Reducing Delay and Packets Loss in IoT-Cloud Based ECG Monitoring by Gaussian Modeling

https://doi.org/10.3991/ijoe.v19i06.38581

Ahmed A. Fadhel(✉), Hamid M. Hasan
Control and Systems Engineering Department, University of Technology, Baghdad, Iraq
11418@uotechnology.edu.iq

Abstract—Health monitoring based on the internet of things (IoT) and cloud computing is regarded as a hot topic to research. However, such systems often face issues with delay and throughput due to the large amount of data that must be transmitted from sensors to the cloud. One important type of data for health monitoring is Electrocardiogram (ECG) signals, which generate a large amount of data to be transmitted. This research treats this problem by modelling these signals in order to reduce their size using Gaussian approximation. The cloud server is an MQTT broker to which the sensors publish their data via a gateway. The Gaussian parameters are calculated in the gateway, which act as a Fog layer, before published to the broker. The monitoring devices can subscribe to the broker and access the transmitted data. Our experiments were conducted using the MIT–BIH dataset and a real broker (HiveMQ). The results showed that the system was able to significantly reduce delay in transmitting data and prevent loss of information. Without using the Gaussian approximation technique, the system was only able to monitor a limited number of patients (17 for Qos1 and 23 for Qos0) without losing information. However, when using the Gaussian approximation model with five functions, the system was able to monitor many more patients (78 for Qos1 and 100 or more for Qos0) without losing any data.

Keywords—ECG, IoT, real-time remote monitoring, healthcare system, MQTT, HiveMQ, Gaussian modelling, fog computing

1 Introduction

In recent years, there has been a growing interest in the development and implementation of remote internet-based ECG monitoring systems. There has been a significant amount of research on the development and implementation of IoT-based ECG monitoring systems.

One of the main motivations for using IoT-based ECG monitoring systems is to enable remote monitoring of patients [1], particularly those with chronic conditions or who are at risk of cardiovascular events. This can reduce the need for hospital visits and improve patient outcomes by enabling timely interventions [2]. Even though it is now quite advanced, the medical field is still unable to assist everyone. There are two causes for this, one of which is that not everyone can afford costly medical care and the lack of universal access to such cutting-edge medical care is the other factor. Accessibility, affordability, and efficiency might all be increased with the use of IoT in the medical sector [3],[4].

There are numerous databases that can use to assess the approaches suggested in studies that focus on the processing of ECG signals. (MIT-BIH) Arrhythmia database is widely used for ECG signal analysis. It contains 48 half-hour patients of ECG recordings [5],[6].

Being represented by a vast volume of data and causing significant network congestion, the real-time transmission of the ECG signal over the Internet is a challenging task. The ECG signal can be compressed in a variety of ways, one of them is representing it using a number of Gaussian functions. Palash et al. [7] demonstrate that using a Gaussian model to model an ECG signal is more memory-efficient than using a different model. Abdul Awal et al. [8] create a simplified ECG model using the fewest possible parameters that may realistically simulate the ECG shape in various cardiac dysrhythmias, they suggest using 7 Gaussians functions to model an ECG beat. While Mohammad et al. [9] study showed that 16 Gaussian functions used to represent the ECG beat.

The use of IoT to remotely monitor the ECG signals has been the subject of numerous research studies. Parmveer et al. [10] presented the implementation of MQTT as a protocol for transferring ECG between the Raspberry Pi and Bluemix Cloud server. Using a Raspberry Pi, Ashwini et al. [11], upload the ECG measurements obtained from sensors at the Raspberry Pi and stored on the IoT server using the HTTP protocol in accordance with patient IDs. This information will be analysed by the doctor, who will then make an assessment and carry out health monitoring as a result. HTTP is not well suited for use in the IoT because it is based on a request-response model.

Stefano et al. [12], The ZigBee protocol was utilized to transmit data from a sensor. According to this investigation, 6 patients only could get an ECG after it had been compressed and sent without an ACK with a 5% data loss. Multi-Patient ECG's Monitoring based on ZigBee and a web server for data storage and web applications is presented by Ryan et al. in [13]. According to this study, the system can operate without issue with less than 20 users. Ayaskanta et al. [4] show that the digital ECG sensor data for one subject is received by the Raspberry Pi from the ADC through Wi-Fi and published to be stored in the Cloud MQTT broker using MQTT Mosquitto client.

While Satija et al. [14] demonstrates how transmission of acceptable ECG signals only through signal quality assessment (SQA), which can dramatically lowering the strain on cloud servers, bandwidth, and maintenance expenses.

Alvee et al. [15] and Tamanna et al. [16] are used a Raspberry Pi as a fog to analyse data from ECG sensors sent over an Arduino to determine whether the patient state was acting abnormally (an alert is sent in the case of a critical situation) and send an SMS to the doctor or nurse. the patient's temperature and heart rate are then routinely sent to a web cloud server where they are stored as a historical record. The predetermined people can access the website and use the pi camera to keep track of the patient's health.

Hoe et al. [17] presented two modes: the store-and-forward mode, which does not have issues with data integrity and does not require a strong network, and the real-time mode, which is more complex because data integrity can be affected by transmission latency and packet loss during real-time transmission, particularly for ECG signal transmission. The use of inaccurate health data can result in misdiagnosis.

Manju et al. [18] and Zhe et al. [19] are suggested developing a real-time ECG monitoring system using cloud computing. The ECG data from the sensor is sent to a web service through a mobile gateway. The communication protocol used between the controller device and the mobile gateway is Bluetooth. As a result, the HTTP server is able to accept the client's request and provide the necessary response. The HTTP server only provides a GUI for the ECG signal. Additionally, the smartphone app, which acts as a mobile gateway, uses the MQTT protocol to transmit the ECG signal from the ECG module to the app. James et al. [20] study state that, an ECG signal was obtained using an AD 8232 sensor, and an Arduino MKR1010 microcontroller was used to transmit the signal over Bluetooth to a matlab application, which displayed the processed ECG signal, while beat per minute (BPM) uploaded to a thinkspeak web server. Amjad et al. [21] proposed that The ECG patient signal is communicated to the MYSQL database on the IoT cloud stored as a historical record, using the HTTP protocol. The patient's doctor is able to access the web server to get the heart rate (HR) in order to decide whether a catastrophic scenario has occurred and what action to take.

