UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

ENERGY EFFICIENT MACHINE LEARNING-BASED CLASSIFICATION OF ECG HEARTBEAT TYPES

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

By

MORGHAN S HARTMANN
Norman, Oklahoma
2018

ENERGY EFFICIENT MACHINE LEARNING-BASED CLASSIFICATION OF ECG HEARTBEAT TYPES

A THESIS APPROVED FOR THE

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

_____

Dr. Ali Imran, Chair

_____

Dr. Thordur Runolfsson

_____

Dr. Gregory MacDonald

Table of Contents

Abstract

To meet the accuracy, latency and energy efficiency requirements during real-time collection and analysis of health data, a distributed edge computing environment is the answer, combined with 5G speeds and modern computing techniques. Using the state-of-the-art machine learning based classification techniques plays a crucial role in creating the optimal healthcare system on the edge. This thesis first provides a background on the current and emerging edge computing classification techniques for healthcare applications, specifically for electrocardiogram (ECG) beat classification. We then present key findings from an extensive survey of over hundred studies on the topic while taxonomizing the literature with respect to key architectural differences, application areas and requirements. Leveraging the insights drawn from the extensive analysis of the pertinent literature we select a set of most promising machine learning based classification techniques for ECG beats, based on their suitability for implementation on a small edge device called a Raspberry Pi. After implementing these classification techniques on a Raspberry Pi based platform we perform a comparison of the performance of these classification techniques with respect to three key performance indicators (KPI) of interest for health care applications namely accuracy, energy efficiency, and latency.

ECG measures the electrical activity of the heart and help healthcare professionals to evaluate heart conditions of a patient, sometimes diagnosing life-threatening conditions. The features of ECG signals are pre-processed and fed into the classification algorithms to detect and classify abnormal beat types. ECG classification requires low complexity but still high level of performance in terms of aforementioned three KPIs. The classification algorithms chosen, namely Naïve Bayes, Multilayer Perceptron (MLP), and distilled deep neural network (DNN) are all energy efficient methods hence suitable for implementation for small edge devices. The comparative multi-faceted evaluation presented in this thesis is a new contribution to research that exists on edge based classification as it offers comparison of selected classification algorithms in terms three KPIs instead of one while using real edge device based implementation. The performance of analyzed machine learning classification

techniques is ranked according to each KPI. Benefiting from the results of the comparative analysis presented in this thesis a particular classification algorithm can be selected for optimal deployment in given scenario in healthcare system depending on the specific requirements of the given scenario. Edge computing paves the way for a new generation of health devices that can offer a higher quality of life for users if low-latency, low-energy, and high- performance requirements are addressed.

