventricular arrhythmias (ventricular fibrillation and ventricular tachycardia) can result in sudden cardiac arrest. ECG signals can reflect underlying ventricular arrhythmias [60] thus predicting possibility of cardiac arrest. Patients who have coronary artery disease or are susceptible to ventricular arrhythmias always have a risk of sudden cardiac death. Sudden cardiac death can be predicted by early detection of abnormalities in ECG report parameters [61-62]. All these serious health risks can be identified at an early stage by early detection of abnormalities in the ECG signal. Early detection of such fatal health risks is very important as lifesaving treatment can be administered immediately. A few seconds can be the difference between life and death. First few seconds after a sudden cardiac arrest occurs are of utmost importance. Cardiac arrests disrupt the heart's pumping action causing lack of blood to organs and seconds later the patient becomes unconscious and has no pulse [63]. The patient dies if treatment is not administered immediately.

3.2 What is Health-as-a-Service (Haas) and why it is better supported in Fog Computing?

Cloud Computing is well known for its main services like Infrastructure as a Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). In all of these services large volume of data moves to and fro between your devices to the cloud servers and cloud data centres. These services are totally based on the service of your service provider. If the internet bandwidth fluctuates these services are affected and causes delays in the delivery of services. But more or less these services works very well for non-real time services where the delays in milliseconds in decision making will not impact the business model.

If we use Cloud Computing services to process real-time health care data then it is very advantageous. The 24 X 7 monitoring of signals are possible, system becomes less dependent from full time health care giving staff, it is cost effective and the data patient data is accessible anytime, anywhere by the health care staff. If such services are made web based then it makes any data access away by few clicks. Nowadays, most of the web based applications are supported by almost all computing platforms which even gives more ease in accessing the remote data. But when real-time critical health care data like ECG analysis is processed on the cloud servers, it may delay the decision making because of dynamic resource allocation. The dynamic resource allocation may get delayed [64] when two applications are accessing the same resource, when there is a scarcity of a resource, when resources are isolated and resource fragmentation arises, few applications have allocated more resources then the need and applications are assigned fewer resources than the demand.

If such delays get occurs in dynamic resource allocation or cloud processing due to virtual machine migration then the decision in the health care services are delayed and the QoS in Health care service gets affected and is delayed causing life threat to patient. The Health care services should be based on real-time decision making which is as early as possible. It should not tolerate any sense of delays in decision

making due to resource allocation or due to transmission delays caused by network bandwidth issues.

To overcome the above limits of the Health care service in Cloud Computing, the Health-as-a-Service in Fog Computing is proposed. This service will be implemented on the dedicated Fog Node which makes independent of internet connectivity issue and takes very less transmission time to read data on the Fog Node. Plus, the node is dedicated node, meant only to process health care data with enough resources to avoid dynamic resource allocation problems.

3.3 Reading ECG signal using Einthoven's Triangle

ECG waveforms represent the heart's depolarization and repolarization. Different numbers of bipolar and unipolar leads are used to measure ECG. The numbers can vary as 3 lead, 4 lead, 5 lead, 6 lead and 12 lead [65]. The advantage of the three lead approach, however, is that it is more convenient, pervasive and IoT friendly.

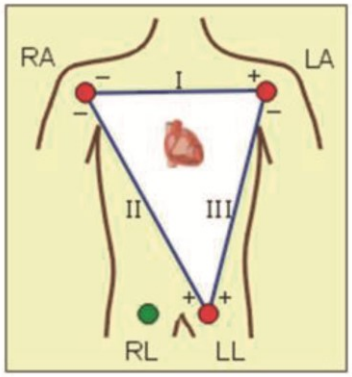

Figure 3.1 Einthoven's Triangle

Three sensors are used to record voltage coming from 3 limb electrodes. These electrodes are placed on right arm, left arm and left leg forming a triangle. This triangle is known as Einthoven's Triangle shown in figure 3.1 [66]. Einthoven's law states that sum of LEAD I and III are equal to that of lead II. Also, the polarity of lead II is anticlockwise whereas that of lead I and III is clockwise [67].

3.4 Pre-processing to De-noise the ECG Signal

While recording the ECG signal, due to environmental factors, noise gets induced in it. The noisy ECG signal cannot be processed in the digitized wave as it considers the value of amplitude of the reference points. There exist multiple pre-processing techniques to denoise the ECG signals. They are mainly classified into hardware and software-based approaches. In hardware-based approach, one has to create different filters like band-pass and band-reject. These filters are the combinational circuits of L-C-R. These filters are mainly Adaptive Notch Filter, High pass IIR filter, Zero phase filter.

In software-based filter, concept of moving window is used. This window moves along the points and performs operations like averaging to smoothen the curve. One such popular filter is Savitzky-Golay filter. This filter is known for preserving ECG characteristics while removing the noise. One noised ECG signal is shown in figure 3.2.

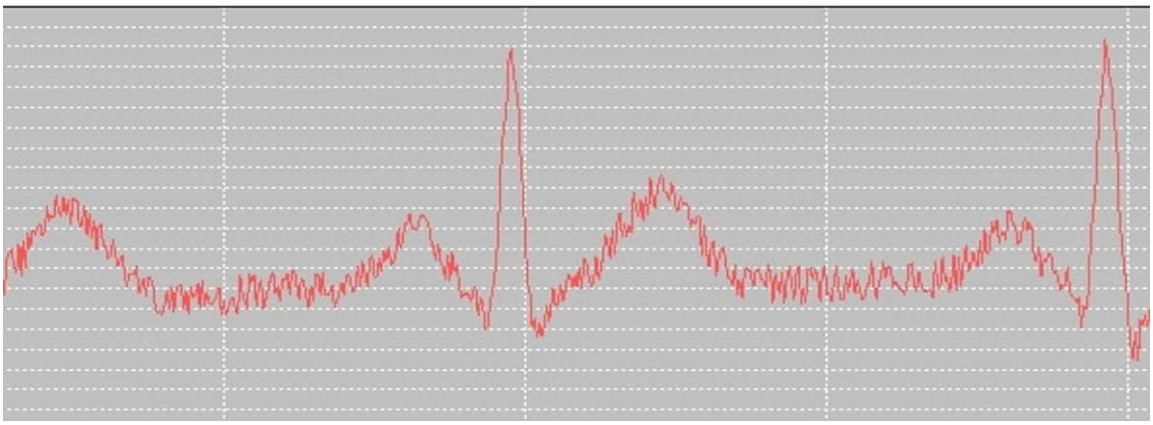

Figure 3.2 ECG signal with noise

The above signal is taken from MIT ECG Simulator [68]. After applying the Golay Filter the ECG signal is denoised which is shown in figure 3.3. But while applying the Golay Filter, deciding its window size and the degree of polynomial is very important. If these parameters are chosen wrongly then it will affect the result performance. For current transformation of signals the polynomial degree is 3 and the window size is 21.