The majority of these studies did not consider the impact of network issues like congestion and delay on the real-time monitoring of ECG signals. Furthermore, they did not examine the amount of data that gets lost when monitoring a higher number of patients. This data loss is concerning when monitoring a large patient population.

The propose system offers a solution to the issue of packets loss and latency in remotely monitoring ECG signals in real time for a large number of patients by lowering the size of the ECG beats utilizes a Gaussian function to model the signal [8]. The characteristic waves of an ECG beat, such as P, Q, R, S, and T, often have shapes that closely resemble a Gaussian distribution [9]. This, along with the mathematical properties and usefulness for statistical analysis, makes Gaussian a desirable choice for modeling ECG signals. The method fits a Gaussian function to the ECG signal data using Scipy's curve fit function [22]. As a result, a set of optimized parameters are obtained for each ECG beat, which can be used to reconstruct the ECG signal. The proposed system uses a Raspberry Pi as a fog computing layer [23],[24] to implement the modelling. The information is then sent to the online cloud via HiveMQ, a message broker that enables communication between devices by utilizing the MQTT publish-subscribe protocol. Monitoring devices have access to the transmitted data and can subscribe to the broker, which enables them to reconstruct the ECG signal beats.

2 Material and methods

2.1 System design

The proposed system consists of sensor, transmitter module, controller and a broker server as shown in Figure 1. After the AD8232 sensor senses ECG data, the ESP 32 module transmits the raw data to the Raspberry Pi 4 (8G Ram) controller through Wi-Fi [25],[26],[27] acting as a fog layer for all pre-processing computations [24] including denoising, normalization, feature extraction, obtaining heartbeats and Gaussian modelling, this information is sent to the broker (HiveMQ), which is a message broker that facilitates the communication between devices.

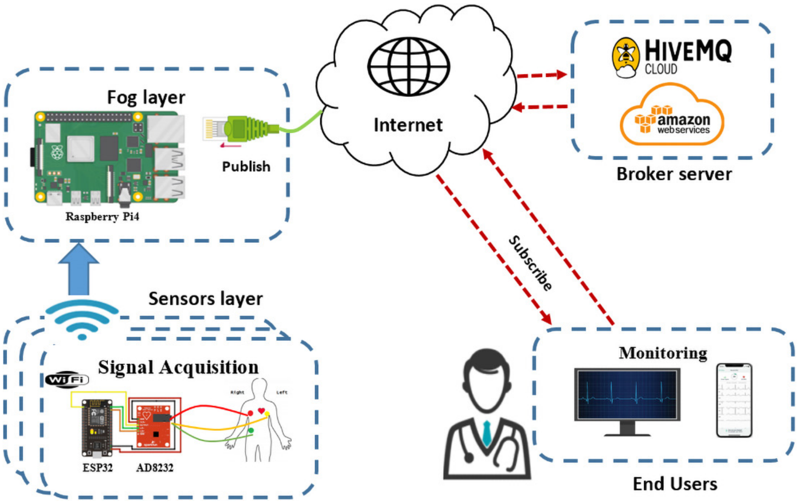

Fig. 1. Frame work architecture of the proposed real-time multi-patient monitoring system

Paho MQTT is an open-source library that enables MQTT communication, allowing connection to MQTT brokers to send or receive messages. It supports secure communication and is efficient with message queuing and automatic reconnections. This making it suitable for low-power and unreliable networks. MQTT is a lightweight publish-subscribe messaging protocol that is commonly used in IoT applications [25],[28]. The Raspberry device was linked to the internet using Ethernet over fiber optic technology, instead of WiFi, in order to connect to the broker server (HiveMQ). This is because fiber optic offers benefits, such as high-speed connectivity and immunity to electromagnetic interference as shown in Figure 1.

2.2 Data collection

The MIT-BIH arrhythmia database has been used in this study to test its performance and effectiveness as an open-source ECG database. It includes 48 Holter recordings from 48 people, each with a duration of 30 minutes. Of these, 25 were chosen to include clinically severe arrhythmias and 23 were healthy. Annotations for numerous rhythmic and morphological arrhythmias can be found in this database. The primary purpose of it is to identify abnormalities in individual heartbeats [29]. The data were extracted from this database made up of Normal Sinus Rhythm beats (NB), Left Bundle Branch Block (LBBB) beats, Atrial Premature Contraction (APC) beats, Right Bundle Branch Block (RBBB) beats and Premature Ventricular Contraction (PVC) beats.

2.3 Modelling ECG signal

Gaussian fitting is a useful tool for modelling ECG signals and has several benefits, including reducing the size of the ECG signal. By compressing the ECG signal, the issue of bandwidth and network congestion is reduced, and the storage issue is also addressed. The optimized parameters for each ECG beat can be used to reconstruct the

ECG signal, allowing for a significant reduction in the size of the ECG data. This compression of ECG signals makes it easier to transmit, store, and analyze large amounts of ECG data, making it an essential tool in medical applications.

In This study, the ECG signal was modelled using summation of five Gaussian functions, which is given by the Equation (1).

$$F(x) = \sum_{n=1}^{5} a e^{\frac{-(x-\mu)^2}{2\sigma^2}} \tag{1}$$

The initial guesses for the height (a), position (μ) and standard deviation (σ) of each Gaussian function are required to be provided in order to perform the modeling using the Scipy's curve fit function [22], and it returns the model's optimal parameters (OP) for the supplied data (ECG Beats). Every single beat gives (15) optimal parameters, the ECG beat can be reconstructed from these model parameters. Also this can remove the noise without any extra work. The proposed algorithm was applied to model ECG beats for the five classes and the overall Root Mean Square error value RMS for each class of ECG beats was calculated as listed in Table 1.