List of Tables

List of Figures

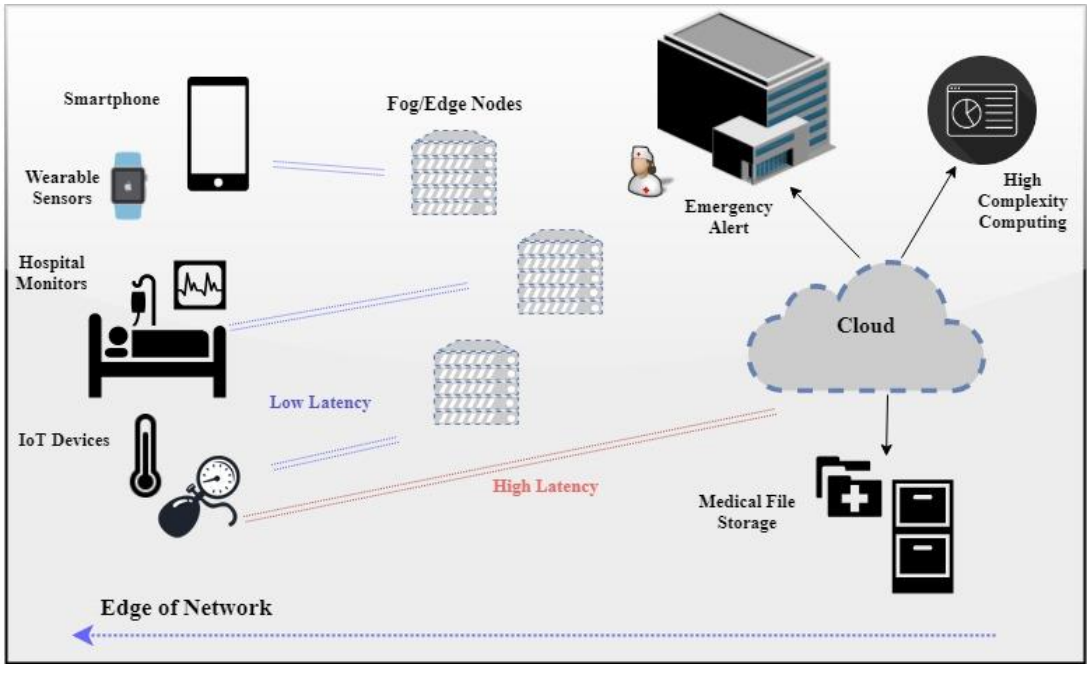
Introduction

1.1 State of the Art in Fog Computing and Edge Devices

The first few papers published on topic of edge computing relate to the use of small, portable devices on the edge of the network for real-time computation and their role in a larger system called fog computing. Bonomi, et al. [1] was the first published work that referred to the term "fog computing." The ideal system would provide low latency IoT services with a large number of nodes. The term "fog" describes a location in reference to the Cloud, which are large data

FIGURE 1.1

Edge Device Location in E-Health Network



centers used for computation and storage. Fog computing hopes to move computing capabilities towards the edge of the network, therefore eliminating the need to transfer the majority of a system's data to the Cloud, which is expensive, time-consuming, and not fast

enough for latency-sensitive health applications. Fog, or edge computing has the following requirements [2]:

- Low latency
- Increased mobility and location awareness
- Energy efficiency
- High level of security
- Usability
- Low operating cost

Current research in edge computing for healthcare uses focuses on measuring certain KPIs that are important for the progression of health services, such as response time, energy efficiency, and bandwidth cost. Papers tend to focus on one of the KPIs for a certain portion of the edge computing process, so the aim of this section is to provide a picture of best data operations techniques for a healthcare edge device.

Classification of raw data collected by health sensors is normally completed using simple or advanced algorithms, depending on the computing power of the device, and is a very common research theme in healthcare-related computing. Activity-based recognition is the most popular research relating to classification in healthcare edge computing, since robust techniques are needed for devices that have lower storage and computing capabilities.

Fall detection algorithms, for example, can be done on the smartphone device. In [3] and [4], fall detection algorithms are run both on a smartphone initially, then on a back-end module connected to a cloud server. The front-end algorithms contain both a root sum squares (RSS) filter for detecting fall-like activity and an activity daily living (ADL) filter that reduces false alarms by matching activities that might be fall-like in nature, like bending over to pick up a dropped item, etc. Accuracy is the general goal for many papers that have results for

classification algorithms. One Class SVM with Gaussian Kernel's accuracy is assessed in [5] to be up to 75% accurate in classifying visiting events in an elderly person's home when room sensors in combination with a wearable Fitbit device is used as a data source. This research shows that sometimes an additional source of data can be useful in increasing accuracy of a system.

Comparing different classification algorithms directly is the best way to determine the most energy efficient or low latency method. For example, in [6], the authors compare three types of machine learning techniques, namely Bayesian belief network (BBN), support vector machine (SVM) and K-nearest neighbors (KNN) on a dataset of breath rate and humidity level. BBN managed to reach the highest accuracy compared to SVM and KNN, however, there was no quantitative study of most energy efficient approach, which is an oversight of much of the research found. However, [7] does have information on low latency measures of four classification algorithms, including BBN. BBN has the lowest latency compared to linear regression, nearest neighbor, and KNN methods for similar data as used in [6]. The authors of "iHealth" use a fuzzy approach to the classification of raw sensor data [8]. The fuzzifier uses membership functions to determine whether temperature and heart rate data are normal or not. Additionally, this fuzzy technique on a device uses 8-10mW less than a device that is Weka J48 decision tree and performs a comparison with two other classification methods for a data set with vital sign and not equipped with the fuzzifier. A similar study [9] takes the environmental information. The experiment reveals that J48 has the lowest classification time as compared to fuzzy c-means (FCM) and random tree (RT). Artificial neural networks (ANN) have exploded recently in classification and as shown by [10], it is a method that has lower error than other techniques, like linear regression and decision trees. [11] also uses a neural network, specifically convolutional neural network (CNN) to classify ECG rhythms with low latency at an edge gateway. Much research takes an existing classification method and modifies

