QATAR UNIVERSITY

COLLEGE OF ENGINEERING

ADVANCED MACHINE LEARNING TECHNIQUES FOR ARRHYTHMIA CLASSIFICATION

BY

ABDULLA MARAWAN ABOUMADI

A Thesis Submitted to

the Faculty of the College of Engineering

in Partial Fulfillment of the Requirements for the Degree of

Masters of Science in Computing

June 2022

COMMITTEE PAGE

The members of the Committee approve the Thesis of
Abdulla Marawan Aboumadi defended on 19/04/2022.

Elias Yaacoub
Thesis/Dissertation Supervisor

Hassan Noura
Committee Member

Uvais Qidway
Committee Member

Loay Ismail
Committee Member

Approved:

Khalid Kamal Naj , Dean, College of Engineering

ABSTRACT

ABOUMADI, ABDULLA, M., Masters : June : [2022], Masters of Science in Computing

Title: Advanced Machine Learning Techniques for Arrhythmia Classification

Supervisor of Thesis: Elias, E, Yaacoub.

With the development of Internet-of-Things (IoT) applications, the concept of smart healthcare applications has gradually emerged to be the main factor in medicine. In fact, this raises the need to have a secure system that is efficient at the same time, due to the limited resources of IoT devices. Many different techniques have been developed and studied recently. For example, with centralized learning (CL), all data are collected and processed in one place. But many of these models are heavy and lead to an infringement of patient's privacy. Hence, a Federated Learning (FL) approach helps in developing global application without storing the data in centralized cloud. Therefore, in this thesis, the concept of CL and FL using a convolutional neural (CNN) network is performed to identify and classify arrhythmia, while taking into consideration the accuracy and simplicity in simulating a system model that would be used in medical devices. The MIT-BIH dataset was used in this work to test and validate the proposed approach and compare it to other methods in the literature.

ACKNOWLEDGMENTS

This work was made possible by NPRP grant # 13S-0205-200270 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

With the rapid evolution of adding intelligence to embedded systems such as the Internet of Things (IoT) [1], and the presence of data that has been collected widely from different IoT environments, the path is paved for deploying new interactive applications that transfer the traditional industry service to intelligently active services. However, many challenges were raised due to these trends such as latency and reliability and the need for processing after transmitting big data generated by IoT while ensuring the same quality of service. Artificial intelligence brings such real-time interactive applications to reality with the help of active learning and deep learning. Indeed, an efficient intelligent healthcare application can take place in event detection, categorization, and online real-time monitoring for a patient that is highly exposed to health deterioration. Such application is useful in many scenarios such as the recent pandemic which causes health evaluations in online health care. An electrocardiogram (ECG) is a primary vital sign that allows a doctor to monitor the patient and can be updated online.

The essential target of the ECG is to monitor the heartbeat to identify the heart state; ECG is the only methodology that can identify the rhythm of the heart, especially for patients who suffer from chronic diseases. The ECG could be extracted from cardio tests or heart monitoring devices, and many extensive studies have been conducted in medical engineering to build early warning systems that notify patients and doctors of abnormal beats. Moreover, after cardiac operation, the chance of getting heart attack again increases and it does not mean that the patient fully recovers from heart disease. So, doctors need to keep monitoring patient's' heart activities in addition to medication; thus, using AI based application in discovering diseases in early stages would increase. Each year, many patients die because of neglecting home medical care, where the patient is actually tired of going frequently to hospitals for a regular check-up, and in many cases, they are waiting for their appointment but at the same time they are suffering from such diseases. It has been reported by American Heart Association [2], in 2020, that around 100 million people around the world suffer from Cardiovascular disease (CVD). CVD causes blocking blood vessels which leads to death, and the main cause of blood blocking is arrhythmia, especially if it has been detected late. Doctors have been using visual inspection of ECG signals to detect and locate arrhythmia. However, this technique can be time-consuming when cardiovascular diseases are at their early stage, and such constraints come behind the fact that the doctor needs to inspect a long record of the patient’s heartbeat. Therefore, Doctors use the heartbeat cardio test to detect arrhythmia, with the help of certain devices. Some are used for short duration monitoring such as Holter, which stays for one to two days, and others are used for long-term patients’ monitoring [3], but still this method is used for patients who already have heart diseases and it does not transfer data. Instead, patients need to wait until the recording finishes and then take the data back to hospital to be analysed. Moreover, the fact that doctors are human, can lead to the chance of the wrong inspection taking place due to error during the analysis [4]. Hence, using IoT as smart medical instrument will allow immediate data analysis, since the IoT will keep sending the data whenever it collects some. In addition, bringing AI to embedded systems such as IoT will limit the power consumption due to transmitting data, and collaboration between the IoT devices can build and train model that can detect and classify abnormal vital signs.

1.1 Centralized learning

There are mainly three types of active learning that could be used with big data to implement a real-time application, namely, centralized learning, decentralized learning, and federated learning. Researchers introduce decentralized learning with the trend of active learning and sharing data between the users, but this type of learning was not efficient since the model is fitted with the shared data, and usually only nearby IoT devices of one environment share data with each other. This methodology was missing a generalization factor, while Centralized Learning guarantees the model's generalization with the big data since all the training is done in the cloud (in one place). Moreover, centralized learning achieves high performance, which is why most of the existing research focuses on centralized learning with Deep Neural Networks (DNN)[5]. However, such application on IoT devices does not meet the requirement, since the power consumption is high due to the computational power needed to run DNN. Moreover, the data processing is done in a centralized paradigm putting users' privacy at high risk. Hence, to keep the user's data private it is not a practical solution to forward it to a centralized entity for training or in some cases for prediction [5]. Moreover, the communication path between the IoT and the cloud will be highly loaded with data. So bringing the intelligence to the IoT device will maintain data privacy in addition to efficient communication between the IoT and the cloud [6].

1.2 Federated learning

The new Federated Learning is a promising solution for model training without storing the data at a centralized location. While it is known as distributed machine learning, hence, it allows users to collaborate in building one model with their combined data, this is done without any sharing of data to a centralized entity [7]. The privacy-preserving and collaborative approach is carried out by three main steps where: (i) all the participating users receive the latest updated weight (W) from the centralized cloud that connects all of them. (ii) The participating users train the model based on the local data they own to have W for each user after training. (iii) Each user uploads their weight to the centralized cloud for the combination of weights and formation of a global model [8]. Such a method uses the centralized cloud to form a global model from users' knowledge after each local training until it reaches a certain convergence criterion. The devices following this collaborative approach never transfer their local data, since only the data knowledge is transferred in the shape of a locally trained model. However, these models are wasting the power of the IoT device since the model is being trained locally [9]. Moreover, the load of communication is large and it exponentially increases with the distance between the cloud and the clients which causes congestion, especially if the IoT devices train a deep learning model which requires large amounts of data transfer after each communication round [10].

1.2 proposed work

To address the above challenges, we propose a light Neural Network for Heart monitoring. The proposed work focuses on light Neural Network, which means fewer neurons of each layer and less number of layers. Hence, IoT acquires from the patient 1D-time-domain ECG data, focusing on the data itself instead of NN for better performance. If the data is split and processed correctly, the 1D-CNN can extract features with less number of CNN layers without the need for more filters. The lighter the NN the least computational power is needed from IoT's processor to train the model. Based on this, we implement this method on Centralized learning, where we gather the information centrally, and by tuning the segmentation frequency to be around one heartbeat, the data passes to the model. We used 1D-CNN to extract the heartbeat features, and then sent them to the model to be trained. After we managed to get a high performance compared to segmented data and compared to other papers, we moved to collaborative learning, with the same data pre-processing setup. In federated learning, the data will have the concept of its environment, which means the data is represented by its users' local dataset, not by the population distribution. Hence, the design of the federated algorithm needs to address the above constraints. The imbalance in the dataset and wrong distribution of classes over clients lead to bias toward class or environment which leads to poor performance. However, the federated average can solve such an issue by averaging the weights of all the environments to make it global, and a global model will be used later on in IoTs as an application. Overall, our system model targets IoT devices and addresses their constraints, at the same time and without loss of generality, this thesis is considered as a case study, applying lightweight CL and FL in an intelligent healthcare system. The Healthcare system includes thousands of patients that need regular monitoring, and because of the new pandemic situations, a huge amount of load faced the healthcare systems. Thus, the healthcare system needs to move to the home site with remote monitoring of the patient’s condition or home care. Hence, such an intelligent system can enable those services to patients by collecting information from the patients and processing it to identify the patient's health status. However, the collected data needs to be stored locally instead of violating patients’ privacy. Therefore, we found that federated learning enables remote monitoring of patients and helps envision an intelligent healthcare system.

The developed approaches have been tested comprehensively on different scenarios via real-world datasets. Comparing our results to other papers that use real-world datasets confirm that the proposed approach has near-optimal performance for different data and configuration distribution.

CHAPTER 2: BACKGROUND AND LITERATURE REVIEW

There are three ways of connecting artificially intelligent applications to the cloud, allowing data and feedback communication in between, where the client is the patients, and the cloud is the hospital. By the three methodologies namely, centralized, decentralized, and federated learning, we can add intelligence to embedded systems. In this chapter, Centralized Learning and Collaborative Learning will be reviewed. In addition, the performance metric equations used to evaluate our system model are presented, in addition to the dataset used in training the model.

2.1 Centralized Learning

Centralized learning occurs when nodes are connected to the cloud. The nodes are mainly Internet of Things (IoT) devices that do not have enough power to process data and train a large amount of data, or the application is shared between many other IoT devices. So, they use centralized cloud computing to save the dataset and train a model. Such a learning paradigm made a quantum leap in Artificial intelligence, where the old way of building smart systems using AI models is by collecting data, training AI models in the labs, and tuning it for a certain application, then leaving it into devices [11]. The architecture in [12] proposed multichannel neural network in a smartwatch. Each lead is assigned to a different channel, the convolutional filter extracts features from each channel, then maps all the features to the features' vector before passing it to the MLP to classify the heartbeat. But for real-time application and variable data drift, centralized learning collects all the data on the cloud to train the AI model, and they use clients' embedded systems from different environments for data acquisition. Then the training and sometimes the prediction happens online. The generalization is the main advantage of the centralized methodology, where the model can generalize based on a group of embedded systems and thus instantly work with other embedded systems. In our first system, we used centralized learning with a machine learning model to identify and classify different types of arrhythmias. The model uses embedded systems to early notify a patient with chronic diseases of an abnormal beat. Where the patient or client is not using medical instruments anymore, IoT–cloud monitoring system can be used instead.

2.1.1 Centralized Learning Arrhythmia related work

In detecting arrhythmia, developers use the same techniques the doctors use for identifying and classifying arrhythmia, and the methodology uses sensors that record heartbeat, and from the shape of the heartbeat, arrhythmia could be classified. Different AI methodologies and different models could be used, and each model depends on the data type or understanding of the data (features). These features are used to feed a deep neural network to recognize diverse kinds of arrhythmias. Rajkumar, A, et al. [13] present a system model that uses the time domain of ECG signal's extracted feature from a dataset and passes it to a one-dimensional neural network, the Exponential Linear Unit (ELU) activation function was used in training the model since it gave them better accuracy than other activation functions. But those features in many cases are contaminated by the noisy signal, this is because of the highly amplified ECG signal by the acquisition instruments [14]. To overcome such a problem, Nurmaini, Siti, et al. [15] proposed a Deep Learning model with an autoencoder (AE) for feature learning. Instead of removing the noise from the high and low-frequency signal, they mapped the signal into different pre-defined frequencies, and then they used the autoencoder to reconstruct the signal again. Then, the neural network layer was pre-trained and fine-tuned to identify arrhythmia. As a result, the model can extract high-level features from unseen data.

Other techniques attempt to extract ECG signals from the frequency domain and time-frequency domain besides the time domain and use different models depending on the feature extracted. The processing of ECG signals is not easy due to its complex stationary formation. In fact, the formation changes with respect to time forming different Heartbeat waves. This variation of the heartbeat is noticed not only between two patients, but the variation also varies with the same patient too [16]. Hence, the best way of processing data is using a nonlinear combination to extract such hidden features in the signal. Thus, the model will perform well even if the data is noisy. In one study [17], higher-order spectrum (HOS) was used to extract hidden features in the ECG signal instead of finding known features such as amplitude or peaks lengths. Instead, the second, third, and fourth order were used as the selected features with a fuzzy neural network (FNN) as an early warning application. The FNN was able to classify seven ECG classes with one normal class and the other six abnormal classes of a heartbeat. The proposed application shows a performance accuracy of 98%. R. J. Martis et. Al. [18] use the same HOS methodology, with the third-order only instead of taking three other orders as features. The captured features were fed to a neural network (NN), least squares (LS) support vector machine (SVM) evaluating different algorithms to identify normal heartbeat and classify the four types of arrhythmias. NN performed well with the features extracted from the ECG signal with obtained accuracy of 93.48%. Kutlu and Kuntalp [19] proposed an arrhythmia detection model that classifies the heartbeats that the Association for the Advancement of Medical Instrumentation (AAMI) specifies. Although the algorithm did reach 99% accuracy using K-nearest neighbor (KNN), such a system is not efficient in terms of power and storage. KNN algorithm uses the storage to store all the datasets in one place and, when it is time to classify, it looks through all the data sets to find the nearest K neighbors. For real-time applications such as ECG identification and classification, such an algorithm is not efficient, especially for small embedded systems such as IoT. Fan [20], uses different methodology as an input feature to the SVM algorithm. Two combined features were used namely, wavelet coefficients and power spectral density (PSD). This method did not do as well as other algorithms. But compared to other papers [18], the performance of SVM accuracy increases around 4% to achieve an average accuracy of 96.73%.