Table 1. RMS per ECG class type

Class Type	**NB**	**LBBB**	**RBBB**	**APB**	**PVC**
RMS	0.017	0.007	0.031	0.076	0.018

The results in Table 1 indicated that the algorithm accurately modeled the ECG signal, with low RMS error values across all classes. ECG classes with their model fitting by five Gaussian function are illustrate in the Figure 2.

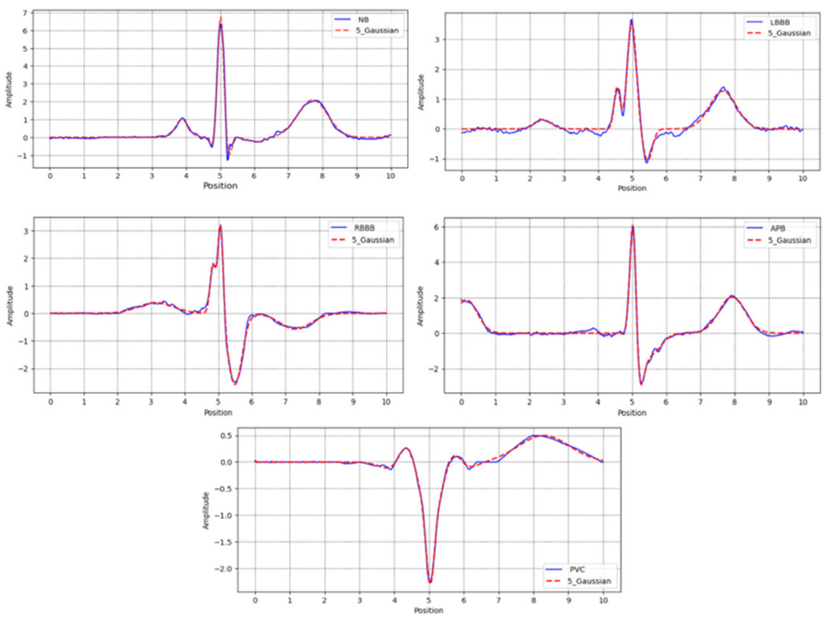

Fig. 2. Model fitting of different ECG classes

2.4 MQTT and HiveMQ broker

MQTT (Message Queuing Telemetry Transport) is a lightweight, publish-subscribe messaging protocol which allows devices to send and receive messages asynchronously that is often used in IoT applications [30],[31],[32] with different levels of quality of service (QoS) to optimize for different types of communication which allow users to specify the level of reliability they require for each message includes three levels of Quality of Service which determine the level of reliability in the delivery of messages. QoS0 ensures that a message is delivered at least once, but may not guarantee delivery to the intended recipient. QoS1 ensures that a message is delivered at least once and attempts to guarantee delivery to the intended recipient. QoS2 ensures that a message is delivered exactly once and guarantees delivery to the intended recipient. MQTT is a messaging platform that offers reliability and security for IoT applications. It supports the SSL/TLS (Secure Sockets Layer/Transport Layer Security) protocol, which establishes a secure communication channel between devices and brokers. SSL/TLS provides encryption and authentication mechanisms to safeguard data transmitted over the internet. SSL/TLS authentication verifies the legitimacy of devices accessing the data, ensuring that only authorized entities can do so [33]. It also supports authentication, so that only authorized devices can connect to the network. It is well-suited for IoT applications because it is designed to be efficient, low-bandwidth, and low-latency, which are important characteristics for devices with limited computing resources and connectivity. In contrast, HTTP (Hypertext Transfer Protocol) is not well suited for use in the IoT because it is based on a request-response model, which means that the client must initiate a request and the server must respond to it, that may not be practical for resource-constrained devices [31],[33]. The HiveMQ is a message broker that facilitates the communication between devices. It is providing a central location (Cloud) for devices to publish and subscribe to messages. When a device wants to publish a message, it sends the message to the broker, which then distributes the message to all devices that were subscribed to that topic. In this way, HiveMQ enables devices to communicate with each other in a decoupled and asynchronous manner. HiveMQ also provides various features that help ensure the reliability and security of MQTT communication. For example, it allows users to define access control policies that restrict which devices can publish and subscribe to which topics [34].

2.5 System implementation

The entire hands-on implementation was carried out on a public network with a bandwidth of 2 Mbps. The ECG signal used in this system is limited to a one-minute duration, as BPM (beats per minute) is the standard method of measuring heart rate [8],[35],[36]. The typical resting heart rate for adults falls within the range of 60 to 100 BPM [36]. To ensure consistency and accuracy, the proposed system adopted 85 BPM as the number of beats sent per minute for each patient. The design and construction of the hardware and software components are necessary to implement an IoT-based monitoring system. The hardware components were implemented as indicated in Figure 1, which describes the framework architecture of the proposed system. In terms of software, the operations were conducted using the Python programming language and the scikit-learn library, which is a computational tool in Python used for modeling the ECG signals. Paho MQTT software was used to connect to MQTT brokers and publish or subscribe to topics, sending or receiving messages as shown in Figure 5. To publish the MQTT payload messages, a loop was used to iterate over a range of patient numbers and use the publish method to send each message, which takes three arguments: the topic (patient ID), the message payload (85 beats), and the QoS type. On the other hand, to receive the data, the subscribe method used as in the python command below:

Publish command: client.publish(topic, payload, QoS)
Subscribe command: client.subscribe (topic, QoS)

In order to effectively evaluate the proposed system, we conducted a series of experiments in actual network settings. The dataset was provided to the Raspberry Pi (fog layer) via an SD card, and the Raspberry Pi was connected to the HiveMQ broker through the MQTT Protocol, along with a host ID and port number that corresponded to a specific account. Each patient gets 85 BPM, and converted to JSON format and bind to the current time in seconds to obtain the payload message. The QoS level was set to specific level to initiate the publishing process. The first experiment was conducted with a single patient, and the number of patients was then incrementally increased by 10 until the final experiment, which included 100 patients. The delay time and packet loss for each experiment were recorded. Two scenarios were performed:

Monitoring ECG signal before modelling. The Raspberry Pi sending the original ECG signal to a MQTT broker. In this case, the message payload that was published for each patient to the HiveMQ server contained 85 ECG beats, with 360 samples for each beat as in Figure 3. Using any device, it was possible to subscribe to each patient's topic and monitor the incoming ECG signals. The publishing signals without modelling from the Raspberry to the broker and the receiving messages in the subscription's side shown in Figure 4.