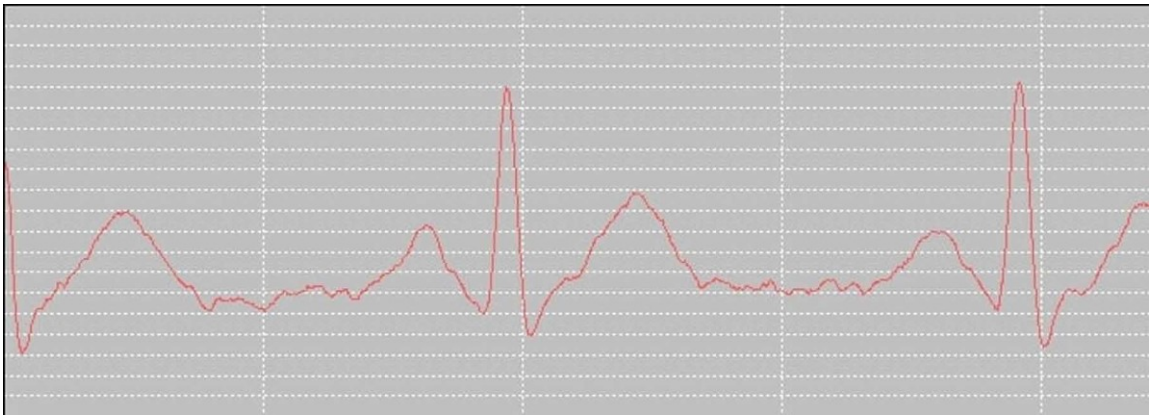

Figure 3.3 Denoised ECG signal after applying Savitzky-Golay Filtering

3.5 AD8232 ECG Sensor IC

AD8232 [69] is a tiny chip that measures the electrical conduction of human heart and then converts this conduction to a waveform known as ECG. This ECG signal is analyzed further to determine any health issues in human heart (Figure 3.4).

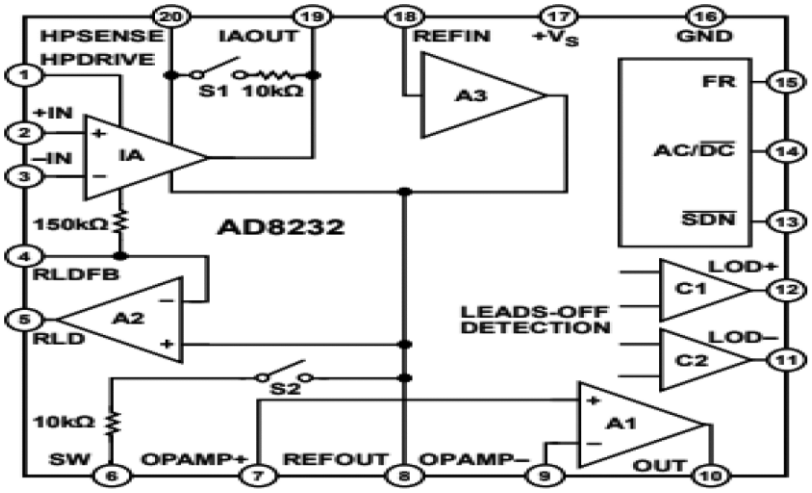

Figure 3.4 AD8232 Functional Block diagram

The IC mainly consists of the following

1. A specialized Instrumentation amplifier
2. An operational amplifier
3. A right leg drive amplifier
4. A mid supply reference buffer
5. High pass and low pass filters

Application: AD8232 is also known as an ECG module. ECG is a time VS voltage graph. Heart activities are measured with ECG. This is done by placing electrodes on left chest, right chest and right leg. This module can also be used for Electromyography (Study of activity in the skeletal muscles). The signal given by this IC is noise free if proper recording care is taken.

3.6 Decision Making in ECG Signal as Normal or Abnormal

Electrocardiogram (ECG) is a graph generated by the electrical activity of the heart muscles. Through ECG one can determine whether the heart is working in normal conditions. Whenever the heart is not working in normal conditions the ECG rhythm changes which is called "Arrhythmia". ECG helps us to detect abnormal heartbeats, the status of blood supply in the heart due to cholesterol clogging and enlargements in the heart [70].

PR and QT are the main intervals in the ECG wave. A PR wave is generated when the left atrium receives an electrical impulse from the right atrium. The QRS complex gets generated when both ventricles begin to pump, and at this time, a "beep" sound is generated in the cardiac monitors. When the initial contraction is over it results in ST-segment generation. And when ventricles are relaxing, the T wave is generated. The normal beats are from 60-100 with these intervals present in the ECG waveform [71]. These intervals are shown in figure 3.5 [72]. After receiving the ECG signal, we can use different algorithms and approaches like wavelet transformations, wavelet analysis, Pen and Tomkins algorithm, template-based matching, QRS peak detection, neural networks or windowing algorithms to process and analyze the same.

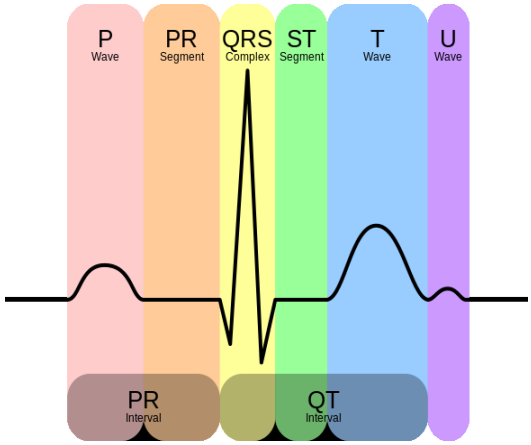

Figure 3.5 Complete ECG Wave with intervals

ECG signals can be analyzed in frequency or in the time domain. The Frequency domain algorithm needs complex mathematical calculations and different transformations, while in the time domain by using a simple windowing algorithm with precise calculations the ECG wave intervals can be found out. In this paper, the Novel windowing algorithm developed by Muhammad et al. in [73] is used and is modified partially to analyze the ECG wave. In this system, the live ECG signal is captured [74] using three AD8232 electrode-based sensor and Analog to Digital Converter (ADC) is done by Arduino Nano. Arduino is a small micro-controller capable of doing many things. It has multiple analog pins. And in the Arduino library, we have functions like AnalogRead(), which will read the analog input from the particular pin of Arduino and will convert it into a digital value ranging in 0-1023. It has a 10-bit binary resolution. Here the ECG readings obtained in real-time using the AD8232 sensors are in the range of 0-1023. These readings when plotted show the actual ECG wave. In this, the sampling frequency (fs) used is 500 Hz, which is between two points the time interval is 2 ms. The recorded ECG signal is plotted in figure 3.6 and it is in the range of 0-1023.

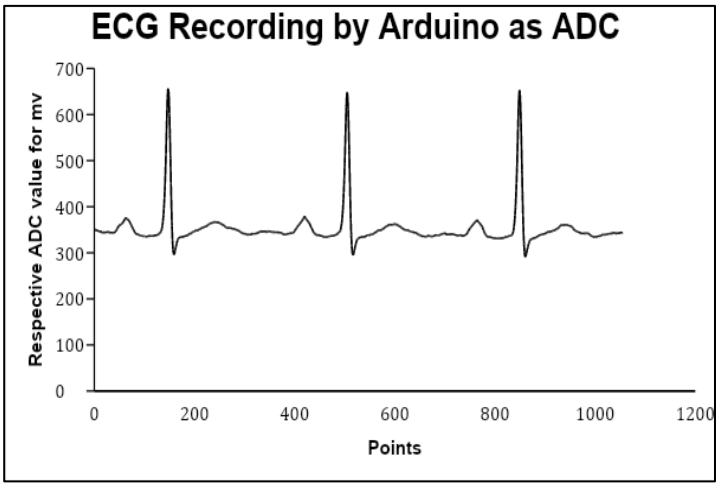

Figure 3.6 ECG wave as discrete signal recorded with AD8232 and after ADC by Arduino

For the proposed system, the obtained real-time ECG records have an R peak above 600 thresholds. The ECG readings are stored in the moving array where each index represents one respective voltage value and the time gap between the adjacent indices is 2 ms. ECG is a periodic wave, it repeats its cycles after a certain interval of time. Now the important part is to get PR, QRS and QT intervals out of these waves for each and every wave.