Arumugam, M. et. Al. [21] use a dedicated wavelet to accurately identify the location and the amplitude of ECG segments P, Q, R, S, and T sub-waves as shown in Figure 1. Extracting different energies at different wavelets to identify arrhythmia from the ECG signal does not efficiently work with IoT devices, where many preprocessing is needed to extract such complex features to train the model. Going back to the time domain, the model could be trained on a stream of sequences that represent the ECG signal called 1D, which takes into consideration the amplitude of the signal. Kiranyaz et. Al.[22], proposed for the first time a compact 1D-CNN ECG classifier in real-time, and achieved the state-of-the-art performance by splitting the MIT-BIH dataset based on AAMI recommendation. However, a recent study [23], pointed to the fact that the convolutional neural network is similar to it is predecessor MLP, where both algorithms are homogeneous networks with a sole linear neural model [24]. Similar to the biological neurons, MLP and CNN have a learning ability to a non-linearly separable problem through crude neurons and future prediction based on the learning paradigm.

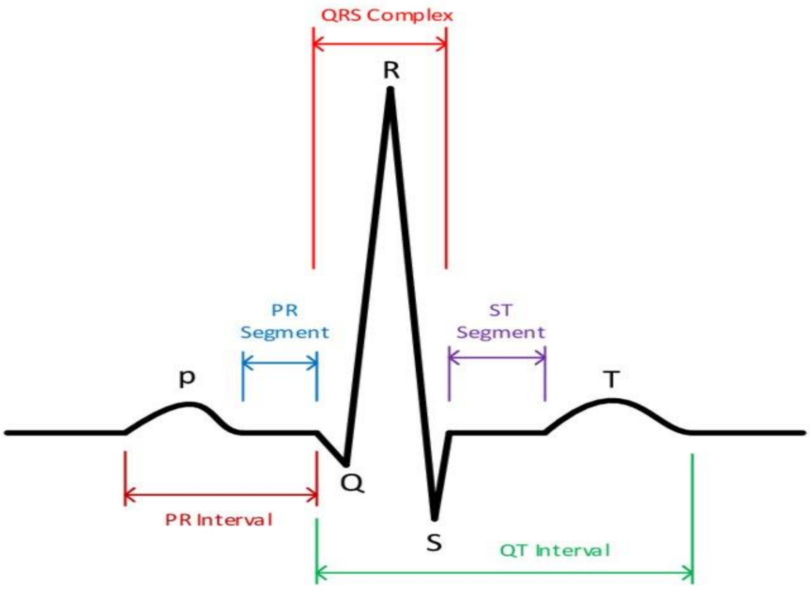

Figure 1. The typical ECG-beat [25]

Thereafter, CNN [22] has been proposed as a dedicated 1D paradigm that can be trained for each patient as a compact classifier, the performance of ECG identification and classification could be performed with the utmost speed with a few hundreds of 1D convolution. As result, the convolutional layers were indeed the best choice for light-weight real-time advanced ECG monitoring and warning application. Similarly, Dokur, Z., & Ölmez, T. [26], proposed a computer-aided system embedded with an AI model that classifies arrhythmia in real-time. They evaluate two different systems: one uses the ECG signal as raw data and feeds a 1D-CNN input layer. The Second measures the performance after converting the raw data into another format, namely image. The system takes as input a 2D picture of the ECG classifying the heartbeat visually. This method requires an extra step before training the model, where the heartbeat needs to be plotted and then 2D-CNN extract the features such as curves and straight lines using convolutional filters. This methodology did not extract extra features to support the prediction accuracy. The average performance of the image approach was not as good as the 1D-CNN method, with a 1% difference between the 1D and 2D, 99% and 98%, respectively. Moreover, taking into consideration the different network structures of the one-dimensional and two-dimensional models, training a 1D model takes a much shorter time than 2D. However, both training and testing were quite fast and this is because in both methods a small-size network where used.

The above method opens a discussion on how efficient converting ECG from one format to another is, after acquiring data. In the study of Wu et al. [27], the MIT-BIH dataset was split into two subsets normal and abnormal, with 2D input space converting the data into an image in order to have better classification performance with a large network. The performance of the model achieves an average accuracy of 98% with tuning different hyperparameters. In a study done by Jun et al. [28] and Rubin et al. [29], both papers used the same technique, but [29] uses the short-time FFT of ECG and converts the FFT to an image before training the model, the CNN extracts features to train MLP. The proposed methodology gives an accuracy of 90%, which is less than using the heartbeat image which achieves an accuracy around 98%.

2.2 Federated Learning

It is hard to create an AI application without the dataset. Moreover, users' privacy is one of the most important priorities nowadays. Federated learning brings users' knowledge about a specific application to the cloud without bringing the data to the cloud to train a specific model. This methodology requires training at some stage, where the researchers thought of using the user’s embedded system to do that. But this methodology will be biased to the user’s environment and other users might have different environments with which the application is not familiar [30]. So, gathering all the models at the cloud and summing them up will give the cloud an overall weight of different environments. And then the cloud can send those summed weights to the clients giving them a different view on the clients' environments, besides themselves. In a real-time application such as ECG classification and identification and with the growth of nations and diseases, the data flow moves fast, and to keep such a system up to date requires a lot of resources and computational power; therefore, acquiring data and training it locally will have many benefits [31].

2.2.1 Federated Learning Arrhythmia related work

In a real-time-based application for ECG monitoring and classification using federated learning, several techniques were used, and it has been discussed in CL. But many of the techniques are not suitable for such small embedded systems due to IoT constraints. The smaller the network the better for IoT, but this might raise other constraints such as the performance of a smaller network, which brings the scientist to a closed loop. Therefore, other methodologies were used next to the small network to increase the performance of the models. Recent research done by Sakib, S., [31], uses an asynchronous federated learning-based approach. The methodology was new for federated learning. New hyperparameters were introduced to the method such as the number of clients to wait before the start of federated averaging. This idea did not only enhance the performance but also reduced bandwidth consumption too. The main disadvantage of this asynchronous approach is that the model is not generalized to all clients' environments anymore. Instead, the model is biased toward the clients who finish first, since any client who is below the model's threshold of the number of participants has been ignored by the current round. The proposed method was done on the MIT BIH dataset to classify and identify arrhythmia. As mentioned before, the drawback of the asynchronous method is that the performance of the model was slightly less than the synchronized approach, with a performance accuracy of 95% only in 20 communication rounds. Using minimized federated learning into hierarchical layers for communication in ECG classification application is an efficient way of reducing bandwidth consumption. Abdellatif, A. et. Al. [32], present an early Heretical Federated Learning HFL approach to classify heartbeat for IoT heterogeneous systems. The non-uniformly of distributed data is another issue facing the collaborative learning approach, where if the classes are present in one node, the learning will be easier. This study takes into consideration the non-IID next to the generic class of machine learning models that are trained using a gradient-descent-based scheme. The proposed solution shows an effective performance of reducing communication overhead by providing a 75–85% reduction in the communication rounds between edge nodes and the centralized server, for the same model accuracy. In [33], a heavy federated learning application model is proposed, where deep learning and expandable artificial intelligence are used in ECG online monitoring. They performed the evaluation over noisy and clean data, with 5-fold cross-validations, and the existing work achieved accuracy up to 94.5% and 98.9%, respectively, for arrhythmia detection. Hence, clean data improves the accuracy next to the deep neural network, but this brings us back to the beginning where the limited resources of the IoT cannot handle the cleaning of the data and training such big network.

2.2.2 Challenges of Federated Learning in healthcare systems

We highlight the uniqueness of the proposed work represented by federated learning compared to other model schemes. The distributed training algorithm enables as many users to collaborate in learning a model without sharing local data. Hence, many factors are controlled by the user such as the computational resources and data distribution due to their learning environment. However, for efficient leveraging of collaborative learning within many patients participating in the healthcare system, many challenges need to be addressed. Those challenges are present in most federated learning applications scenarios.

Given the non-IID nature of the healthcare system, the data is heterogeneous between the patients, since the data acquired at each user significantly varies. Typically, the data for training in collaborative learning is acquired using different devices (IoT) attached or nearby to the patient, which results in a non-homogeneous and not identical data distribution between the collaborative nodes. The result, FedAvg, will suffer due to the big number of communication rounds between the Edge users and the central cloud, especially in the case of imbalanced data [34].

Besides the imbalance and not identically distributed data, given the big number of chronically ill and elderly people, most of the hospitals need to serve this big number of patients daily. This puts a significant load on the health sector. Such a promising application for health demand will transfer the traditional large number of patients with mild conditions to home care, while being nursed remotely through the cloud. Such promising applications need more expansion so that healthcare is not confined within hospitals.

Finally, we discuss the challenge of the limited resources with the users. Given a large number of clients participating with limited computational power in their embedded systems next to the network nodes is really challenging for federated learning. A linear relation between the network traffic and the number of trainers creates congestion. This is because the clients need to update the model. Hence, distributing the available network to clients to guarantee efficient communication is not an easy task. This is because the data distribution discussed earlier varies from one user to another. Moreover, and in addition to the distribution and data variety, the energy availability and the distance from the edge introduce an extra constraint that needs to be optimized.

2.3 MIT-BIH

Electrocardiography (ECG) datasets were collected by hospitals. The aim of research at the beginning from the dataset is to have a reference to different heart activities. Since those data contain normal and abnormal beats, they are suitable to be used by AI models as training and testing datasets. There are many ECG datasets that have been collected and each dataset has its quality such as the amount of noise and the number of samples of each class such, as PTB Diagnostic ECG Database and The Massachusetts Institute of Technology University and Boston's Beth Israel Hospital (MIT-BIH) dataset [35]. The MIT-BIH is widely used by medical schools to train doctors on reading ECG to identify and classify abnormal and normal beats. It contains around 24 hours of recording from 48 different patients, where each record is around 30 minutes per patient, and corresponds to acquiring ECG record from 2-leads. Lead-I is extensively used by many developers in the training of AI models while excluding lead-II since it records ECG from legs. Lead I data was obtained by placing the ECG electrodes on the chest and left arm, where those are the two places our embedded systems record data from. According to the Association for the Advancement of Medical Instrumentation (AAMI) [36], the heartbeat is split into 5 parts (normal beat (N), supraventricular beat (S), fusion beat (F), ventricular beat (V), and unknown beat (Q)), where each part states the heart's behavior. From the heart's behavior, we can study the rhythm of the heart using the MIT-BIH database that contains 109,446 samples as shown in Figure 2, and classify arrhythmias.

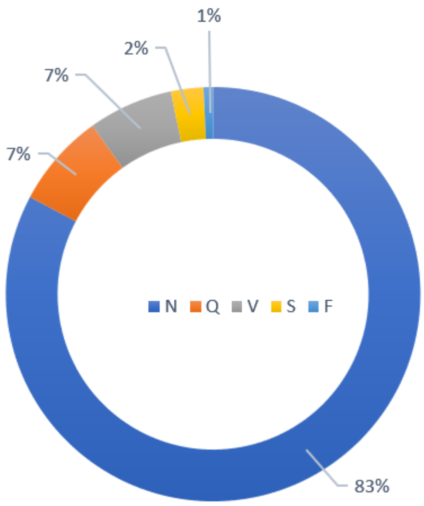

Figure 2. Class distribution in the MIT-BIH Dataset.

The ECG signal was sampled at 360 Hz and digitized at 11-bit resolution. The signal is also segmented and centred at the R-peak; each segment contains around 260 samples, and for other samples, we used padding with zero so, all the segments are at the same size.

However, there is a variable number of datasets that could be used in developing our model. We used MIT-BIH in training and testing, and this is because it has been used to train doctors on detecting arrhythmia. In addition, a huge amount of previous work was done on the same dataset; thus, using it maintains the fairness in comparison with previous work since we did not pre-process the data by removing extra noise and extract other features.