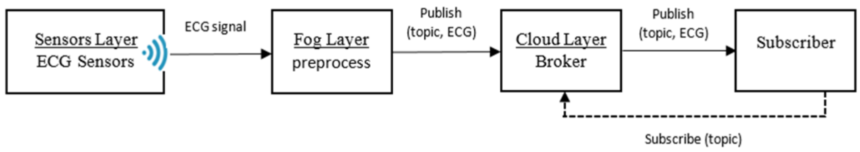

Fig. 3. Work-flow of the proposed system in the first scenario

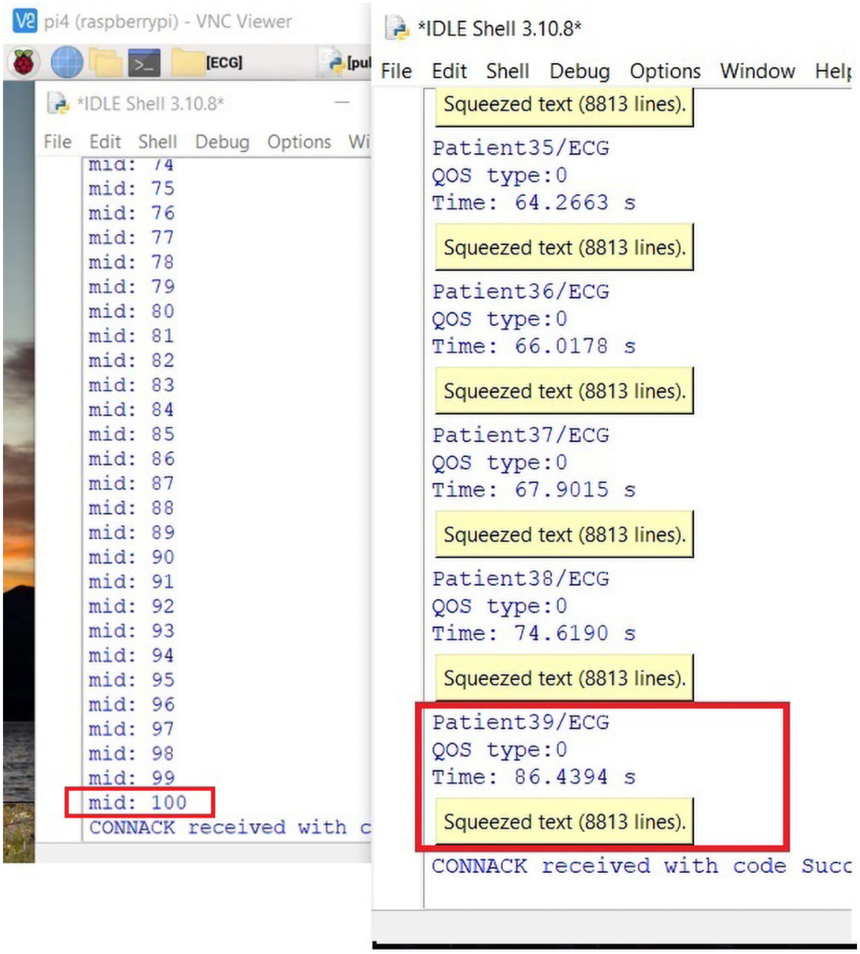

Fig. 4. Publish-subscribe with the first scenario

Monitoring ECG signal after modelling. The Raspberry Pi sending the optimal parameters (OP) obtained from the modelled ECG to the broker server. In this case, the message payload that was published for each patient to the HiveMQ server contained 85 ECG beats, with 16 samples for each beat (15 + class type) as in Figure 5. The publishing of the modelled signal to the broker and the receiving messages in the subscription's side shown in Figure 6. The monitoring devices can subscribe to each patient's topic and monitor the ECG signal after reconstructed.