it, like in [12]. The authors' modified Markov modulated poisson process (MMPP), which they term M3P2, correctly classifies more elderly visiting events that MMPP. Increasing the number of attributes can also make for a more useful program. In [13], a Weka AnswerTree correctly classifies 96% of 17 different heart rhythm types, which is more types than existing research at the time of publication.

Some healthcare applications require classification algorithms to work with video or image feeds, such as the application in [14] for activity recognition. The authors use a convolutional neural with an average of 76.06% accuracy. Although this makes for an accurate recognition system, the authors point out that it requires too much energy to run on a simple embedded device and would run best on a server. [15] offers a solution for the classification of voice signals from Parkinson's disease patients. The algorithms run on the Intel Edison fog computer include dynamic time warping (DTW) for single-word recognition in time series data and clinical speech processing chain (CLIP) for pitch and loudness estimation.

TABLE 1.1

RELATED WORKS ON CLASSIFICATION / PREDICTION

Reference	Technique	Information Type	Contribution	Results
[40]	One-Class SVM with Gaussian Kernel	Visiting events, heart rate, sleep patterns	Greater Classification Accuracy	75% Detection rate for labeled data set when Fitbit added to system
[82]	Bayesian Belief Network (BBN)	Vital sign, environment data	Comparison of classification methods	BBN reached highest accuracy compared to Support Vector Machine (SVM) and K-nearest neighbors (KNN)
[15]	Fuzzy Logic Classifier	Heart rate, respiration rate, skin conductance	Lower power consumption technique	Reduction by 8→10mW for fuzzy system compared to non-fuzzy system
[49]	Weka AnswerTree	ECG	Greater number of rhythm types than other classifiers	Correctly classifies 96% of 17 different rhythm types
[41]	Markov modulated multidimensional non-homogeneous Poisson process (M3P2)	Visiting Events	Comparison of classification methods	Outperforms standard Markov modulated Poisson process (MMPP)
[84]	Weka J48 Decision Tree	Vital sign, environmental data	Comparison of low latency classification methods	J48 has lowest classification time compared to fuzzy c-means (FCM) and random tree (RT)
[55]	K-means Clustering	Speech data samples	Comparison of low latency device classification	Raspberry Pi has lower runtime (160ms) compared to the Intel Edison, but higher average CPU %
[83]	Weka Bayesian belief network (BBN)	Vital sign, environmental data	Comparison of low latency classification methods	BBN has lowest classification time (5 min for 213 patients) compared to linear regression, nearest neighbor, and KNN methods
[54]	Artificial Neural Network (ANN)	Gas pollutant sensing	Comparison of Classification Accuracy	ANN has lowest Root mean square error (RMSE) compared to linear regression and decision tree
[72]	Convolutional Neural Network (CNN)	ECG	Low latency classification and data transmission	Using edge gateway in place of cloud computing yields lower round-trip time

1.2 ECG Background

ECG, or Electrocardiogram, is a test that measures electrical activity of the heart. According to the American Heart Association, the test can identify parts of the heart that have been damaged,

FIGURE 1.2

ECG Signal Components

overworked, or are too large to be healthy [17]. The test is routine and harmless, as no electricity is transmitted to the body. The key parts of the ECG are pictured below in Figure 2.