2.4 Evaluation Metrics

The following equations have been utilized to evaluate the performance of the model [37]. The evaluation metrics of the model use four parameters that represent the status of the prediction. They are generally defined as TP for the truly predicted positive, the truly predicted negative is TN, false negative (FN) is the falsely predicted as negative, and finally the false positive (FP). These parameters could be calculated from the confusion matrix of the model. Accuracy (1) can be used to measure the overall performance of a model where it depends on all the accurate predictions over all the predictions made, and usually accuracy tells how true this method is.

$$Accuracy = \frac{\sum_{i=1}^{l} TP_i + \sum_{i=1}^{l} TN_i}{\sum_{i=1}^{l} TP_i + \sum_{i=1}^{l} TN_i + \sum_{i=1}^{l} FP_i + \sum_{i=1}^{l} FN_i} \qquad (1)$$

But accuracy does not generalize the performance of the model and getting very high accuracy does not mean that the model is not overfitting the testing set. Therefore, other metrics have been used to evaluate the confidence of the model such as F1-score, where it measures the ratio between the recall and precision. Recall or sensitivity (2) is the measurement of all truly positive predictions over all truly predicted observations. It has been used as a metric to measure the capacity of a model to correctly classify an event.

$$Recall = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} TP_i + \sum_{i=1}^{l} FN_i} \qquad (2)$$

While the precision ratio (3) is the truly predicted positives overall positive observations.

$$precision = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} TP_i + \sum_{i=1}^{l} FP_i} \qquad (3)$$

$$F1-Score = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} TP_i + \frac{1}{2}(\sum_{i=1}^{l} FP_i + FN_i)} = 2 \cdot \frac{precision \cdot Recall}{precision + Recall} \tag{4}$$

The study aims to propose a novel approach in classifying multiclass model of arrhythmia using a one-dimensional (1D) convolutional neural network (CNN), by assuming first the hospital as a centralized cloud and storing the dataset in the hospital and training a model in chapter 3 to:

- Detect arrhythmia using CNN,
- Optimize the 1D-CNN with a minimum number of hidden layers with a high level of accuracy, and
- Evaluate the arrhythmia detection method using the MIT-BIH dataset.

The objective is to overcome the weaknesses of the previous works that apply deep Neural Network and many preprocessing which does not match with the IoT constraints in terms of computational power and power resources. While in chapter 4, our proposed work uses the same model in chapter 3 by considering 1D-CNN to:

- Detect arrhythmia using a collaborative approach (Federated learning).
- Optimize the 1D-CNN with a minimum number of hidden layers with a high level of accuracy in Federated learning.
- Evaluate the arrhythmia detection method using the MIT-BIH dataset in Federated learning.

This is because patients' medical records should be maintained private. To overcome the privacy constraint, the centralized cloud should only see the knowledge of the sub-local dataset of each patient. In addition, this permits to overcome the weakness of the centralized learning that most of the previous work has, which is the bandwidth constraint for IoT network in addition to the other constraints discussed in the

Centralized approach.

CHAPTER 3: CENTRALIZED LEARNING

3.1 Introduction:

Typically, Centralized learning (CL) is a learning methodology that assumes that patients provide relevant data to train a model. Hospitals for example are the centralized cloud in our scenario, where all inpatient and normal patients' data is collected and sent to the cloud. Then the cloud preprocesses data before training its model locally.

The data is collected from the inpatient using medical sensors or normal users through IoT devices embedded with a sensor. The collected data allows the centralized cloud to have a look at data first to have a general understanding of the type and the number of classes it has, and then extracts features before training and predicting.

After feature extraction and training the model online on a cloud, the prediction part comes, where the model can predict online by keeping acquiring data from patients and notifying doctors if it detects abnormal behavior. But this method consumes the bandwidth and consumes power since the sensors need to connect to the cloud and send all normal and abnormal behavior. This method could be acceptable for the inpatients since the medical sensors are close to the cloud. While in IoT devices this methodology is expensive due to IoT constraints, since IoT devices are small, have small batteries, and have low computational power. Consequently, consuming all its power in sensing and sending data to the cloud is not efficient.

Therefore, the second methodology is to upload the pre-trained model on the IoT devices, where the prediction will take place locally and the devices will send a notification to the doctor only if the model predicts abnormal behaviour. Figure 3 shows how the system is connected and how it works.

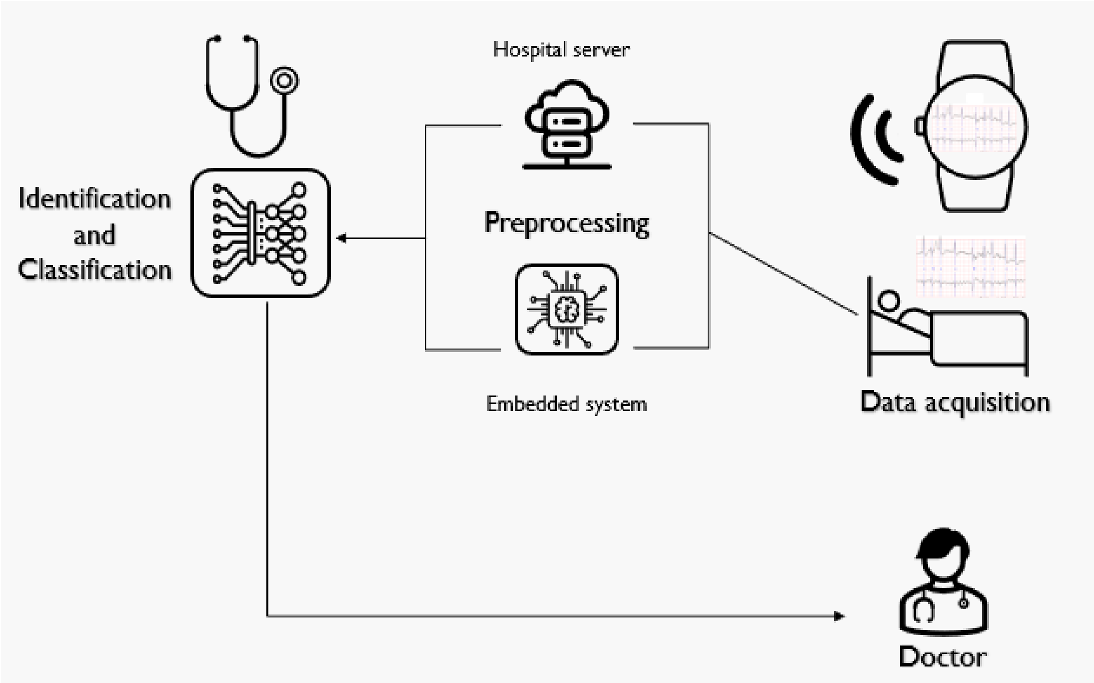

Figure 3. Centralized Learning system Architecture

In new environments such as mobile abnormal behavior detection with an intelligent system, however, the power and processing constraints can significantly affect performance and prevent the use of such a system. Given this issue, a tiny intelligent system should replace traditional heavy-weight algorithms that consume power in addition to communication. In fact, the lightweight system should be faster, and this means that it will consume less power.

Moreover, centralized learning allows supervising the system model and enhances model accuracy, due to the large amount of data collected in one place and training the same model. So, in this chapter, we are modelling a multiclass arrhythmia detection system using a Convolutional neural network and evaluating the system using the MIT-BIH dataset using a centralized learning approach.

3.2 Centralized Learning Model:

The Centralized learning model consists of pre-processing, modelling, and representing a learning algorithm. The proposed architecture starts with data acquisition, where we selected a dataset that has been used to train doctors on how to identify the type of arrhythmia and find abnormal behaviours to train our CNN model. So, the cloud already has all the data. Although the cloud server is powerful and can pre-process the data by filtering the noise and applying various filters to extract some unnecessary features that are captured with the ECG signal, we used one light general pre-processing set that guaranteed the same result whatever the methodology used to predict arrhythmia. The same idea will be used in next chapters where it requires a simple and light way of pre-processing the data.

In an ECG signal CNN, the feature extraction process will look for features that follow patterns and marks such as P, T, Q, and R segments. Increasing the layers of the filter to identify those segments will not increase the accuracy by much. Moreover, the accuracy of the prediction of the filter is much higher than calculating the length of the segment.

The most widely used category cross-entropy is used in our methodology to calculate the loss of the 5 classes discussed in related work. In centralized learning, data is gathered, and the learning is generalized by the main cloud, therefore the loss is simply calculated as [38]:

$$f_i(w) = -\sum_{j=1}^{c} y_j \cdot \log(\hat{y}_j) + (1 - y_j) \cdot \log\left(1 - \hat{y}_j\right) \quad (5)$$

$$f_{cl}(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w) \quad (6)$$

Where w is the model weights and C is the number of classes, y is the true label probability sample (0 or 1), $\hat{y}$ is the predicted probability, and $f_i(w)$ is the loss at iteration i.

As shown above, in formula (6), the loss function does not relate to the number of layers, while it depends on the number of epochs and the rate of learning which generalizes the weight to predict the best fit class. Generalizing the model is important, as overfitting one class could lead to subversion of the model. So, to reduce the hazard of all the above constraints while keeping the model lightweight, or in other words, to predict correctly with the least number of weights involved in prediction, input data's segments should be smaller in size too.

3.3 Centralized Learning Proposed Approach:

3.3.1: Centralized Learning - MIT-BIH:

As mentioned previously we used the MIT-BIH dataset as our collected data from the users, where we split the data into two class two subsets: training and testing, after extracting the electrograms from the recording. Moreover, Doctors did not recommend a specific split of the data, but machine learning scientists who worked on the same dataset recommended [39], the splitting weights for training and testing as indicated in Table 1.

Table 1. scheme of the division of the MIT-BIH database into training (DS1) and testing (DS2).

	N	S	V	F	Q
DS1	46536	946	4034	4034	5628
DS2	44053	1833	3202	388	2411

Where N is the normal beat, S is the supraventricular beat, F is the fusion beat, V and Q is the ventricular beat and an unknown beat respectively.

3.3.2: Centralized Learning Neural Network (NN):

Using the neural network in classification problems is more efficient because it is the simplified version of the human brain using mathematical structure, where it both works as an information processor and memory. The way that the algorithm works is not only by placing threshold boundaries like many other machine learning algorithms, but more or less it drives a prediction based on its bias [40]. The process structure is shown in Figure 4

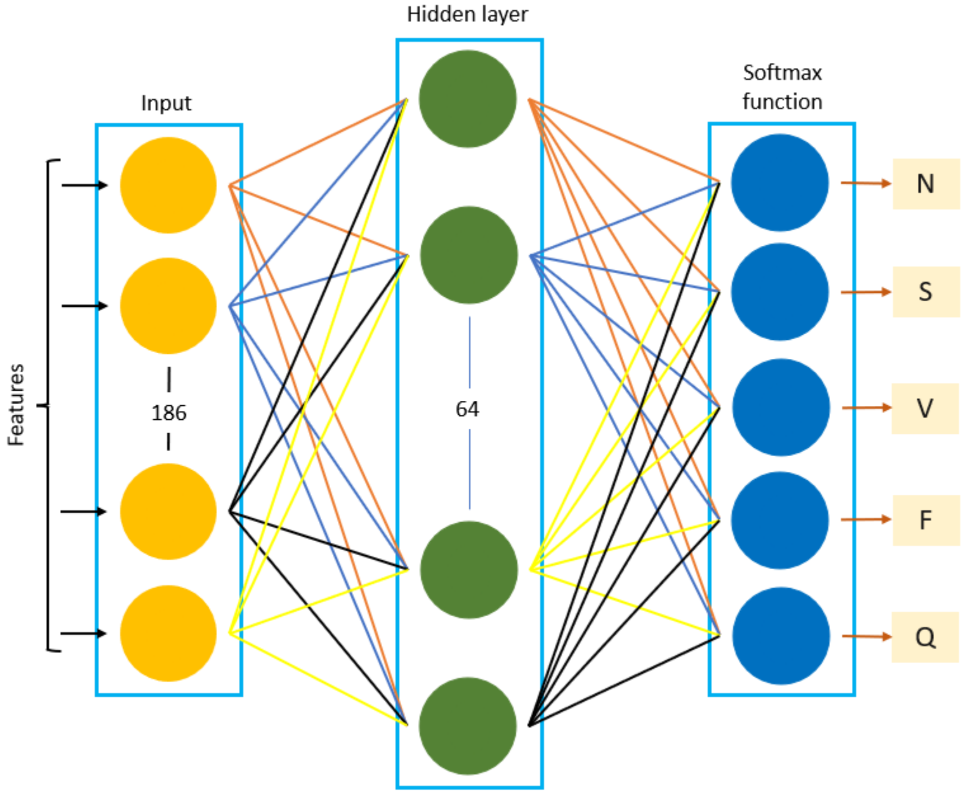

Figure 4. Proposed CNN architecture

Since the mathematical Neural Network learning schemes are based on the input features for prediction, the 30 minutes heartbeat record dataset was broken down into segments. Each segment represents 0.33 seconds which is around one beat, this one beat is the 186-input feature to the NN. In addition, for better generalization, normalization was applied to the input features to confine the peak range of the beat to be between -1 and 1. Moreover, each beat represents one class of the classification problem, therefore one-hot encoding is used to replace the non-numeric labels with a number between 0 and 4: normal, supraventricular, ventricular, fusion, and unknown.