Table 2: Standard ECG Intervals for a healthy adult with standard bpm of 60

Intervals	Normal Value	Normal Variation
QT Intervals	400 ms	±40ms
QRS Interval	100ms	±20ms
PR Interval	160 ms	±40ms

Based on the given table we can find the time intervals in milliseconds and by comparing with table 2 [75-78] we can find whether ECG waves are normal or abnormal. The normal beats per minute (bpm) are 60 to 100 bpm. Further, we have used the windowing algorithm to detect different intervals. All real-time readings are stored in the array and the intervals are found by using the following algorithm.

Algorithm 1 Windowing Algorithm

Input: ECG wave, **Output:** ɳ

// ɳ ∈ {normal, abnormal}

// i is each ECG wave ranging from P-T-R

1. **procedure** detect
2. **for** i from 1 to n **do**
3. R-R interval is $t_{rr} = \frac{R_{(i+1)} - R_{(i)}}{f_s}$
4. P-R interval is $t_{pr} = \frac{R_{(i)} - P_{(i)}}{f_s}$
5. QRS interval is $t_{qrs} = \frac{(S_{(i)}+8) - (Q_{(i)}-8)}{f_s}$
6. QT interval is $t_{qt} = \frac{T_{(i)}+(t_{rr}*0.13)-(Q_{(i)}-8)}{f_s}$
7. bpm $= \frac{t_{rr}*60}{f_s}$
8. **end for**
9. **end procedure**

The ECG arrhythmia detection is implemented using three approaches. The first is by using simple windowing algorithm, the second is by using time-series augmented signals and the last method is by using ECG image classification. While implementing these methods the improvement in the result quality is also taken care of.

3.6.1 Recognizing Real Time ECG Anomalies Using Arduino, AD8232 and Java

The functioning of Heart can be checked by continuous monitoring of heartbeats through Electrocardiogram (ECG). Irregularity in the rhythm of the heartbeat results in arrhythmia. Arrhythmia can be classified based on the origins that cause it. ECG signal comprises of PQRST wave. Analysis of PQRST wave helps identifying the type of arrhythmia. Thus, real-time analysis of ECG is of utmost priority, to acquire immediate medical aid and to avoid fatality. The work discusses the use of the AD8232 sensor to capture ECG signals and its interfacing with Arduino Nano. Arduino is used as a Sampler and Analog to Digital Converter (ADC). The intervals of PQRST wave is analyzed using Java APIs and windowing algorithm. The results are compared with standard ECG signals to detect abnormalities and further analysis. It aids the reader to understand and develop a hand and low cost ECG analysis system, thus, reducing the treatment costs. In this section, authors have successfully proposed and implemented a system in Java that detects real-time anomalies in heartbeat. This system can be used by medical practitioners in both real-time and as well as in static modes. While both the modes require sampling

frequencies, the real-time mode also makes use of a port connected to Arduino, a highly effective IoT unit.

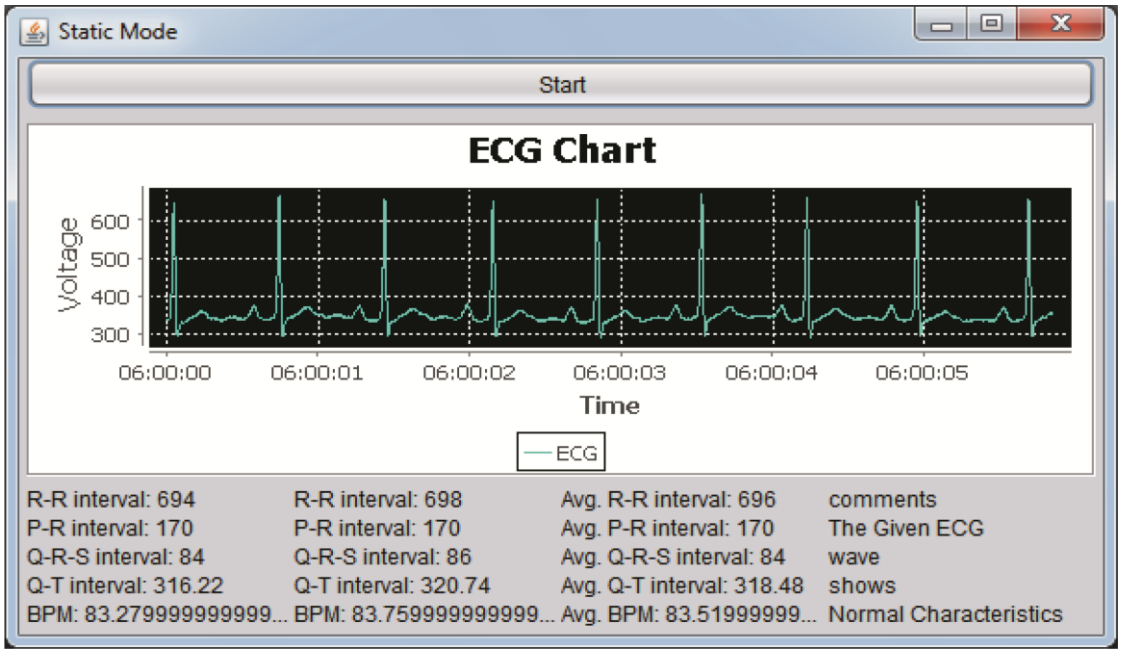

Figure 3.7 Final Output window showing ECG interval and its normality

The AD8232 senor is interfaced with Arduino. The final result displays the ECG signal on a Voltage-Time graph and also gives the R-R, P-R, Q-R-S and Q-T intervals. The practitioners can easily and very accurately detect anomalies from the output. The real-time output of the following system is as shown in figure 3.7. The detailed work is implemented and explained in [79].

3.6.2 ECG Heartbeat Arrhythmia Classification Using Time-Series Augmented Signals and Deep Learning Approach

Automated classification and identification of the ECG arrhythmia signal that provides faster and more accurate result is increasingly becoming the need of the moment. Various machine learning skills have been applied to advance the accuracy of results and increase the speed and robustness of the models. A lot of focus has been given to the architectures and datasets employed but pre-processing of the data being equally important. In this, a pre-processing technique that significantly improves the accuracy of the deep learning models used for ECG

classification is proposed with a modified deep learning architecture that adds to the training stability. With this pre-processing technique and deep learning model, the system is able to attain accuracy levels of more than 99% without overfitting the model.