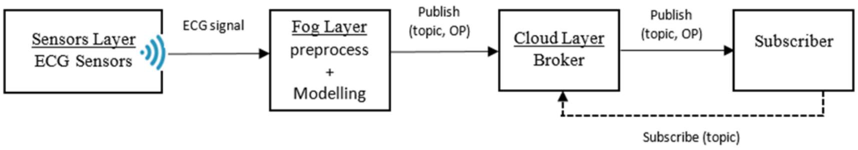

Fig. 5. Work-flow of the proposed system in the second scenario

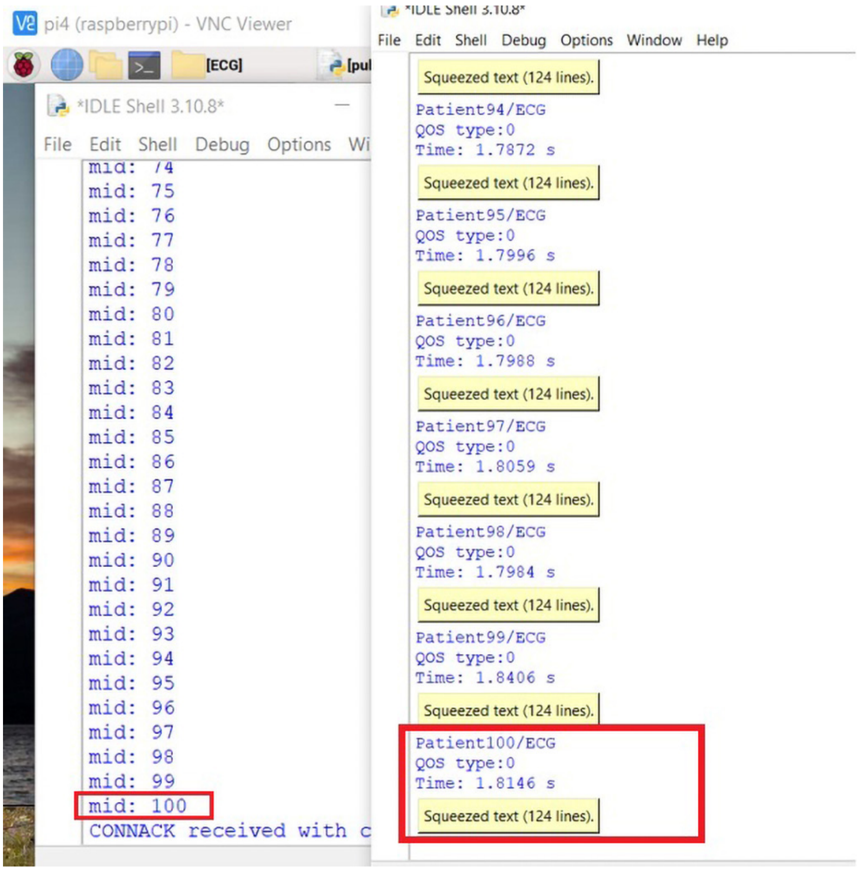

Fig. 6. Publish-subscribe in the second scenario

3 Results and discussion

The proposed system was implemented and initiated the transmission of ECG signals to the cloud broker without prior modeling and the QoS was set to 0. As the number of patients increased, the network rapidly became congested, leading to packet loss and significant time delays. The experimental results, presented in Table 2, illustrate the impact of the congestion on the system's performance. The "No. of Topics sent" column represented the number of patients whose data were published, while the "No. of Topics received in 1min" column represented the number of patients whose data were received on the subscription side within one minute. The "loss" column represented the percentage of data that was received after the minute ended, and the "Topic loss" column represented the number of patients' data that were lost in the network congestion. The experimental results recorded in The Table 2 indicate that only 23 patients' signals were successfully received within the first minute of initiating publishing, with the remaining patient signals being received after the minute had ended. As the number of patients increased, the situation worsened, with a large percentage of patient signals being lost when the number of patients exceeded 40. The practical experiment depicted in Figure 4 demonstrates that out of the 100 published patients' signals, only 39 patients' signals able to receive. Of those, only 23 received their data within the first minute, and the congestion on the network caused the loss of 61 patients' signals data, as shown in Table 2.

Table 2. The performance without modelling (QoS=0)

No. of Topic Sent	No. of Topic Received in 1 min	Loss	Delay Time	Topic Loss
1	1	0%	2.1407 s	0
10	10	0%	24.2055 s	0
20	20	0%	53.0661 s	0
30	23	23%	53.3058 s	0
40	23	43%	53.7891 s	16
50	23	54%	54.3098 s	21
60	23	61%	54.3211 s	26
70	23	67%	55.9878 s	29
80	23	71%	55.9908 s	38
90	23	74%	56.6540 s	51
100	23	77%	56.7865 s	61

When attempted to improve the situation by setting QoS to 1. While this did result in eliminated the topic loss, the time delay increased. Only 17 patients signal data were successfully received within the first minute of initiating publishing, with the remaining patients' signals data being received after the minute had ended as shown in the results in Table 3.

Table 3. The performance without modelling (QoS=1)

No. of Topic Sent	No. of Topic Received in 1 min	Loss	Delay Time	Topic Loss
1	1	0%	2.1407 s	0
10	10	0%	44.2055 s	0
20	17	15%	53.0661 s	0
30	17	43%	53.3058 s	0
40	17	57%	53.7891 s	0
50	17	66%	54.3098 s	0
60	17	72%	54.3211 s	0
70	17	76%	55.9878 s	0
80	17	79%	55.9908 s	0
90	17	82%	56.6540 s	0
100	17	83%	56.7865 s	0

In comparison to the first scenario, the second scenario resulted in completely different outcomes due to the implementation of a Gaussian function to model and reduce the signal data volume by over 90%. The time needed to complete the modelling process for 85 beats was 1.7 s, in addition to the time required to receive and reconstruct the ECG signal. When the QoS was set to 0, the absence of an ACK returns, greatly contributed to eliminating congestion on the network and allowing the system to receive signals parameters from all patients within a short period of time, averaging at 1.9 s with no topic loss, as shown in the Table 4. The practical experiment depicted in Figure 6 demonstrates the successful receipt of parameters for all 100 patients on the subscription side who were published, without experiencing any loss.

Table 4. The performance with modelling (QoS=0)

No. of Topic sent	No. of Topic Received in 1 min	Loss	Delay Time	Topic Loss
1	1	0%	1.7378 s	0
10	10	0%	1.8322 s	0
20	20	0%	1.8180 s	0
30	30	0%	1.7445 s	0
40	40	0%	1.7898 s	0
50	50	0%	1.9178 s	0
60	60	0%	1.8530 s	0
70	70	0%	1.9278 s	0
80	80	0%	1.7481 s	0
90	90	0%	1.7478 s	0
100	100	0%	1.7778 s	0

When the QoS setting was changed to 1, the system stopped to receiving signals after receiving 20 patients signal parameters in order to return the ACK to the sender, before resuming normal receiving. This resulted in only 78 patients signal parameters being received within a minute of the publishing time when the number of patients exceeded 79, as shown in Table 5.

Table 5. The performance with modelling (QoS=1)

No. of Topic Sent	No. of Topic Received in 1 min	Loss	Delay Time	Topic Loss
1	1	0%	1.7682 s	0
10	10	0%	1.7618 s	0
20	20	0%	1.8180 s	0
30	30	0%	16.3381 s	0
40	40	0%	23.5347 s	0
50	50	0%	37.6547 s	0
60	60	0%	45.5227 s	0
70	70	0%	52.5389 s	0
80	78	2.5%	57.5332 s	0
90	78	13%	58.7654 s	0
100	78	22%	59.5347 s	0

In addition, it is also worth noting that the delay in transmitting the signal on the public network is ten times greater than on the private network [17]. This latency can be attributed to network congestion and the number of hops the signal must traverse before reaching its destination. Therefore, it is recommended that remote monitoring operations for ECG signals be implemented on private networks to avoid these issues.