Based on the signal, the NN has an input layer composed of 186 neurons. The results of this layer are the 186 features extracted from the feature extraction layer embedded in the neural network, where the 1-D filters were used in finding lines and curves that represent the heartbeat to suitably distinguish one class from another, followed by one 1-D convolutional neural network hidden layer of 64 neurons, and lastly the output layer of 5- neurons representing the five labels. This is the main setup of the neural network, but experiments did not stop here: Different numbers of hidden layers were tested and different numbers of neurons, as detailed in the results of Section 3.4. In addition, dropping out random neurons in training improves generalizing, where after each hidden layer, dropout took place to reduce overfitting and reduce the loss; thus, improving generalization error.

The activation function is another factor that plays a role in modeling classification problems. A widely used activation function, called Rectified Linear Unit (ReLU), was used in the system. The ReLU activation function runs after each layer and restricts neurons from firing if the input of the neurons is below zero. Since the aim of the model is to identify and classify arrhythmia classes, ReLU with its binary output fires the same neurons' input in case it is positive, otherwise for negative neurons' input fires zero.

3.4 Centralized Learning Results:

In this section, the Centralized learning (CL) system will be evaluated. The system is trained on a different number of epochs to evaluate the accuracy, F1-score, recall, and precision.

3.4.1 Performance Evaluation without Preprocessing:

Figure 5. shows the accuracy of the model using unprocessed input data. The model had an overall accuracy of 93 percent with 10 epochs. But this does not mean that the performance of classifying the classes is accurate. The model performed highest in identifying normal beats with an average accuracy of 96%, but the accuracy dropped down in predicting abnormal beats: Only 23% of correct predictions for the supraventricular beat and 88% in predicting ventricular beat, while for the rest, the prediction drops to around zero as in Figure 6

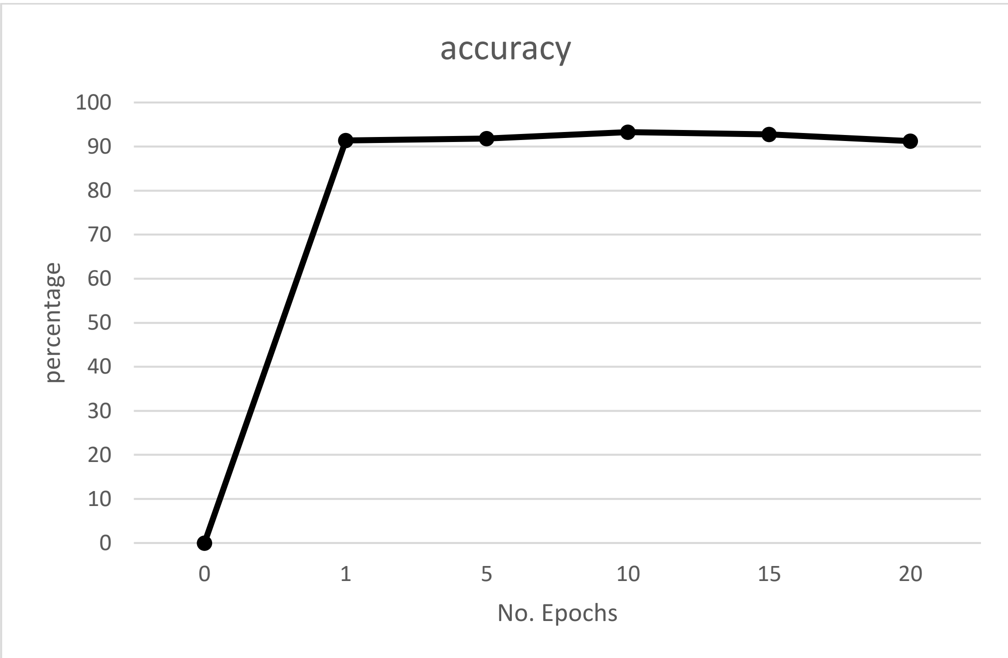

Figure 5. Accuracy vs. number of Epochs

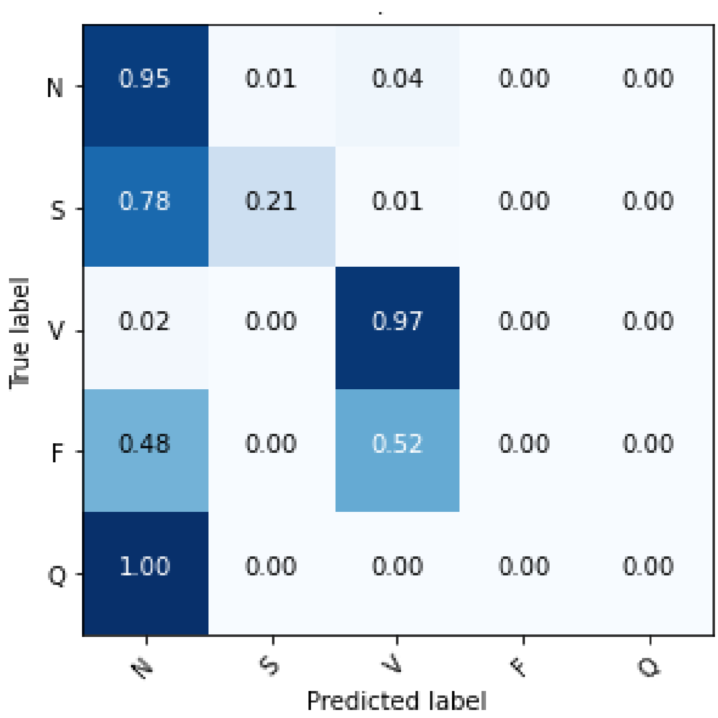

Figure 6. Confusion matrix for 20 epochs

Figure 7. shows poor performance in classifying classes where the average of the classes did not reach 50% of accuracy, which means more than half of predictions were predicted wrong, although the training loss goes to near zero in training the model. But from Table 2 we conclude that the model cannot handle the data with those few hidden layers and the number of neurons.

Table 2. Classification Results 20 Epochs for Centralized Learning.

	Precision	Recall	F1-score
N	0.96	0.96	0.96
S	0.23	0.23	0.23
V	0.81	0.96	0.88
F	0.05	0	0
Q	0	0	0

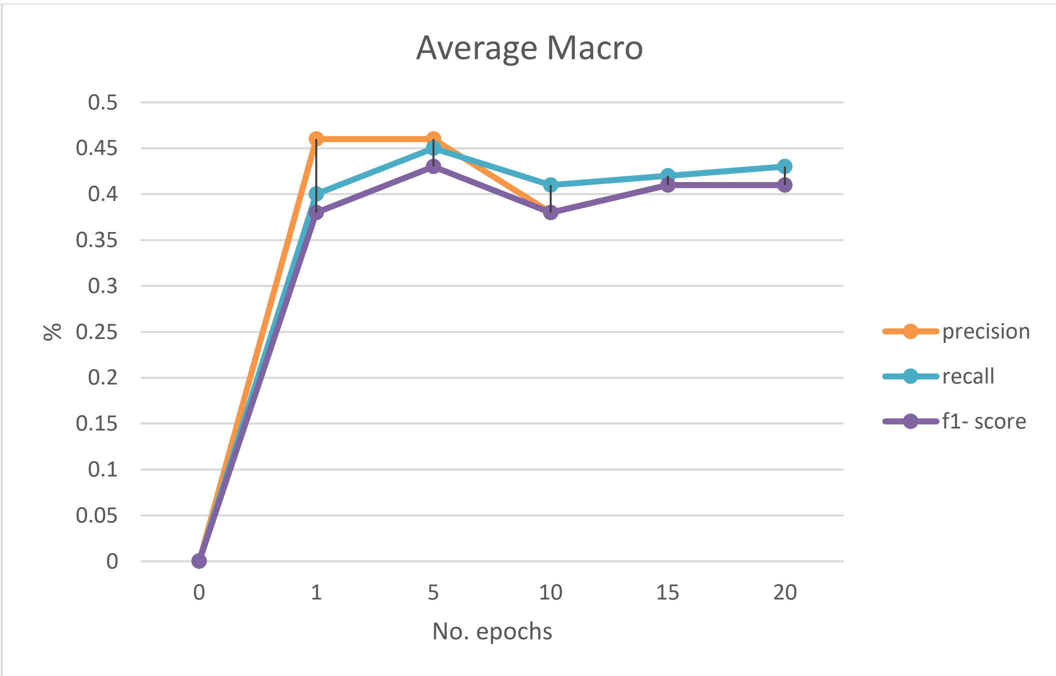

Figure 7. Average Macro precision, recall, and F1-score

3.4.2 Performance Evaluation with Preprocessing:

In this section, the performance of the centralized learning will be evaluated with the pre-processed dataset. The system is trained on a different number of epochs to evaluate the accuracy, precision, recall, and F1-score.

Figure 8 shows the accuracy of the model after data has been preprocessed using segmentation methodology. The accuracy performance increases 8%, noticing that NN is the same. Moreover, the fact that the neural network consists of one hidden layer makes it a fast learner, as with 20 epochs only the accuracy was 99%.

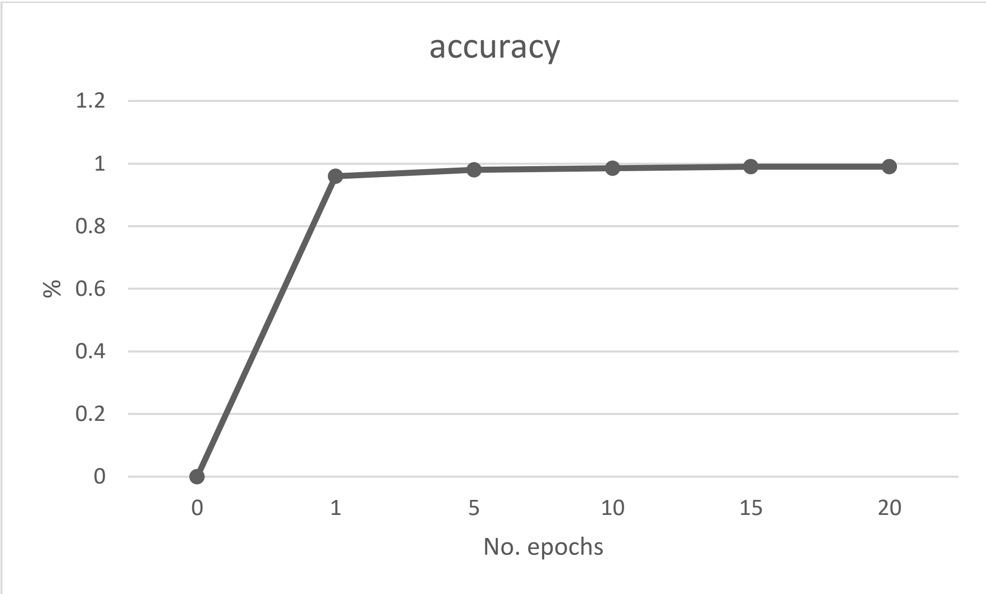

Figure 8. preprocessed data's accuracy vs. number of Epochs

In Figure 9, Figure 10, and Figure 11, it is shown that some classes needed more than five epochs to achieve 90%, but on the other side, many classes achieve more than 95% in five epochs. On the other side, with one hidden layer and segmented data, the performance of the average macro of F1-score increases by 55%, recall increases 52%, and precision increments 56%.

Since detecting arrhythmia is a binary experiment; therefore, if the accuracy of the normal beat is high, this means that all the normal beats could be identified, and the chance of a high-performance classifier would rise, leaving the abnormal samples that are not detected as normal to be identified as unknown. Moreover, the classifier will try to match the abnormal beat to one of the arrhythmia beats, and if it fails it will be classified as an abnormal and unknown beat.