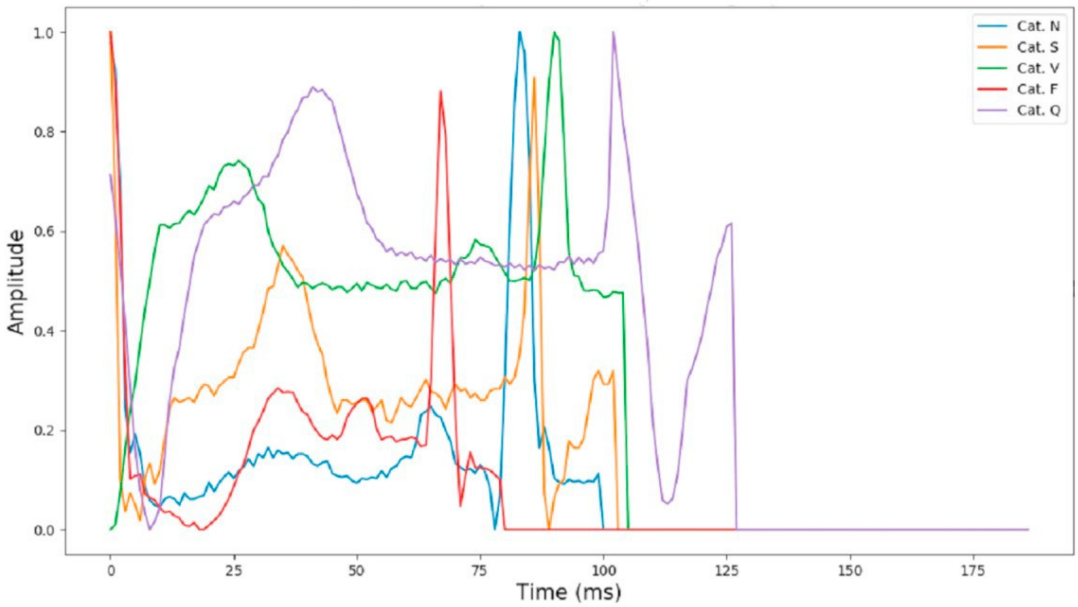

Figure 3.8 ECG signals from all five categories.

Applying augmentations to the dataset can not only make the model training more accurate but also stabilize it at higher accuracies. The proposed model consists of 6 residual blocks which means there is scope of overfitting the data but the augmented dataset also prevents overfitting by making classification difficult in the testing phase. The proposed model still displays high accuracy in such conditions. Thereby depicting its calibre to make highly accurate predictions with an accuracy rate of 99.12%. Here, five classes of ECG signals are first pre-processed and then classified. The complete work with all pre-processing techniques and final results are published in the paper [80].

3.6.3 ECG Image Classification using Deep Learning Approach

The proposed methodology does ECG Arrhythmias Classification by CNN, trained on grayscale images of R-R interval of ECG signals. Outputs are strictly in the terms of a label that classify the beat as normal or abnormal with which abnormality. For

training purpose, around one lakh ECG signals are plotted for different categories and out of these signal images, noisy signal images are removed, then Deep Learning Model is trained. An image-based classification is done which makes the ECG Arrhythmia system independent of recording device types and sampling frequency. A novel idea is proposed that helps cardiologists worldwide, although a lot of improvements can be done which would foster a "wearable ECG Arrhythmia Detection device" and can be used by a common man. Proposed system aims to help cardiologists worldwide and with the developments of AI in the health care sector, our study will add value in this domain. This work proposes to help refine the vast clinical data, find patterns amongst them and to improve the accuracy parameter which is of paramount importance to serve a patient well. By implementing classification using Deep Learning, certain Machine Learning tasks like feature extraction and noise filtering have been avoided. Using the intense computational power to learn from data, workable accuracy has been achieved. The proposed model is achieving the accuracy level of 97.78 %. So, this work is to bridge the gap between technology and expertise in a health monitoring system in our case-Cardiology. The details like removing noise using Golay filter using software approach, the training and testing model with cropping and pre-processing techniques are mentioned in [81].

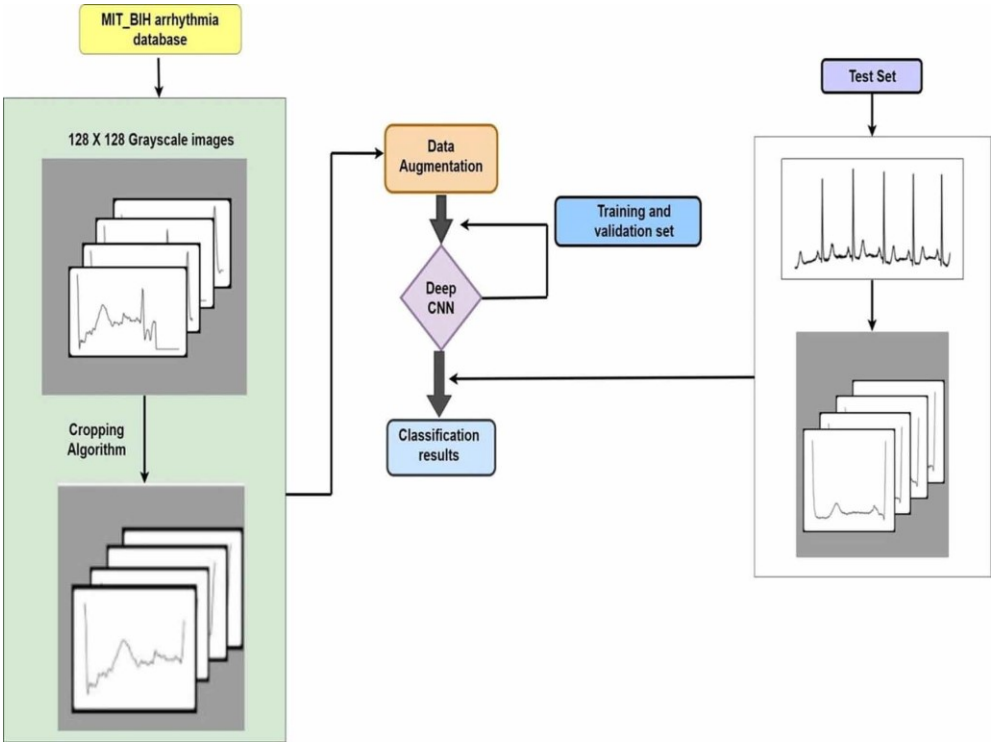

Figure 3.9 System architecture of ECG image classification using Deep Learning Approach

3.7 System Architecture

In the proposed methodology, a real-time health care system is taken into consideration. The system consists of a Data source, Fog Node, Gateway, Decision making, and messaging service. The real-time ECG signal is acquired from the patient [74]. These signals are recorded and sent to Cloud as well as the Fog Computing node. The windowing algorithm [73] is used to find the reference points PQRST in the ECG Signal. Based on these points the ECG time intervals are found out. Later decision making is done to find whether the given ECG signal is normal or abnormal. In the proposed architecture shown in figure 3.10, the real-time ECG signals are sent to the Cloud and Fog Node simultaneously.