The Figure 7 below demonstrates how the transmission latency increases as the number of patients increases. It compares two scenarios, one where the signal is not modelled (labelled as "N") and one where it is modelled using a Gaussian function (labelled as "G"). The monitoring devices can subscribe to each patient's topic and monitor the ECG signal after reconstructed as the example shown in the Figure 8. The Figure 9 also illustrates the proportion of loss of signal as a function of the growth in patients' number. The idea behind it is that as number of patients in a network increases the delay in transmitting a signal and lost also increases. If a Gaussian function is used to model the signal, it helps eliminate the bottleneck in the network and reduce the transmission latency, resulting in less signal loss.

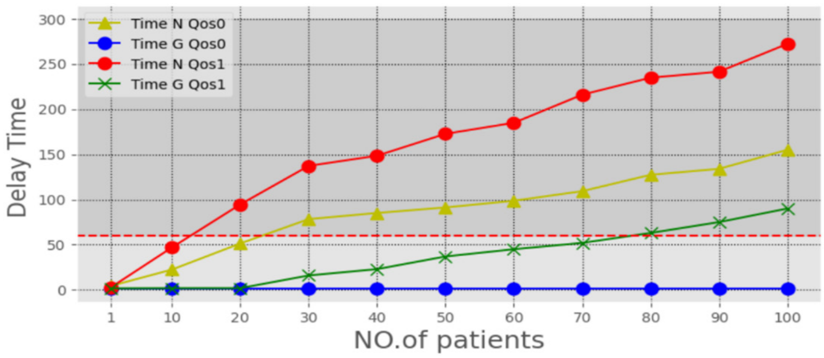

Fig. 7. Time delay as a function of the no. of patients

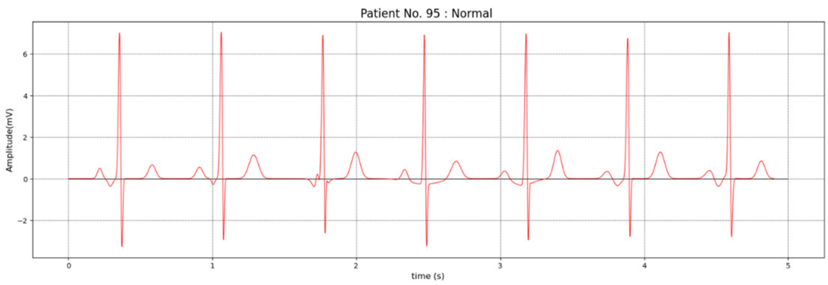

Fig. 8. The received ECG signal after reconstructed from the OP

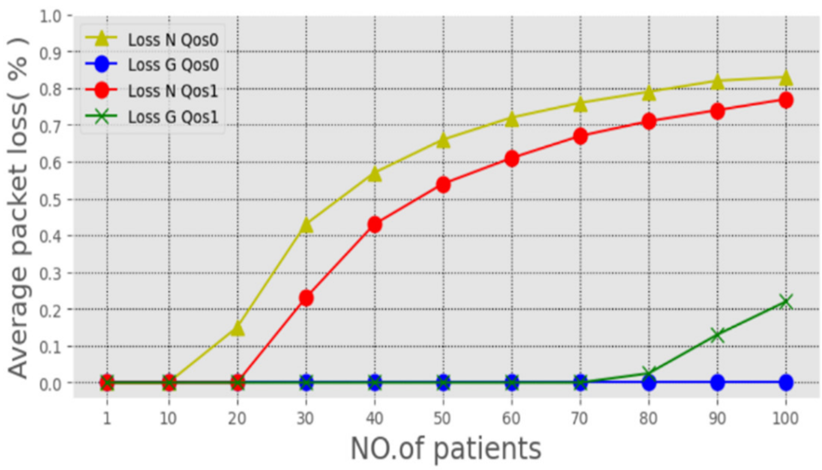

Fig. 9. Topic loss as a function of the no. of patients

4 Conclusion

This study proposes a monitoring system for electrocardiogram (ECG) signals using the Internet of Things (IoT) and cloud technology. The system uses a five Gaussian function to model ECG signal beats at the fog layer, which is a layer of computing and data storage between the sensor layer and the cloud.

To eliminate network bottlenecks and decrease delays, the beats were transformed from 360 samples per beat to 15 parameters represented by five Gaussian functions. This reduction in signal size, which is less than 5% of the original size, did not compromise the quality of the signal as indicated by the RMS values in Table 1. Additionally, noise has been removed to get a cleaner signal.

The actual ECG signals are collected by sensors, modelled at the fog layer using a Raspberry Pi4 device, after filtering and segmentation process, and then transmitted to the cloud layer using the MQTT protocol. The HiveMQ acts as the broker server. The system is able to publish one minute's worth of ECG signal data (85 beats) for each patient to their corresponding topic, so that monitoring clients can subscribe and access this data, just like the example shown in the Figure 8 above. The practical experiments demonstrate that while the system operates with QoS0, before modelling, the delay time for receiving data from 20 patients can be up to 53 seconds and the maximum number of records that can be received per minute is 23 when the number of patients is increased to 100, resulting in 77% of records being lost.

While, when the signal is modelled before publishing, the delay time remained stable at an average of 1.9 seconds (modelling time + transmission time) throughout the increase in patients' number from one to 100, as shown in Figure 7, with no losses.

When the QoS setting was changed to 1, before modeling, to stay in the real-time monitoring the maximum number of records that can be received per minute drops to 17 in 53 seconds, which resulted in 83% of records being received out of time when publishing for 100 patients. But, when the signal is modeled before publishing, the delay time remained stable at an average of 1.9 seconds (modeling time + transmission time) throughout the increase in patients' number from one to 20, without any losses. However, the subscribing client cannot receive any record until the ACKs returned to the fog layer, causing only 78 records can be received within a minute when the number of patients is more than 78, and the remaining records are received out of time, as shown in Figure 9.