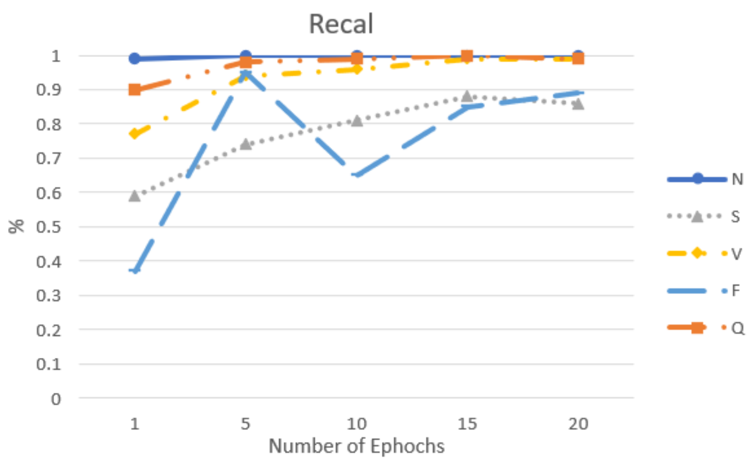

Figure 9. Recall vs. number of Epochs

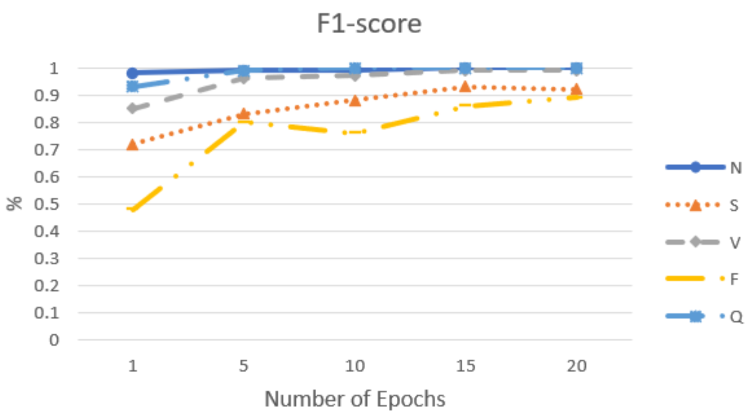

Figure 10. F1-score vs. number of Epochs

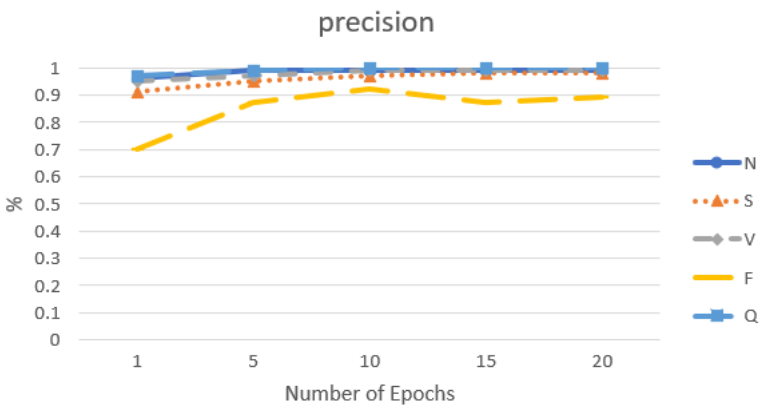

Figure 11. Precision vs. number of Epochs

Table 3. summarizes the results of Figure 9 to Figure 11 with 20 epochs. In addition, to further demonstrate our model's ability, a confusion matrix was derived and shown in Figure 12. The selection of the last activation layer is based on multiple experiments in which it was deduced that the categorical cross-entropy is optimized when it was based on the SoftMax function. The SoftMax function was tuned to be suitable for such classifier applications with a learning rate of 0.0001.

Moreover, compared with the unsegmented data, there is a lot of difference in the confidence matrix. The performance increases for several classes where more than half of the predictions were classified wrong. Moreover, compared with Table 2 that summarized unprocessed data, it can be seen that most of the classes have better performance such as ventricular beat and an unknown beat where the average performance increased by around 94%.

Table 3. Classification Results in 20 Epochs with preprocessed data

	Precision	Recall	F1-score
N	0.99	1.0	1.0
S	0.98	0.86	0.92
V	0.99	0.99	0.99
F	0.89	0.89	0.89
Q	1.0	0.99	1.0

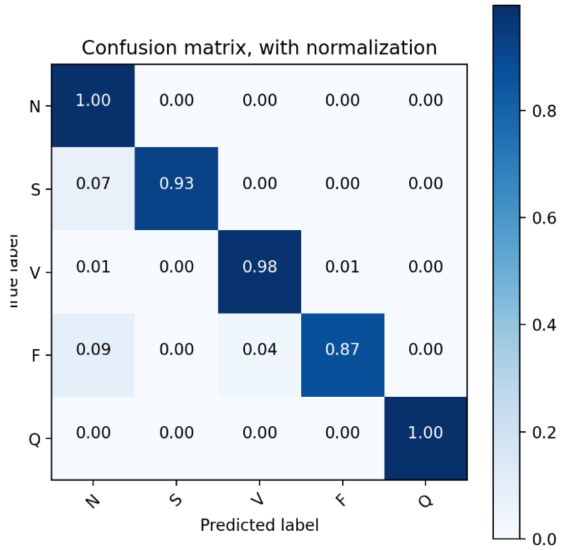

Figure 12. Confusion matrix for 20 epochs.

3.5 Centralized Learning Result comparison:

Table 4 shows a comparison of our approach to other methods investigated in the relevant literature. Our method achieves an average accuracy of 99%, which exceeds, or is comparable to, the accuracies of other methods. However, our method achieved this high accuracy while having the least number of hidden layers to avoid overfitting the model. Moreover, our method takes the least training time with less than eight minutes, whereas most of the methods in other papers took more than one hour, especially with those that have more than five CNN hidden layers. This is because our system model has been implemented by one hidden layer next to the input layer and SoftMax activation function before the output layer.

It should be noted that all the system models in the reviewed literature used the same dataset. However, they generally used different samples distribution between the training and the testing sets. For example, Shi et al. [41], proposed 16 classes of cardiac arrhythmias showing new state of art methods that could be used in feature work.

Table 4. Studies of ECG classification using MIT-BIH

papers	Method	Acc - %	Epochs
Acharya et al.[42]	9-layer CNN	94.03%	20
Shi et al. [41]	CNN with Multi-input layer	94.20%	100
Z. Yan et al. [43]	7-layer SNN	77.5%	150
Rajkumar, A, et al. [13]	1D-CNN and ELU	93.6%	500
B. Mathunjwa et al. [44]	Multi-layer CNN	99%	300
J. wang et al.[45]	Multi-layer CNN	98.6%	10
Siti, et al. [15]	DNN with AE	99%	200
Proposed method 2021	1-layer input, 1-layer CNN	99%	20

CHAPTER 4: FEDERATED LEARNING

4.1 Introduction

In the centralized learning chapter, we specified the traditional machine learning done to train internet of things IoT devices by connecting to the cloud, to train generic models, and distribute the knowledge – model's weights – to all the devices. The advantage of this methodology is the generalization problem of compatible devices. But the one main issue that faces this methodology is the bandwidth, which is limited in many cases. Moreover, the targeted IoT devices such as smartwatches and such devices with sensitive operational data must be on-site, given that such real-time application requires very low latency and that the data travels at a stable connection [46].

While in decentralized learning the devices use the training and run models locally by avoiding the cloud, this methodology helps in real-time prediction applications, and in addition, it solves centralized learning constraints such as privacy and connectivity. However, each model learns its environment; in the case of data drift this methodology is performing well, but other knowledge is missing from other environments that are useful in predicting events that it is not familiar with. This means that generalization is not present in such a methodology [47].

Therefore, a new methodology is required, such as collaborative learning that shares such environments' experience and ensures data privacy in addition to the bandwidth constraint. Collaborative learning or Federated Learning (FL) is the ability to extend local training between clients using a centralized cloud [48]. This methodology is used for the privacy of data, where the client trains the data locally using the cloud model and shares the weights with the cloud. The centralized cloud averages the weights to have overall knowledge and sends the model after averaging to the clients for prediction. This process of learning is slower than centralized learning since the training process takes cycles to learn what others know and reduce overfitting the local data [49].

Figure 13 shows how the cloud starts by assigning the model’s paradigm to the client's with initially random weights. Then the local training process starts with machine learning, noticing the device needs to run the application next to training the model. Therefore, power constraints are raised here, where the more the number of epochs, the more the consumption of the IoT device’s computational resources, and hence the more the consumption of power. In addition, poisoning the model by applying very few numbers of epochs affects the overall results [49]. This is a debatable topic, and our assumption is that all the clients have the same weight gaining knowledge, which means that all the clients are equally important to our federated model.

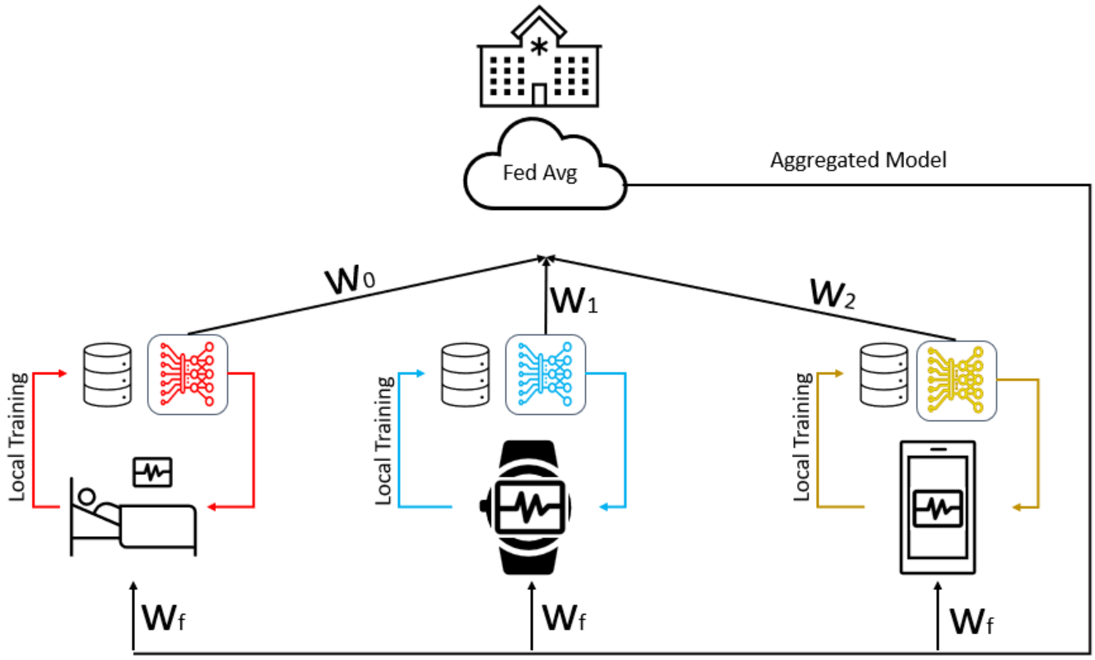

Figure 13. Federated Learning system Architecture

After training the model locally, the weights are sent to the cloud for averaging. The idea behind averaging is to average the knowledge and get a neutral model not biased to a client or a class. As explained before, the training process takes cycles of training by the client and averaging by the cloud back and forth until the model paradigm converges. The faster the convergence the lower the bandwidth usage. This is because the client does not keep updating anymore, which is also another debatable constraint [50]. So, our stopping criterion is that whenever the model converges, the learning cycle stops before it starts overfitting.

The bandwidth consumption in federated learning is lower compared to the bandwidth in centralized learning since clients are not anymore sending raw data. However, it sends the model weight, therefore federated learning has been used widely nowadays in many IoT applications.

4.2 Federated Learning Model

In federated learning, the aim is to optimize the final federated model from n clients, in which, the client trains the least number of epochs, and the learning process takes the least number of cycles per client. Moreover, it should be noted that each client has its own centralized learning, and this process is repeated in all clients. So, (6) is repeated for each client, and at the cloud, they are summed, and weights are averaged. Therefore, the loss function of FL is calculated as [38]:

Where at client k:

$$F_k(w) = \frac{1}{n_k} \sum_{i \in P_k} f_i(w) \qquad (7)$$

But this is not the case in the cloud, where the loss function is calculated as the following:

$$f_{FL}(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w) \qquad (8)$$

K is the number of clients, n_k is the number of samples given to client *k*, P_k is the set of the indexes of the samples given to client *k*, and f_i is the same as (5).

4.3 Federated Learning proposed approach.

4.3.1: Federated Learning - MIT-BIH

The model in federated learning will be the same model that has been used in centralized learning. Moreover, we used the same MIT-BIH dataset but this time the data is split equally over the clients, where randomly the classes will be assigned to each client following independent and identically distributed (IID) methodology. This is because if all users have an equal amount of each class and data, their federated learning accuracy will remain constant, and no new knowledge will be distributed. Therefore, each client will fit the model by the sub dataset. This reflects the scenario of collecting data and training it locally. Table 5 shows two subsets, the first subset is the training subset, where the total number of the subset is divided by the number of users *n,* and the second subset is the testing subset, and this subset is used to evaluate the performance of the collaborative average model.