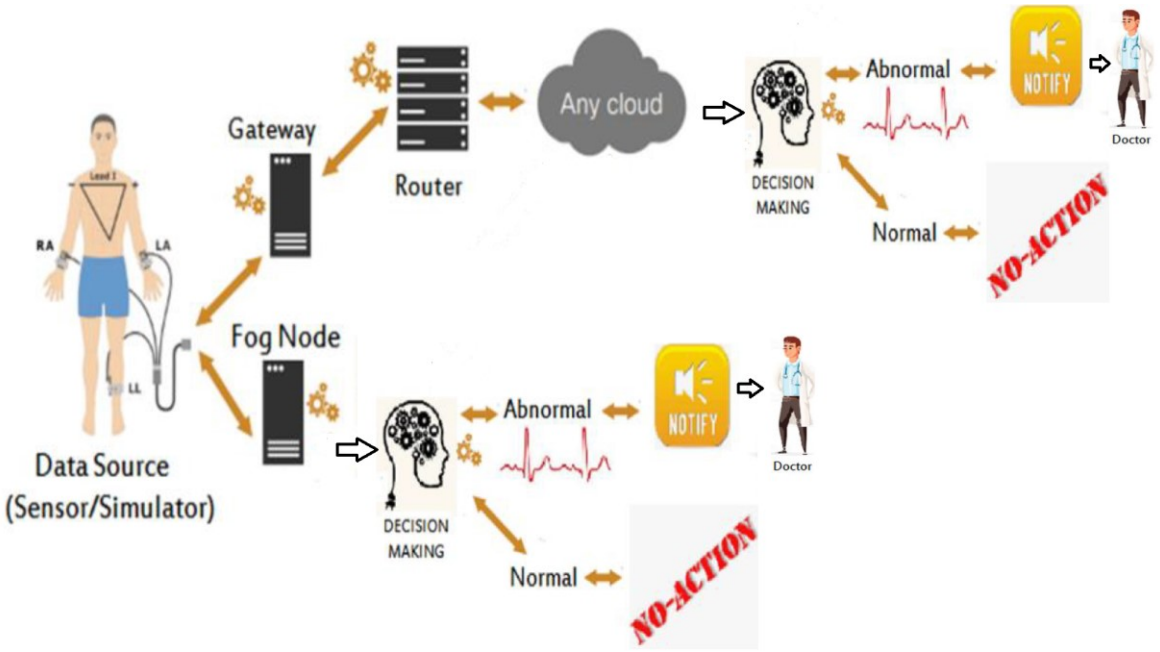

Figure 3.10 Cloud Computing and Fog Computing based Health Care System

The task of performing analysis is carried out on both the systems on the same signal. The generated results are then compared in terms of different parameters like computation time, transmission time, Carbon dioxide (CO2) generated and the total response time. Whenever an abnormality in the signal is found, at that very instant a text message consisting of the timestamp, signal interval values, and the patient data, is sent to the doctor and based on the timestamp the response efficiency is calculated.

3.8 QoS Parameters

Different QoS parameters like memory [82-83], Transmission delay, Computation delay, CO2 emission measurement [84-87], data transferred and Response time are as follows.

1) Memory: The amount of memory utilized by a programming module can be calculated using Java methods. One can use the Runtime class functions for finding the memory utilized.

long usedMemory= Runtime.getRuntime() .totalMemory()-Runtime.getRuntime().freeMemory()……(1)

The memory used for the ECG analysis is calculated as the difference of total available memory and the free memory.

2) Transmission Delay: Transmission delay is the total time taken by the network to send the data from one point (source) to the second point (Destination).

3) Computation Delay: Computation Delay is the total time taken for the computation. In the proposed system it is the time measure after the entire signal is received in the system until it is processed and the output as normal or abnormal is produced. Based on the computation time the Speed up in the Fog Computing can be calculated by

$$Speedup_{overall} = \frac{Execution\ time_{cloud}}{Execution\ time_{fog}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

Speedup helps to determine which computation is faster and how much. Here the overall speedup is showing how faster the fog decisions are with respect to the Cloud Computing. Here, the execution time is representing the overall execution time of Cloud and the Fog Node respectively in numerator and in the denominator.

4) CO_2 Measurement: Fog Computing uses far fewer resources than the Cloud Computing infrastructures which reduce the CO_2 generation. The amount of CO_2 that can be saved from Fog Computing is further explained. In 2011, According to Cisco 1.8 ZB of data was sent to the Cloud data centers. So, if 5.12 kWh of energy is required to send 1 GB data across then the total energy required to send 1.8 trillion GB of data is 9.216 trillion kWh of energy. To generate this much amount of energy, a total of 5.76 trillion kg of CO_2 is emitted. Using the above inferences, to transfer 1 Byte of data 2.98 x 10-3mgm of CO_2 is emitted.

Now, if one can adapt to Fog Computing and assume that the data only travels to the data center only for storage purposes, then one reduces the CO_2 emission by 50%, and the CO_2 emission can be potentially reduced by 2.88 billion metric tons.

That is almost 34 times more savings when compared to Cloud Computing. The amount of CO_2 can also be found out by finding the power used by the devices and relating CO2 with power.

$$Power_{static} = Current_{static} \times Voltage \quad \text{.....(3)}$$

$$Power_{dynamic} = \frac{1}{2} \times Capacitive\ load \times Voltage^2 \times Frequency\ switched \quad \text{.....(4)}$$

Static power is the power consumed by the device where no input is active and the dynamic power is the power consumed by the device when the input signal is active and output signal is getting processed/generated. The power consumption of any device is proportional to the amount of CO_2 generated indirectly. Here, current and the voltage are representing the input current and the voltage need of the device. Capacitive load is the capacitance of the wires and transistors used. And the frequency switched is the clock rate used.

5) Data Transferred: The data transferred is measured in bytes. Here, the ECG signals are sent to the Cloud and the Fog Node. The network distance is measured in terms of the number of Hops. Usually, the Fog Computing node is at least 2 to 3 hops away from the data source.

6) Response Time: For the proposed system architecture, the Response time considered is the overall response time of the ECG signal processing unit. Here it shows the total time span starting from the ECG signal generation until the final response is generated and given to the doctor. It includes the addition of other delay times like Processing Delay, Queuing Delay, Transmission Delay, and the Propagation Delay.

$$d_{response} = d_{proc} + d_{queue} + d_{trans} + d_{prop} \quad \text{.....(5)}$$

The final response time to get the output is the sum of processing, queueing, transmission and propagation delays respectively.

For the current system, the timestamp value of the Java programming language counts as the current system time in milliseconds which is noted by using the java method.

System.currentTimeMillis(); ……(6)

This function represents the current system clock time in milliseconds. It is a long data type number. Here, the final system improvement of the Fog Computing is shown by the difference of Response time taken by both the systems.

3.9 Different time stamps and defining the QoS parameters

In this system, different parameters are defined to measure the QoS parameters, and arithmetic relations among these parameters are used to calculate different delays. The timestamp Parameters with their denotations are as follows.

Table 3: Different Time stamp calculations

Different Timestamp instants	QoS Parameters
T_{gen} = ECG signal generation time	*Overall Response Time improved by Fog* = $T_{pc} - T_{pf}$
T_{rf} = Time at which ECG signal is reaching the Fog Node	*Transmission Delay (Fog)* = $T_{rf} - T_{gen}$
	Transmission Delay (Cloud) = $T_{rc} - T_{gen}$
T_{pf} = Time at which ECG signal is processed at Fog Node	*Fog Computation Time* = $T_{pf} - T_{rf}$
	Cloud Computation Time = $T_{pc} - T_{rc}$
T_{rc} = Time at which ECG signal reaches the Cloud Computing	*End to end Data Transferred = Data bytes x no. of Hops*
T_{pc} = Time at which ECG signal is processed by the Cloud Computing	*CO_2 generated = Data Transferred × emitted CO_2/Byte*

3.10 Experimental Results

The web interface is developed to understand the ECG signal processing and its analysis in more detail. The same web interface is run on both Cloud and the Fog Computing node. The interface shows the "Patient Name", a red colour label if signal Abnormality exists, the ".txt" file name where the ECG signals are stored for processing, different QoS parameters, ECG waveform, patient details, and different intervals for each ECG wave along with their abnormality. If the signal is found abnormal then the system will send SMS to the Health Care supporting staff. All

historic ".txt" files are also stored on both the nodes for future referral. The Cloud infrastructure chosen for the system is EverData.com.