Based on all of this information, we can conclude that the system operates smoothly and efficiently, without any data loss or network overload, when sending the ECG signal on QoS0, after it has been processed through the Gaussian function. Even though the QoS0 is a fire-and-forget protocol but it gains the reliability from the TCP protocol. That is because the MQTT is an application protocol runs in application layer over the TCP which runs at the transport layer of TCP/IP suit. Hence, we can confidently rely on this efficient and reliable system for sending ECG signals for remote monitoring.

5 References

[1] A. W. Al-Mutairi, K. M. Al-Aubidy, and F. N. Alhalaiqa, "IoT-Based Real-Time Monitoring System for Epidemic Diseases Patients: Design and Evaluation," *Int. J. online Biomed. Eng.*, vol. 17, no. 1, pp. 63–82, 2021, https://doi.org/10.3991/ijoe.v17i01.18849

[2] M. A. Serhani, H. T. El Kassabi, H. Ismail, and A. N. Navaz, "ECG Monitoring Systems: Review, Architecture, Processes, and Key Challenges," *Sensors (Switzerland)*, vol. 20, no. 6. 2020. https://doi.org/10.3390/s20061796

[3] M. M. Abdellatif and W. Mohamed, "Telemedicine: An IoT Based Remote Healthcare System," *Int. J. online Biomed. Eng.*, vol. 16, no. 6, pp. 72–81, 2020, https://doi.org/10.3991/ijoe.v16i06.13651

[4] A. Mishra, A. Kumari, P. Sajit, and P. Pandey, "Remote web based ECG Monitoring using MQTT Protocol for IoT in Healthcare," *Int. J. Appl. Eng. Res.*, vol. 5, pp. 1096–1101, 2018, https://doi.org/10.21090/IJAERD.71092

[5] "MIT-BIH Arrhythmia Database v1.0.0." https://physionet.org/content/mitdb/1.0.0/ (accessed Jan. 10, 2023).

[6] S. Kaplan Berkaya, A. K. Uysal, E. Sora Gunal, S. Ergin, S. Gunal, and M. B. Gulmezoglu, "A survey on ECG analysis," *Biomedical Signal Processing and Control*, vol. 43. Elsevier Ltd, pp. 216–235, 2018. https://doi.org/10.1016/j.bspc.2018.03.003

[7] P. Kundu and R. Gupta, "Electrocardiogram Synthesis Using Gaussian and Fourier Models," in *2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, pp. 312–317, 2015. https://doi.org/10.1109/ICRCICN.2015.7434256

[8] M. A. Awal *et al.*, "Design and Optimization of ECG Modeling for Generating Different Cardiac Dysrhythmias," *Sensors*, vol. 21, no. 5, pp. 1–27, 2021, https://doi.org/10.3390/s21051638

[9] M. S. Billah, T. B. Mahmud, F. S. Snigdha, and M. A. Arafat, "A Novel Method to Model ECG Beats Using Gaussian Functions," in *2011 4th International Conference on Biomedical Engineering and Informatics (BMEI)*, vol. 2, pp. 612–616, 2011. https://doi.org/10.1109/BMEI.2011.6098409

[10] P. Singh and A. Jasuja, "IoT Based Low-Cost Distant Patient ECG Monitoring System," *Proceeding – IEEE Int. Conf. Comput. Commun. Autom. ICCCA 2017*, vol. 2017-Janua, pp. 1330–1334, 2017, https://doi.org/10.1109/CCAA.2017.8230003

[11] A. Gutte and R. Vadali, "IoT Based Health Monitoring System Using Raspberry Pi," in *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pp. 1–5, 2018. https://doi.org/10.1109/ICCUBEA.2018.8697681

[12] E. Spanò, S. Di Pascoli, and G. Iannaccone, "Low-Power Wearable ECG Monitoring System for Multiple-Patient Remote Monitoring," *IEEE Sens. J.*, vol. 16, no. 13, pp. 5452–5462, 2016, https://doi.org/10.1109/JSEN.2016.2564995

[13] M. R. F. Nurdin, S. Hadiyoso, and A. Rizal, "A low-cost Internet of Things (IoT) system for multi-patient ECG's monitoring," in *2016 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, pp. 7–11, 2016. https://doi.org/10.1109/ICCEREC.2016.7814958

[14] U. Satija, B. Ramkumar, and S. M. Manikandan, "Real-Time Signal Quality-Aware ECG Telemetry System for IoT-Based Health Care Monitoring," *IEEE Internet Things J.*, vol. 4, no. 3, pp. 815–823, 2017, https://doi.org/10.1109/JIOT.2017.2670022

[15] A. Rahman, T. Rahman, N. H. Ghani, S. Hossain, and J. Uddin, "IoT Based patient monitoring system using ECG sensor," *1st Int. Conf. Robot. Electr. Signal Process. Tech. ICREST 2019*, pp. 378–382, 2019, https://doi.org/10.1109/ICREST.2019.8644065