Table 5. MIT-BIH subsets for Federated Learning

	N	S	V	F	Q
DS1	46536/n	946/n	4034/n	415/n	5628/n
DS2	44053	1833	3202	388	2411

4.3.2: Federated Learning Neural Network (NN):

Adding intelligence to IoT sometimes is costly. Federated learning uses the client's device to train the Neural Network model, and this process takes hours to train one cycle. Thus, the less the power used and the more performing the model, the better is the overall rating. Figure 14 shows 1-D CNN with a different model structure to measure the performance of the application over the dataset. The model structure is made of a number n of hidden layers, and at each hidden layer, there is a number m of neurons. The criteria for finding the best model are when any parameter n or m increases and the accuracy does not increase much, the performance at m-1 or n-1 converges.

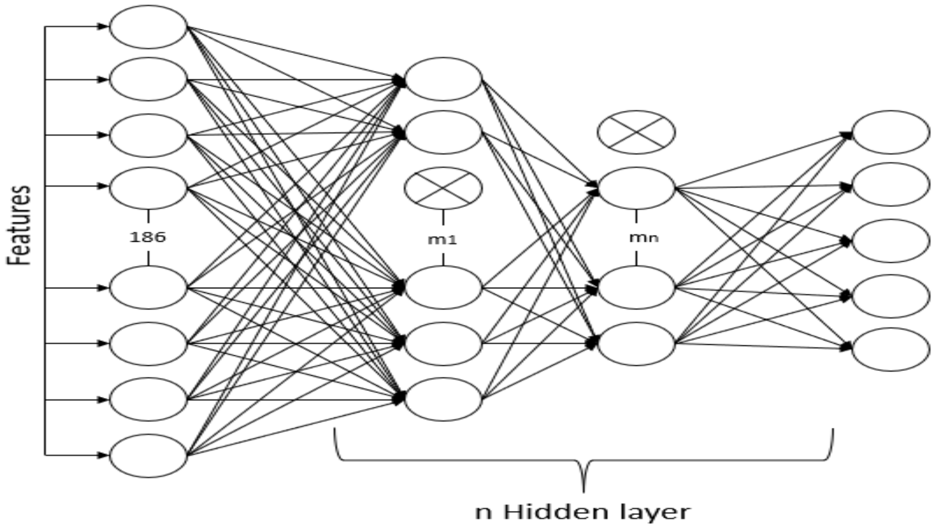

Figure 14. Federated Learning Neural Network

The paradigm of the convolutional neural network is explained in chapter 3.3. The evaluation of different setups took place, and the performance is shown in the results section. To measure the gain of the information and to show that useful information has been transferred between clients, the testing subset has been evaluated using the federated average model. When the model performance increases after each cycle and has not converged yet, there is more useful information to gain. Otherwise, the model converges, and wasting more computational power is not helpful.

4.4 Federated Learning Results:

In this section, we evaluate the model performance over hyperparameters such as the number of layers, number of neurons in each layer, and number of epochs. Moreover, the experiment took place on a different number of clients participating in building the collaborative model. Then, we experiment with the performance of the devices in terms of time and loss versus different hyperparameters.

To investigate the performance of the CNN over the dataset, firstly we measure the federated average performance versus the number of epochs per user. Figure 15 shows the performance of the FedAvg with five epochs per user and different numbers of users. It shows that the best performance was done by the least number of users in 30 rounds with 96%, this is because the more the number of clients the more the number of cycles needed to converge. But on the other side, when the number of users increases to the double, 2n, the time per user to train the model is less by almost half for a 1% difference in performance. Moreover, Figure 16 shows that the accuracy is higher, but the amplitude between the cycles is high, and this is because the model at a higher number of epochs needs more time to converge. In addition, we can see that by increasing the number of epochs, the model overfits the local sub-dataset. Hence, the forward and backpropagation are memorizing the local data. When both the number of users and epochs increase, the model needs more and more cycles to converge.

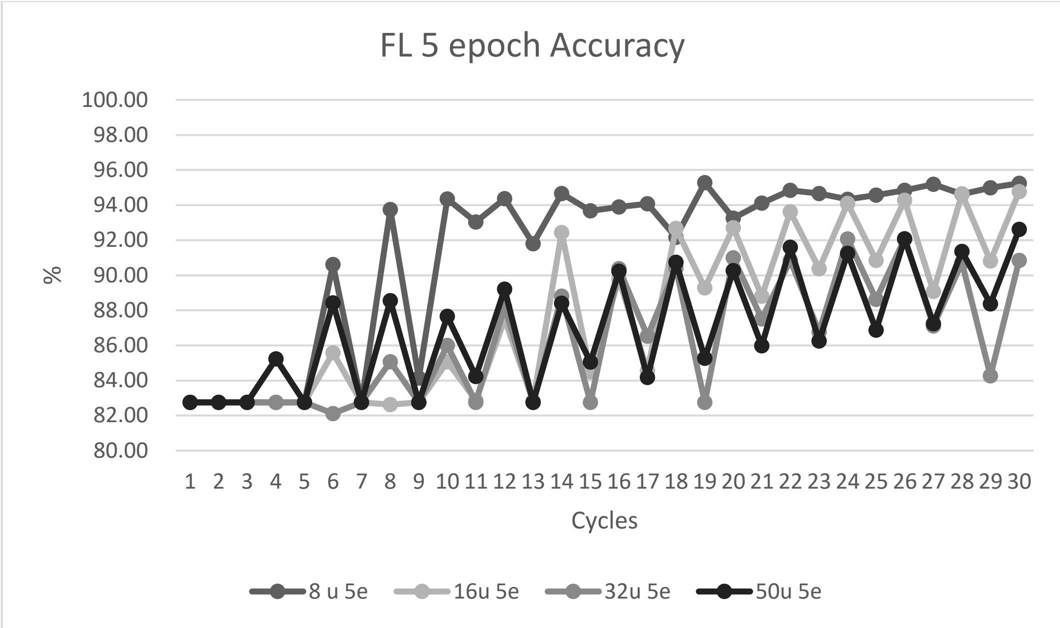

Figure 15. Federated Learning 5e Accuracy of different number of clients

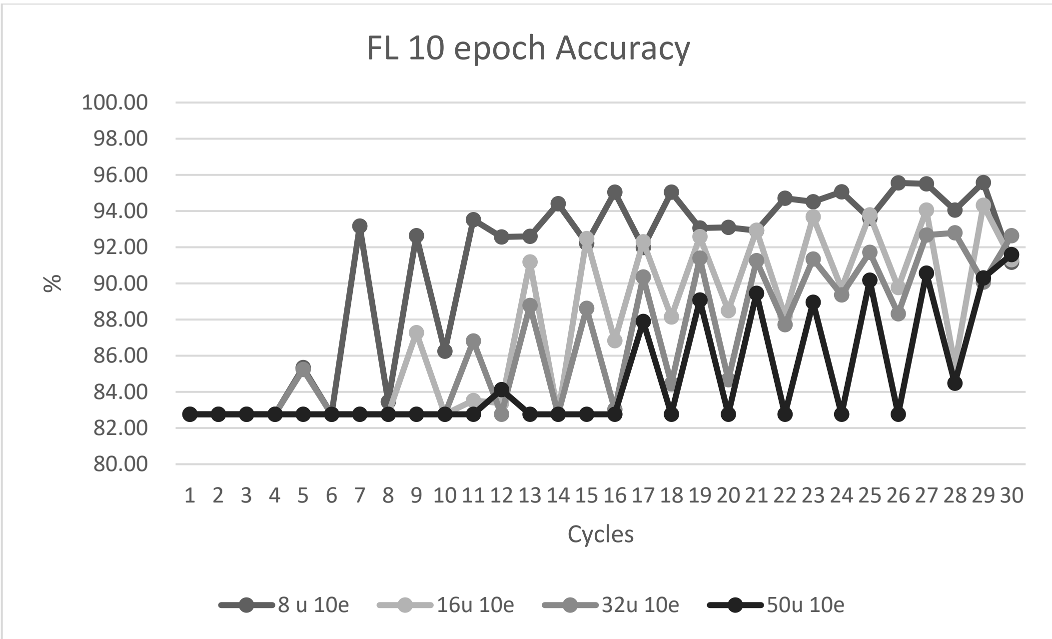

Figure 16. Federated Learning 10e Accuracy of different number of clients

Moreover, Figure 17 shows the loss of the FedAvg versus the number of users. It can be seen that when the user number increases, each user has a fewer amount of data, due to the training subset split equally between the users. Therefore, each user has (training subset/number of users), which leads to higher loss compared with a smaller batch of users. But on the other side, the loss of 50 users after the model converges decreases to around 0.1% compared to the loss of 8 users which after convergence barely scores 0.04%.

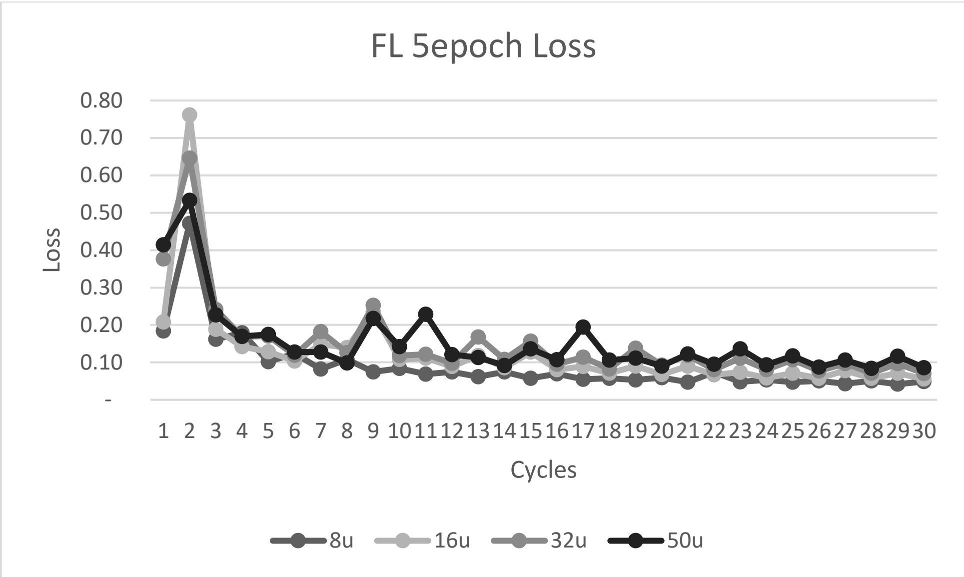

Figure 17. loss in Federated Learning for 5 epochs

Secondly, we evaluate the model when increasing the number of hidden layers. The disadvantage of increasing the number of hidden layers is the number of weights to train, which requires more time. But the advantage is that the performance increases compared with less number of hidden layers. Figure 18 shows the performance of the MIT-BIH when there are 16 users and 5 epochs over 1, 2, and 3 1-D CNN hidden layers. The target is to keep the performance as good as possible while considering the IoT constraints. So, when we increase the number of layers the performance of the model increases by around 1% to 2% for each hidden layer. But this achievement needs more computational power, where it is shown in the graph that the performance of one hidden layer converges in 28 cycles, but when the number of hidden layers increases it takes more cycles. The graph shows that the model completes 50 cycles and still the performance is bouncing, meaning that it did not converge yet. Moreover, in Figure 18, we can see that the model started learning much earlier with one hidden layer compared to 2 hidden layers, that were started performing after 10 cycles, and 3 hidden layers after 18 cycles. This leads to the conclusion that when the number of hidden layers increases the performance gets better; on the other hand, the computational resources get consumed due to delays in convergence which requires more cycles and more power to train the model per cycle.