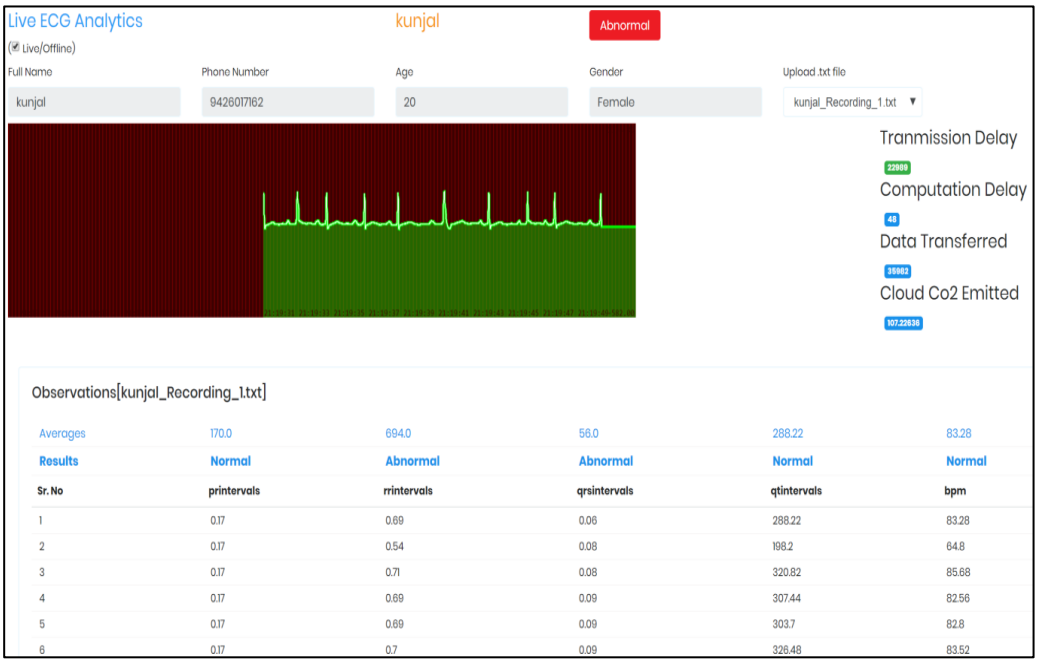

Figure 3.11 ECG Analyzing System interface on Cloud Computing

The same web interface is deployed on Raspberry Pi 3 b+ node which is acting as a Fog Computing Node. The difference one can see is, it is running on the localhost at the Gateway. The final responses of both the systems are conveyed by the SMSs, where each message contains the computing node from where the message has come, patient name, mobile number, the timestamp at which the message is generated, file name where the signals are stored and the respective ECG intervals which doctor and hospital staff can refer for advance preparations before patient arrival. The network delays from the service provider can affect the receiving time of the SMS, which may cause a delay in response.

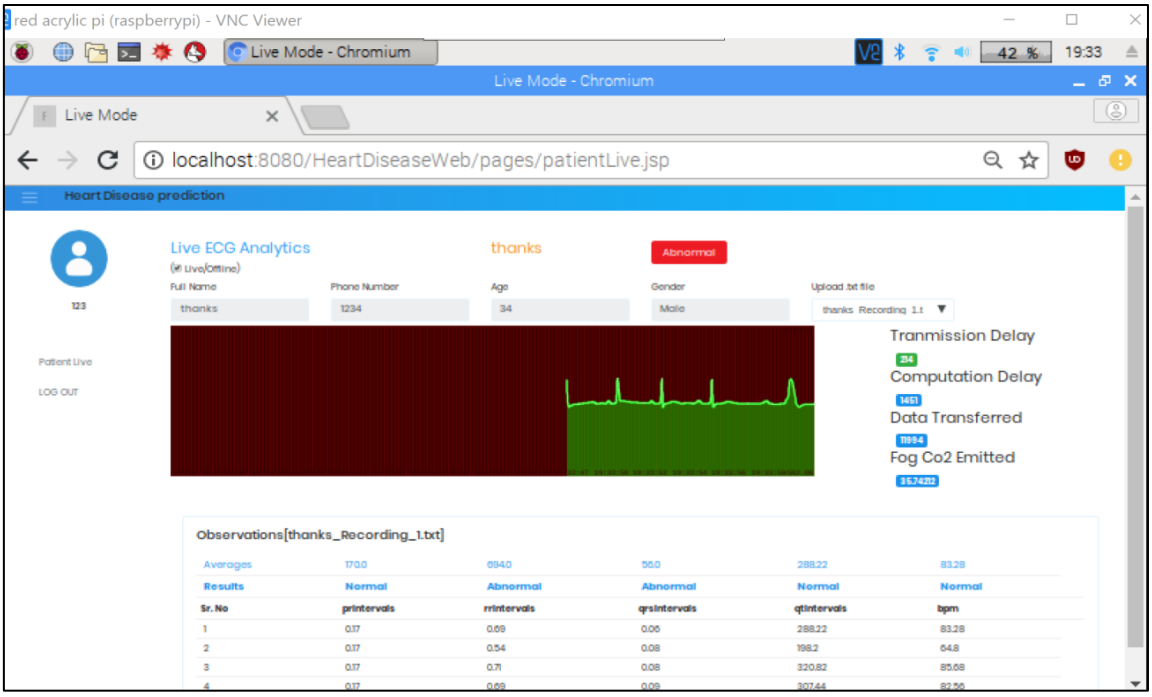

Figure 3.12 ECG Analyzing system interface on Raspberry Pi as a Fog Computing Node

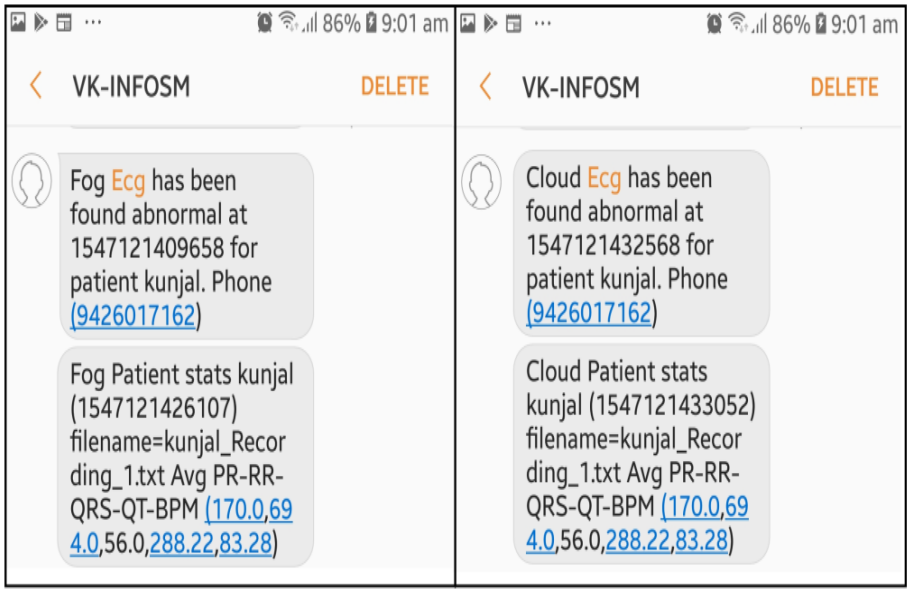

Figure 3.13 SMS responses for the Fog Computing and the Cloud Computing

When the millisecond timestamps shown in figure 3.13 are Converted [88] to actual times, they are 5.27.06 and 5.27.13 for Fog and Cloud respectively. This shows that the Fog Node is responding almost six seconds before the Cloud node which is very vital to save the patients when they are in critical condition.