[16] T. Shaown, I. Hasan, M. M. R. Mim, and M. S. Hossain, "IoT-based Portable ECG Monitoring System for Smart Healthcare," *1st Int. Conf. Adv. Sci. Eng. Robot. Technol. 2019, ICASERT 2019*, pp. 1–5, 2019, https://doi.org/10.1109/ICASERT.2019.8934622
[17] H. T. Yew, M. F. Ng, S. Z. Ping, S. K. Chung, A. Chekima, and J. A. Dargham, "IoT Based Real-Time Remote Patient Monitoring System," *Proc. – 2020 16th IEEE Int. Colloq. Signal Process. its Appl. CSPA 2020*, pp. 176–179, 2020, https://doi.org/10.1109/CSPA48992.2020.9068699
[18] M. L. Sahu, M. Atulkar, M. K. Ahirwal, and A. Ahamad, "IoT-Enabled Cloud-Based Real-Time Remote ECG Monitoring System," *J. Med. Eng. Technol.*, vol. 45, no. 6, pp. 473–485, 2021, https://doi.org/10.1080/03091902.2021.1921870
[19] Z. Yang, Q. Zhou, L. Lei, K. Zheng, and W. Xiang, "An IoT-cloud Based Wearable ECG Monitoring System for Smart Healthcare," *J. Med. Syst.*, vol. 40, no. 12, 2016, https://doi.org/10.1007/s10916-016-0644-9
[20] J. Heaney, J. Buick, M. U. Hadi, and N. Soin, "Internet of Things-Based ECG and Vitals Healthcare Monitoring System," *Micromachines*, vol. 13, no. 12, p. 2153, 2022, https://doi.org/10.3390/mi13122153
[21] A. Qtaish and A. Al-Shrouf, "A Portable IoT-cloud ECG Monitoring System for Healthcare," *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 22, no. 1, 2022, [Online]. Available: https://doi.org/10.22937/IJCSNS.2022.22.1.37
[22] M. Gardegaront, D. Farlay, O. Peyruchaud, and H. Follet, "Automation of the Peak Fitting Method in Bone FTIR Microspectroscopy Spectrum Analysis: Human and Mice Bone Study," *J. Spectrosc.*, vol. 2018, 2018, https://doi.org/10.1155/2018/4131029
[23] A. M. Rahmani *et al.*, "Exploiting Smart e-Health Gateways at the Edge of Healthcare Internet-of-Things: A Fog Computing Approach," *Futur. Gener. Comput. Syst.*, vol. 78, pp. 641–658, 2018, https://doi.org/10.1016/j.future.2017.02.014
[24] H. Dubey *et al.*, "Fog Computing in Medical Internet-of-Things: Architecture, Implementation, and Applications," pp. 281–321, 2017, https://doi.org/10.1007/978-3-319-58280-1_11
[25] H. A. Mansour and H. N. Abdullah, "Design and Implementation of Remote Evaluation System for the Prosthetic Limbs," *2022 3rd Int. Conf. Emerg. Technol. INCET 2022*, vol. 40, pp. 60–65, 2022, https://doi.org/10.1109/INCET54531.2022.9824666
[26] F. I. Ali, T. E. Ali, and A. Z. Tarik, "Private Backend Server Software-Based Telehealthcare Tracking and Monitoring System," *Int. J. Online Biomed. Eng.*, vol. 19, no. 1, pp. 119–134, 2023, https://doi.org/10.3991/ijoe.v19i01.32433
[27] B. G. Mohammed and D. S. Hasan, "Smart Healthcare Monitoring System Using IoT," *Int. J. Interact. Mob. Technol.*, vol. 17, no. 1, pp. 141–152, 2023, https://doi.org/10.3991/ijim.v17i01.34675
[28] N. A-hussein and A. D. Salman, "IoT Monitoring System Based on MQTT Publisher/Subscriber Protocol," *Iraqi J. Comput. Commun. Control Syst. Eng.*, vol. 20, no. 3, pp. 75–83, 2020, https://doi.org/10.33103/uot.ijccce.20.3.7
[29] E. Merdjanovska and A. Rashkovska, "Comprehensive survey of computational ECG analysis: Databases, methods and applications," *Expert Syst. Appl.*, vol. 203, 2022, https://doi.org/10.1016/j.eswa.2022.117206
[30] F. Tsvetanov and M. Pandurski, "Selection of Protocols for Integration of Sensory Data Networks in Cloud Structures," *Int. J. online Biomed. Eng.*, vol. 18, no. 9, pp. 29–40, 2022, https://doi.org/10.3991/ijoe.v18i09.31321
[31] H. M. Hasan and S. A. Jawad, "IoT Protocols for Health Care Systems: A Comparative Study," *Int. J. Comput. Sci. Mob. Comput.*, vol. 7, no. 11, pp. 38–45, 2018, [Online]. Available: www.ijcsmc.com

[32] H. R. A. Ameer and H. M. Hasan, "Enhanced MQTT Protocol by Smart Gateway," *Iraqi J. Comput. Commun. Control Syst. Eng.*, pp. 53–67, 2020, https://doi.org/10.33103/uot.ijccce.20.1.6
[33] G. Xu, "IoT-Assisted ECG Monitoring Framework with Secure Data Transmission for Health Care Applications," *IEEE Access*, vol. 8, pp. 74586–74594, 2020, https://doi.org/10.1109/ACCESS.2020.2988059
[34] "HiveMQ – Enterprise ready MQTT to move your IoT data." https://www.hivemq.com/ (accessed Jan. 10, 2023).
[35] S. Abba and A. M. Garba, "An IoT-Based Smart Framework for a Human Heartbeat Rate Monitoring and Control System," p. 36, 2020, https://doi.org/10.3390/ecsa-6-06543
[36] "Heart rate: What's normal? – Mayo Clinic." https://www.mayoclinic.org/healthy-lifestyle/fitness/expert-answers/heart-rate/faq-20057979 (accessed Jan. 10, 2023).

6 Authors

Ahmed A. Fadhel received a B.Sc. in control and systems engineering, in 2010 from the University of Technology, Iraq. Now he is studying M.Sc. in computer engineering at the university of technology, Iraq. He is interested in Machine learning, Signal processing, Fog computing, Cloud computing. (email: 11418@uotechnology.edu.iq).

Hamid M. Hasan received a B.Sc. in control and systems engineering, and an M.Sc. in computer engineering from the university of technology, Iraq, in 1986 and 1990 respectively. He received Ph.D. in computer engineering, in 2012, from Basra University, Iraq. He is currently working as an Assistant professor at the faculty of control and systems engineering, university of technology, Iraq. He is interested in computer network studies and research, wireless sensor network realization, mobile robot navigation, IoT development and investigation, pattern recognition, face recognition, image processing, and stochastic DSP applications. (email: Hamid.M.Hasan@uotechnology.edu.iq).

Article submitted 2023-02-03. Resubmitted 2023-03-07. Final acceptance 2023-03-12. Final version published as submitted by the authors.