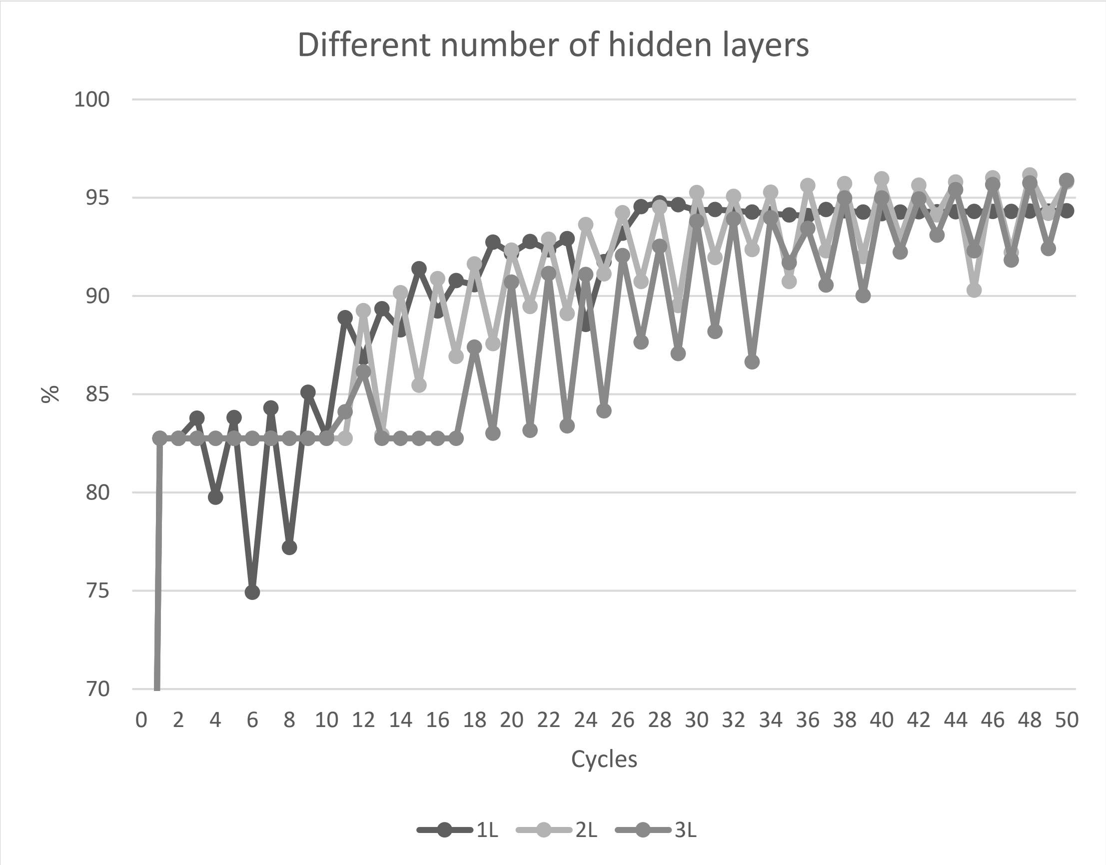

Figure 18. Federated Learning performance with a different number of hidden layers.

Therefore, we experiment with the effect of increasing the dropout. Figure 19 shows the performance of 2 hidden layers over 50% and 15% dropout. It is clear that at the beginning of training cycles the performance of 15% dropout is better, but while the number of cycles increases and after 25 cycles the difference is minimized between the two performances to less than 0.5%. However, the advantage of increasing the dropout rate is fewer neurons are participating in the model randomly, which means that each client takes less energy in training the model as shown in Figure 20. Increasing the dropout by 35% reduces the energy consumption by around 6.6% on average for each client.

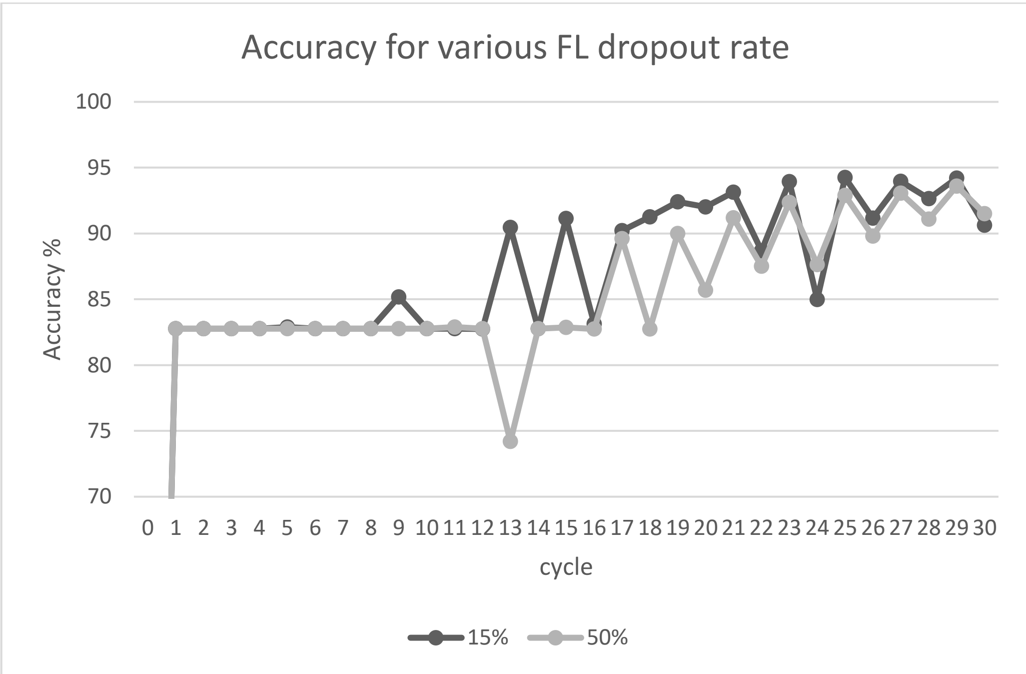

Figure 19. Federated Learning performance with a different dropout rate

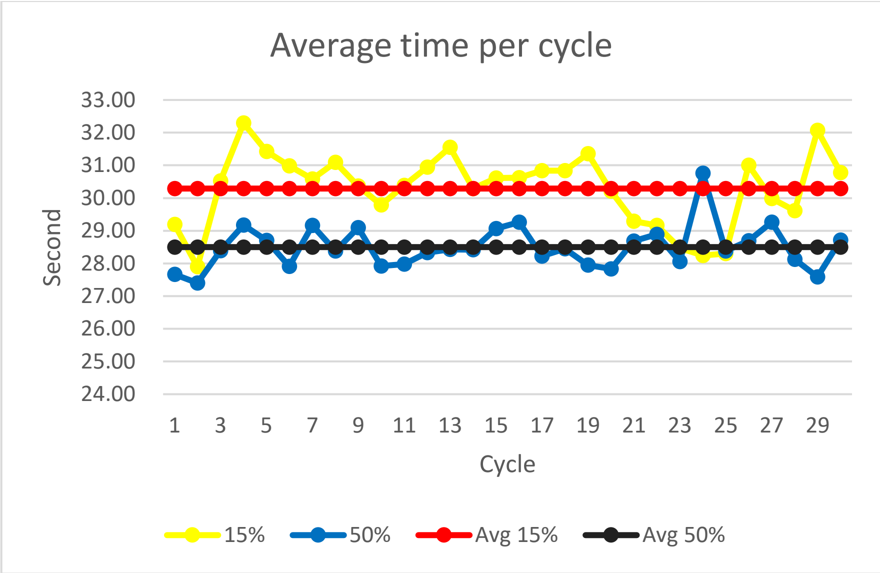

Figure 20. Federated Learning average time per cycle for 15% and 50% dropout

Thirdly, we experiment with the FL model different numbers of neurons with 2 hidden layers to evaluate the performance. The test was done on a model with 5 epochs per user per cycle and 50% neurons dropout. In Figure 21, we can see that when the number of neurons increases, the model performs well in terms of converging. But in fact, the accuracy difference is just 0.5% to 1% for doubling and tripling the number of neurons.

This leads to the conclusion that the hyperparameters should be tuned to such applications. In our case, for example, we are targeting to have the best performance with the least number of weights in the model. This is because we are targeting IoT devices with different environments that have many constraints such as power consumption due to small batteries and low computational power compared to computers.

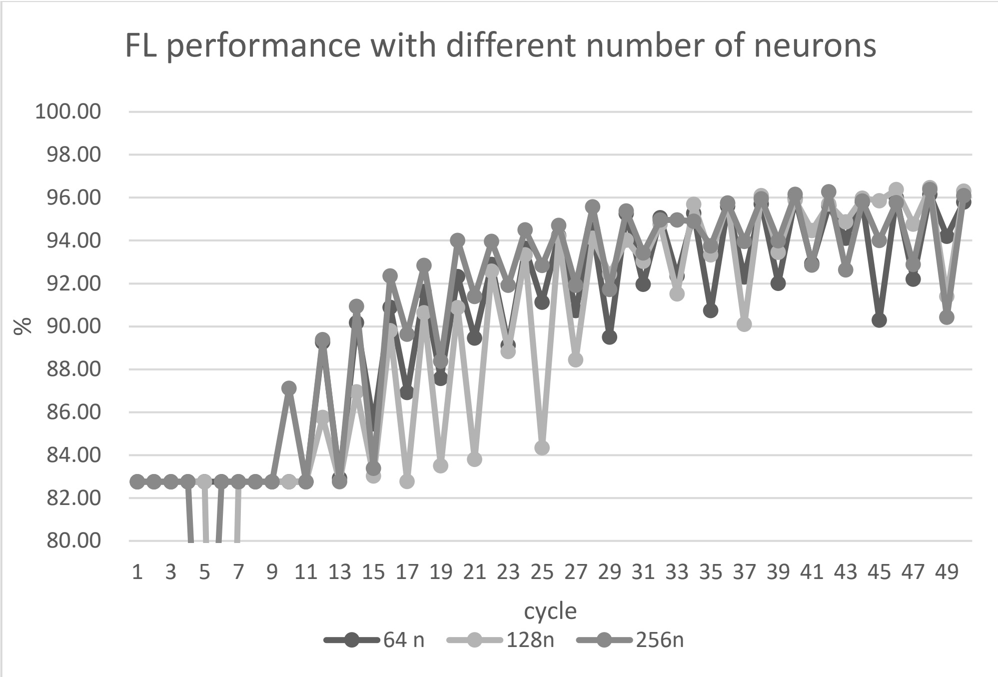

Figure 21. Federated Learning performance with a different number of neurons

4.5 Federated Learning comparison:

Table 6 shows the recent approaches using federated learning to identify and classify arrhythmia. Most of the methods did not focus on the IoT constraints which is why we can see they are getting very good performance with 97.3% and 98.9%. In addition, this is because of the high data treatment, by removing the noise and using a complex model next to a deep neural network [33]. While others consider IoT constraints and use other methodologies such as Asynchronous federated learning and obtained 95% [31]. In our approach, we tried to minimize the client's cost as much as possible with the high-performance model. Our model focuses on feeding the 1-D CNN with segmented data and leaving the feature extraction job to the filter layer. This methodology gives us a 97.27% in 100 training cycles, where the model starts to converge from the 70^{th} cycle using only two 1D-CNN hidden layers. In addition, and since we are trying to minimize the complexity of the model, one CNN hidden layer achieves 95% in 30 cycles. It should be noted that all other approaches use the same MIT-BIH dataset.

Table 6. Studies of ECG classification using Federated Learning

papers	Method	Accuracy %	Cycles
[31]	Asynchronous federated learning-based ECG	95%	20
[32]	hierarchical federated learning for IoT	90%	100
[51]	GRP-FED: 2-layer fully connected, 5 epochs	56.9%	100
[52]	4-conv, 2-dense 1D CNN	97.78%	400
[33]	Data process and noise filter with 7 layers 2D-CNN	98.9%	-
Proposed method	1- Two hidden layers 1D-CNN 2- one hidden layer 1D-CNN	97.27% 95.27%	100 30

4.6 Federated Learning challenges:

Although federated learning is a promising paradigm for real-time applications that ensure preserving privacy by training user's data locally, before sharing the knowledge of the data only without sharing the data. However, there are other concerns facing such a paradigm such as some data needing to be collated in one batch before the model starts training on data. If the model is trained on data whenever it receives data, we will start having resource limitation issue. Since it will need almost infinite power to run such a strategy. Moreover, the IoT's processor will not handle multiple real-time applications working in parallel for almost 24 hours. In addition, in case of storing the data to train the model after collecting an amount of data raises another question, where the data will be stored, since most of the IoT devices are limited in storage too. Hence, such concerns are debatable, and it depends on the developer, whether the developer will increase the space to store data, or with the 5G evolution where he can store it online in a private cloud container dedicated for each IoT device and ignore bandwidth consumption. Many concerns could be raised and sorting one could raise the other.

In addition, for such real-time application development on IoT devices, there are two ways to do it maintaining the same or better performance. The first way is by having excellent sensors that can acquire heartbeat data with almost zero noise. Such sensors are now present in the market, and they can acquire data similar and even better that the MIT-BIH dataset that we used. In addition, if the data is perfectly acquired, the prediction performance will increase due to less noise appearance in the data that can change training weights. However, the second way depends on IoT's processor. Hence, heavy signal processing needs to be done on the signal before it enters such a model and today's IoT processor is compatible. The data limitation challenge of arrhythmia detection application is another concern since the data is old. However, almost 90% of the developers use the same dataset as a standard dataset for training or testing their model for arrhythmia detection. Moreover, generating new data takes time and equipment quality insurance. Since, as mentioned before, the sensors vary in quality and performance and testing the right sensor that is capable with all IoTs is nearly impossible. In addition, MIT-BIH is used to train doctors (cardiologists) in reading heartbeats. Hence, training the model on the same dataset might be efficient.