The Fog and Cloud Computing system is first subjected only for 1 patient, and all resultant parameters are measured and shown in Table 4. The system shows that Fog Computing surpasses overall response time performance than Cloud Computing and performs better in terms of Response time, data transferred and CO2 generated. But Fog Computing is hanging back in terms of computation power. Cloud Computing takes only 55 ms to computer the given job while the Fog processor takes 670 ms.

Table 4: QoS parameters and their values

Parameters	Cloud Computing	Fog Computing	Improvement
Transmission Delay (ms)	7677	117	7560
Computational Delay (ms)	55	670	-615
Data Transferred (bytes)	35982	11994	23888
CO_2 Emitted (mgm)	107.22	35.74	71.48
Response Time in MS (Time Format in Java)	1547121433052	1547121426107	6945

The system is tested against by varying the number of patients to study the system behaviour in depth. Each and every parameter value is taken and shown as an average value for n patients to discuss further.

3.10.1 Transmission Delay

Transmission Delay depends on many factors like the number of hops between the source and the destination, available network bandwidth, layer conversion, VPN set up, wired-wireless configurations, congestions, tunnelling, and the number of users, etc. A hop occurs when a packet is passed from one network segment to the next. Data packets pass through routers as they travel between source and destination [89]. As WANs tend to have more number of devices connected to it and its huge network covering geographic areas, there is a slight drop in network bandwidth as the time taken to deliver the packet from source to destination has

increased. So as number of hops increases, the chances of bandwidth decreasing increases.

$$Bandwidth = \frac{Number\ of\ bits}{Time\ taken}\ bits/_{sec} \quad \text{.....(7)}$$

The bandwidth is defined as the ration of number of bits transferred and the time taken to transfer those bits. The number of hops in Fog Computing is one, but in case of Cloud Computing, the number varies as it is in the WAN. Different Cloud servers are tested for their distance by the number of hops and by using "tracert" command [90-92]. It is shown in Table 5. Of course, the number of hops count varies from place to place.

Table 5: Cloud server Distances in number of Hops

Cloud Servers	Number of Hops
Amazon Web Services	20
Microsoft Azure	12
Google Cloud Platform	09
IBM Cloud	06
Verizon Cloud	11
ThingSpeak	>30
EverData	11

From Table 5, it is clear that the cloud servers take more hops to reach. Hence by increasing transmission delays. The proposed system is tested for a different number of patients from 1-5. And the average transmission time is shown in figure 3.14. Figure 3.14 makes it clear that the transmission delay in terms of Cloud and Fog Computing remains almost the same for the same source and destination for a different number of patients. And it is also evident that the transmission delay in Cloud Computing is very much higher than the Fog Computing.

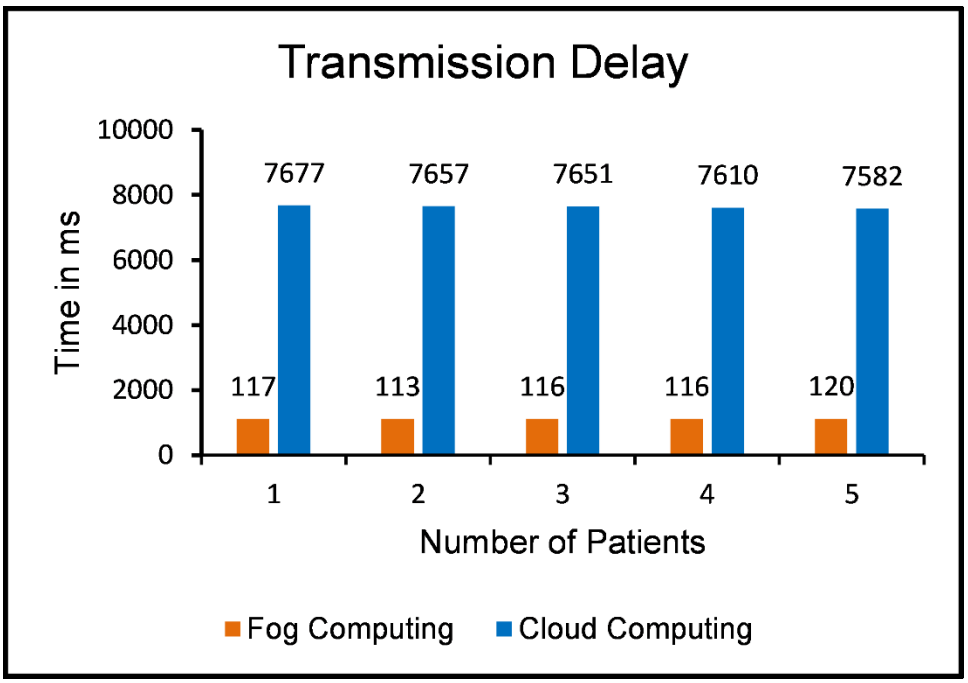

Figure 3.14 Transmission Delays of Fog and Cloud Computing

3.10.2 Computation Delay

Computing power depends on the device capability in terms of its hardware configuration. It depends on cache memory, processor, operating frequency, scheduling algorithm, communication bus, memory and number of cores. The current Fog device is configured to read and analyse more real-time ECG waves simultaneously on its different ports. By varying the number of patients its average computational delay in Fog Node is measured and shown in figure 3.15.

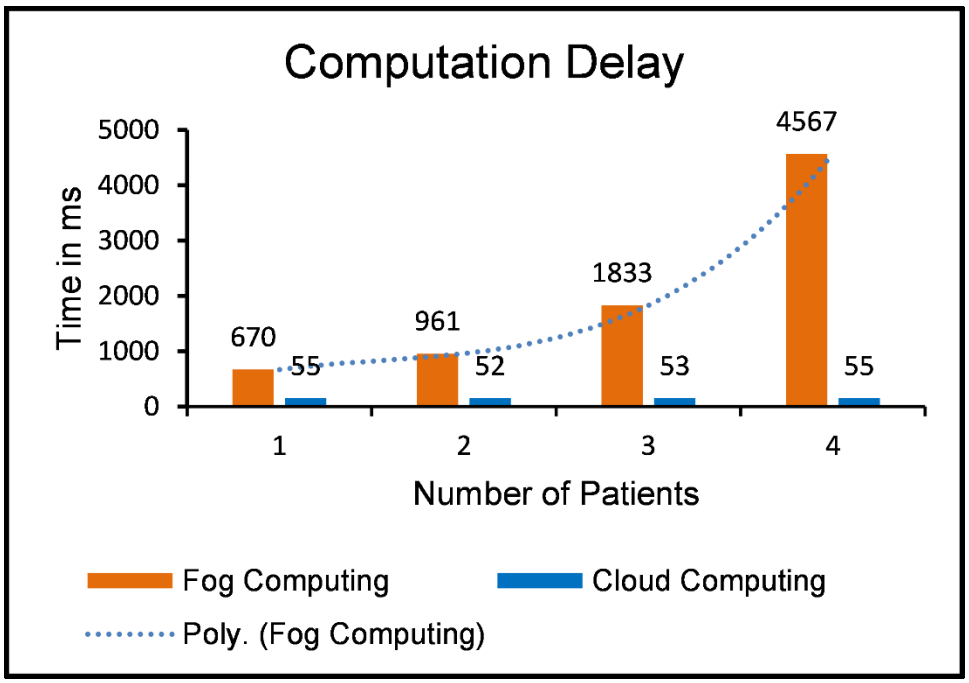

Figure 3.15 Computation delay of Fog and Cloud Computing

The computation delay is almost identical in terms of Cloud Computing but it varies a lot in the case of Fog Computing. Fog Computing shows the polynomial growth of the order of 3˚. The computational delay also depends on how the system is made i.e. the GUI computation, background computation, the refresh rate and the number of parallel tasks, etc. For four patients the computational delays are 4523, 4662, 4487 and 4593 ms respectively, which comes out as 4567 as an average value shown in figure 3.15. So giving more load on the Fog system is not suitable for the time-sensitive decision-making systems.