CHAPTER 5: CONCLUSION AND FUTURE WORK

In this thesis, we studied a smart health system paradigm, where users synchronize the cloud's model, and the cloud synchronizes the user's model. The object is to obtain a light model to be used in IoT devices and at the same time is accurate while having the least number of neural network layers as they are the dominant source of power consumption in this system. Toward this end, we first optimized a centralized learning model which is lighter than other proposed models in the literature for a smart health system, in terms of the number of NN which affects the time consumed to train a model. The performance accuracy of this model depends on the right segmentation frequency and the 1D CNN filters used in extracting features from the data.

We then proposed a private approach for our application based on federated learning. Hence, we bring intelligence to IoT, fine-tuning the model to overcome the IoT's constraints in terms of power consumption and waste of bandwidth. The collaborative approach aims to train the models locally and share the knowledge of different environments such that a global model represents the application. In addition, this method allows data drift to be considered, as we saw recently in the new covid pandemic, such application could be modified with the environment by enabling local training. Eventually, we argued that both the system needed to be light and the communication should be considered in the design of future distributed and learning systems.

To that end, promising future work can look at other algorithms that extract from the ECG signal more numerical values such as the width of the peaks by looking at smaller segments from one heartbeat. In addition, for federating learning we are going to look at online learning in choosing for the client the communication channel based on the congestion at the base station. The idea of not considering all the nodes equally, especially when the IoT is inherently robust to non-IID sub-datasets, is also a topic of interest for future investigation.

Moreover, such a paradigm is still not compatible with centralized learning, where the non-IID affects its performance in terms of accuracy and time needed to converge. However, we could boost the federated model if we manage to set the right initial weights of the global model before the first communication round. Hence, idle data could be used for training in the could before we send the first model, since we know for example how the idle data looks like we can generate some at the server layer first using generative models, or portion of users data and we can generate more data using the same generative models. In addition, the previous way can also save the model from data poisoning, since we cannot know that the data is poisoned unless we can see it.

In addition, although the collaborative approach is more efficient in terms of bandwidth consumption than centralized learning, where only the weights will be sent to the central server over the network unlike its counterpart, where the data needs to be collected and sent to the central server to be trained there. However, developers always would like to go near zero in bandwidth consumption utilizing present approaches. In the future, there might be an application that requires many communication rounds, and such a process consumes bandwidth. Hence, model compression could be one-way of minimizing network consumption. However, this might drop the performance of the model. Another way of doing this is by quantization, where the model has a map in the server that reflects the model’s real weight.

REFERENCES

[1] A. Ghasempour, “Internet of things in smart grid: Architecture, applications, services, key technologies, and challenges,” *Inventions*, vol. 4, no. 1, p. 22, 2019.

[2] S. Sidney, W. D. Rosamond, V. J. Howard, and R. V Luepker, “The ‘heart disease and stroke statistics—2013 update’ and the need for a national cardiovascular surveillance system.” Am Heart Assoc, 2013.

[3] E. Hoefman, P. J. E. Bindels, and H. C. P. M. van Weert, “Efficacy of diagnostic tools for detecting cardiac arrhythmias: systematic literature search,” *Netherlands Hear. J.*, vol. 18, no. 11, pp. 543–551, 2010.

[4] J. Wang, P. Liu, M. F. H. She, S. Nahavandi, and A. Kouzani, “Bag-of-words representation for biomedical time series classification,” *Biomed. Signal Process. Control*, vol. 8, no. 6, pp. 634–644, 2013.

[5] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, “Robust and communication-efficient federated learning from non-iid data,” *IEEE Trans. neural networks Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, 2019.

[6] S. Wang *et al.*, “Adaptive federated learning in resource constrained edge computing systems,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, 2019.

[7] K. Bonawitz *et al.*, “Towards federated learning at scale: System design,” *arXiv Prepr. arXiv1902.01046*, 2019.

[8] M. Abadi *et al.*, “{TensorFlow}: A System for {Large-Scale} Machine Learning,” in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 2016, pp. 265–283.

[9] J. Mills, J. Hu, and G. Min, “Communication-efficient federated learning for wireless edge intelligence in IoT,” *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5986–5994, 2019.

[10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[11] V. Kumar, A. Ghimire, and H. K. Hoon, “Machine Learning and IoT based solutions for detection of arrhythmia using ECG signals,” in *2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*, 2021, pp. 477–484.

[12] Y. Cao *et al.*, “ML-Net: Multi-Channel lightweight network for detecting myocardial infarction,” *IEEE J. Biomed. Heal. Informatics*, vol. 25, no. 10, pp. 3721–3731, 2021.

[13] A. Rajkumar, M. Ganesan, and R. Lavanya, “Arrhythmia classification on ECG using Deep Learning,” in *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, 2019, pp. 365–369.

[14] M. García, M. Martínez-Iniesta, J. Ródenas, J. J. Rieta, and R. Alcaraz, “A novel wavelet-based filtering strategy to remove powerline interference from electrocardiograms with atrial fibrillation,” *Physiol. Meas.*, vol. 39, no. 11, p. 115006, 2018.

[15] S. Nurmaini, A. Darmawahyuni, A. N. Sakti Mukti, M. N. Rachmatullah, F. Firdaus, and B. Tutuko, “Deep learning-based stacked denoising and autoencoder for ECG heartbeat classification,” *Electronics*, vol. 9, no. 1, p. 135, 2020.

[16] R. J. Martis, U. R. Acharya, K. M. Mandana, A. K. Ray, and C. Chakraborty,

"Application of principal component analysis to ECG signals for automated diagnosis of cardiac health," *Expert Syst. Appl.*, vol. 39, no. 14, pp. 11792–11800, 2012.

[17] S. M. Rafi and S. Akthar, "ECG Classification using a Hybrid Deeplearning Approach," in *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, 2021, pp. 302–305.

[18] R. J. Martis, U. R. Acharya, K. M. Mandana, A. K. Ray, and C. Chakraborty, "Cardiac decision making using higher order spectra," *Biomed. Signal Process. Control*, vol. 8, no. 2, pp. 193–203, 2013.

[19] Y. Kutlu and D. Kuntalp, "A multi-stage automatic arrhythmia recognition and classification system," *Comput. Biol. Med.*, vol. 41, no. 1, pp. 37–45, 2011.

[20] F. Fu *et al.*, "Comparison of Machine Learning Algorithms for the Quality Assessment of Wearable ECG Signals Via Lenovo H3 Devices," *J. Med. Biol. Eng.*, vol. 41, no. 2, pp. 231–240, 2021.

[21] M. Arumugam and A. K. Sangaiah, "Arrhythmia identification and classification using wavelet centered methodology in ECG signals," *Concurr. Comput. Pract. Exp.*, vol. 32, no. 17, p. e5553, 2020.

[22] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 3, pp. 664–675, 2015.

[23] S. Kiranyaz, T. Ince, A. Iosifidis, and M. Gabbouj, "Generalized model of biological neural networks: progressive operational perceptrons," in *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 2477–2485.

[24] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in

nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, 1943.

[25] M.-H. Nguyen, Vu-Hoang-Tran, T.-H. Nguyen, and T.-N. Nguyen, "A deep learning framework for inter-patient ECG classification," *Int. J. Comput. Sci. Netw. Secur.*, vol. 19, no. 1, pp. 74–84, 2019.

[26] Z. Dokur and T. Ölmez, "Heartbeat classification by using a convolutional neural network trained with Walsh functions," *Neural Comput. Appl.*, vol. 32, no. 16, pp. 12515–12534, 2020.

[27] Y. Wu, F. Yang, Y. Liu, X. Zha, and S. Yuan, "A comparison of 1-D and 2-D deep convolutional neural networks in ECG classification," *arXiv Prepr. arXiv1810.07088*, 2018.

[28] T. J. Jun, H. M. Nguyen, D. Kang, D. Kim, D. Kim, and Y.-H. Kim, "ECG arrhythmia classification using a 2-D convolutional neural network," *arXiv Prepr. arXiv1804.06812*, 2018.

[29] J. Rubin, R. Abreu, A. Ganguli, S. Nelaturi, I. Matei, and K. Sricharan, "Classifying heart sound recordings using deep convolutional neural networks and mel-frequency cepstral coefficients," in *2016 Computing in cardiology conference (CinC)*, 2016, pp. 813–816.

[30] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated Learning," *Synth. Lect. Artif. Intell. Mach. Learn.*, vol. 13, no. 3, pp. 1–207, 2020.

[31] S. Sakib, M. M. Fouda, Z. M. Fadlullah, K. Abualsaud, E. Yaacoub, and M. Guizani, "Asynchronous federated learning-based ECG analysis for arrhythmia detection," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, 2021, pp. 277–282.

[32] A. A. Abdellatif *et al.*, "Communication-efficient hierarchical federated

learning for IoT heterogeneous systems with imbalanced data," *Futur. Gener. Comput. Syst.*, vol. 128, pp. 406–419, 2022.

[33] A. Raza, K. P. Tran, L. Koehl, and S. Li, "Designing ecg monitoring healthcare system with federated transfer learning and explainable ai," *Knowledge-Based Syst.*, vol. 236, p. 107763, 2022.

[34] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv Prepr. arXiv1907.02189*, 2019.

[35] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Eng. Med. Biol. Mag.*, vol. 20, no. 3, pp. 45–50, 2001.

[36] S. M. Anwar, M. Gul, M. Majid, and M. Alnowami, "Arrhythmia classification of ECG signals using hybrid features," *Comput. Math. Methods Med.*, vol. 2018, 2018.

[37] H. Daumé, *A course in machine learning*. Hal Daumé III, 2017.

[38] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, 2017, pp. 1273–1282.

[39] P.-C. Chang, J.-J. Lin, J.-C. Hsieh, and J. Weng, "Myocardial infarction classification with multi-lead ECG using hidden Markov models and Gaussian mixture models," *Appl. Soft Comput.*, vol. 12, no. 10, pp. 3165–3175, 2012.

[40] M. Feindt and U. Kerzel, "The NeuroBayes neural network package," *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.*, vol. 559, no. 1, pp. 190–194, 2006.

[41] Y. Gao, H. Wang, and Z. Liu, "An end-to-end atrial fibrillation detection by a novel residual-based temporal attention convolutional neural network with exponential nonlinearity loss," *Knowledge-Based Syst.*, vol. 212, p. 106589,

2021.

[42] U. R. Acharya *et al.*, “A deep convolutional neural network model to classify heartbeats,” *Comput. Biol. Med.*, vol. 89, pp. 389–396, 2017.

[43] Z. Yan, J. Zhou, and W.-F. Wong, “Energy efficient ECG classification with spiking neural network,” *Biomed. Signal Process. Control*, vol. 63, p. 102170, 2021.

[44] B. M. Mathunjwa, Y.-T. Lin, C.-H. Lin, M. F. Abbod, and J.-S. Shieh, “ECG arrhythmia classification by using a recurrence plot and convolutional neural network,” *Biomed. Signal Process. Control*, vol. 64, p. 102262, 2021.

[45] J. Wang *et al.*, “Automated ECG classification using a non-local convolutional block attention module,” *Comput. Methods Programs Biomed.*, vol. 203, p. 106006, 2021.

[46] A. Thakkar and K. Kotecha, “Cluster head election for energy and delay constraint applications of wireless sensor network,” *IEEE Sens. J.*, vol. 14, no. 8, pp. 2658–2664, 2014.

[47] L. He, A. Bian, and M. Jaggi, “Cola: Decentralized linear learning,” *Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[48] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, “Federated learning,” *Synth. Lect. Artif. Intell. Mach. Learn.*, vol. 13, no. 3, pp. 1–207, 2019.

[49] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, 2020.

[50] M. Chen, H. V. Poor, W. Saad, and S. Cui, “Convergence time optimization for federated learning over wireless networks,” *IEEE Trans. Wirel. Commun.*, vol.

20, no. 4, pp. 2457–2471, 2020.

[51] Y.-H. Chou, S. Hong, C. Sun, D. Cai, M. Song, and H. Li, “GRP-FED: Addressing Client Imbalance in Federated Learning via Global-Regularized Personalization,” *arXiv Prepr. arXiv2108.13858*, 2021.

[52] Y. Gao *et al.*, “End-to-end evaluation of federated learning and split learning for internet of things,” *arXiv Prepr. arXiv2003.13376*, 2020.