3.10.3 Response Time

The Response time is the overall performance time of the system. It shows the time difference between the generation of the ECG signal and the generation of response or decision in terms of normality and abnormality. Here, the average response time is found by varying the number of patients as shown in figure 3.16. In the proposed system, the Fog Node serves early responses than the cloud architecture, but only when the number of users is lesser than four. It responds very early if the number of users is less. While in case of Cloud Computing the Response time is almost equal.

If the number of users is five or more than that then the proposed Fog Computing architecture will underperform than the cloud.

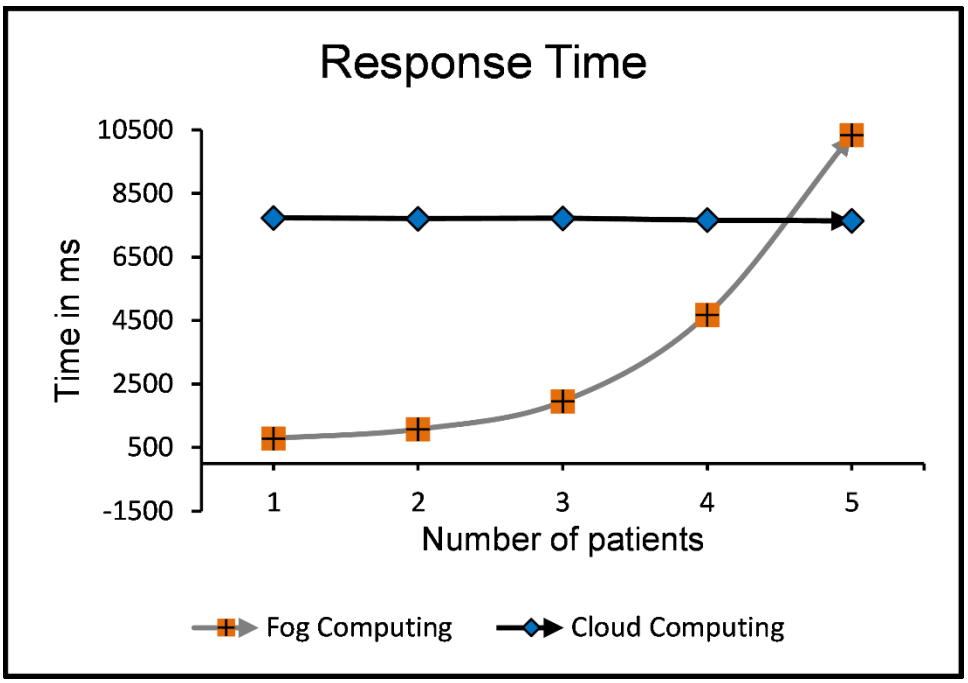

Figure 3.16 Response time of Fog and Cloud Computing

Hence it is not suggested for more number of patients in real-time health care analytics.

3.10.4 Standard Deviation

Standard deviation shows the spread of numbers from the mean or the expected value. When standard deviation is calculated for the different factors like transmission time and computation time for the Fog and Cloud Computing, the Fog processing time shows maximum standard deviation. This deviation is due to varying number of patients in the Fog Node. It is shown in figure 3.17.

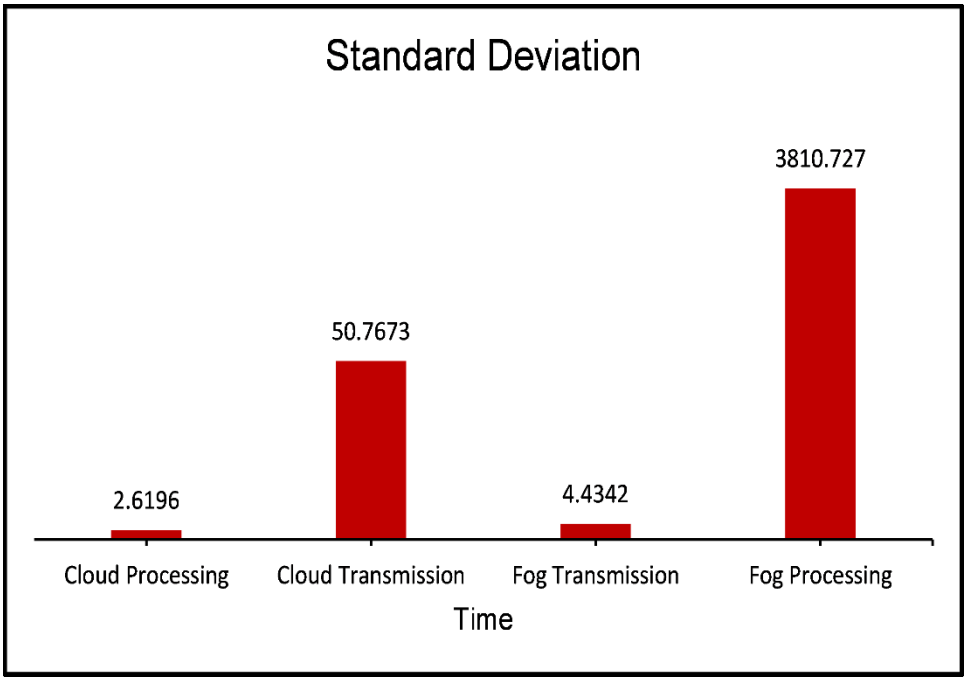

Figure 3.17 Standard Deviation of different Parameters

Concluding Remarks

This work introduced a cloud-assisted smart fog gateway for delay-sensitive IoT-driven healthcare applications. It ensures the tolerable delay while providing the service to the healthcare applications, by applying the smart partitioning and allocation using a decision tree. The decision rules are based on the application context and resource availability in fog and cloud infrastructure. In the smart fog gateway, the proposed approach intelligently takes the decision to determine the corresponding data stream based on the application context. The proposed approach provides the service to the end-user promptly. Cloud Computing Based IoT architecture is delay-sensitive for Critical Health Care applications. So, the LAN based Fog Computing Processing approach can be used to reduce the delay. Also, this technique helps to reduce the data burden on the Cloud. Moreover, Fog Computing should have memory, processing and computation capabilities and Fog Nodes can be placed in either LAN or as a Gateway. Since Raspberry Pi has networking, memory, storage and computation abilities, it becomes a suitable option to use as a Fog Node. We also discuss different Raspberry Pi based Fog installations. Furthermore, fog based health care systems are better than the Cloud-based health care system in terms of network bandwidth and response time, but it lags behind in computation power. The overall response by fog to

find any abnormality in the ECG signal is given way before the Cloud does - which is very vital in health care scenarios to save patient's lives. The Windowing algorithm is suitable to classify ECG signal in real time. Fog Computing gives a better response when the number of patients is less and it is observed that the Fog Computing response time is directly proportional to the number of patients. After a particular threshold value of the number of patients, Fog Computing will not perform better than Cloud Computing. The transmission delay and the computation delay plays a major role in the Fog Computing domain.