The features labeled refer to the stimulation and contraction of different parts of the heart muscle. The P-Wave (see Figure 1) is the action of the atria, or upper parts of the heart. The QRS Wave refers to the ventricles, or the lower parts of the heart contracting. The P wave signals the end of the heartbeat and represents the heart muscles resetting for another contraction sequence. P interval, QRS area, and T interval are common extracted feature for the classification. Each person has a slightly different normal ECG signal, depending on gender, height, and weight, among other factors [18]. The problem with ECG data is the large amount of data that can be amassed by the sensors, especially if the number of leads is increased. Even ECG samples for small periods of time can take up megabytes of storage, which is the case for the MIT arrhythmia database. Each sample contains 30 minutes of two-

lead data, which consists of approximately 18 MB. For a small device used on the edge of the network with limited computing and storage capabilities, this is not an acceptable amount.

1.3 ECG Classification on the Network Edge

[19] presents a vision of a smart health care system using an Arduino and Smartphone data aggregator and processor. Data from multiple sensors, including a two-lead ECG sensor, is analyzed using a fuzzy interference system to classify results of each vital sign as “normal”, “above-normal”, or “emergency.” The device alerts doctors and emergency services for timely treatment of a life-threatening situation. The ECG sensor data is used to classify heart rate as “bradycardia” (low), “normal”, or “tachycardia” (high). Similar membership plots are shown for blood pressure and body temperature. [20] uses Android development board and 7 ECG leads to form a fog-type system for heart abnormality detection. The edge of the system is the sensor tier, which consists of an ECG module with a node that wirelessly transmits data to a mobile computing tier. This tier uses an Android development board and connects to sensors via ZigBee interface for the remote monitoring of patients. A fog IoT health monitoring system, called iCare, is proposed in [21]. In this system, ECG signal data is sent to a smart gateway for feature extraction for the eventual calculation of heart rate. The features extracted from the data is the P-R interval, Q-T interval, S-T interval, and QRS area. Then, with the help of these features, heart rate can be calculated.

On these IoT devices, an algorithm will need to be used for the fast and energy efficient classification of IoT ECG sensor data. In [22], one-minute samples of ECG cycles, with and without arrhythmia, are analyzed using Support Vector Machine (SVM) learning. The use of this low latency technique, combined with the fog node computation in the place of Cloud computing, contributes to a very low latency device (759 ms delay) that can be deployed in an IoT network. The computation is performed on a gateway fog node, an HP Compaq 8200 Elite,

which has 3 GHz speed and 16 GB RAM. The SVM algorithm distinguishes between normal and abnormal heart rhythms based on features from the ECG with an accuracy of 93.6%. However, this work uses a different source of data, namely the "long-term ST Database" on Physiobank.

The choice to use a Raspberry Pi 3 for classification task was formed mainly by the research presented in [23] and [16]. In [23], the authors used the Raspberry Pi 3 as a data processor and for temporary storage of data relating to cardiac monitoring. It achieves a high throughput and is 5G compliant, which is a requirement for future IoT networks. The Raspberry Pi has 1GB RAM and is able to communicate using the 802.11 protocol along with Bluetooth and has ethernet ports. The authors of [16] propose a Smart-Fog architecture for real-time classification of speech abnormalities in Parkinson's patients. When using a K-means clustering algorithm, The Raspberry Pi outperforms the Intel Edison in terms of lower runtime but has a slightly higher memory and average CPU usage than the Edison. However, in a healthcare field, having a fast device is the goal for patients to have lifesaving medical attention, so this slight difference can be overlooked.

The use of machine learning techniques for medical data classification on the edge is a common theme of today's relevant research, however, there are a few that focus specifically on ECG signal classification. For example, in [18], an Intel mote runs an RBF algorithm for the analysis of normal and abnormal heart beats. At random, 23 patients' data were picked from the MIT database of 48 patients. The training data consisted of several features extracted from the raw data, including the P-Wave, RST-Wave, and T-Wave offsets. However, the algorithm was run separately on each individual, and in addition, many of the results on some reports are quite low in cases where the number of abnormal beats is sparse. In [13], classification of ECG abnormalities is done on a PDA, which is a somewhat outdated technology. Nevertheless, the reports on classification comparisons are highly detailed, which was helpful in the selection of

models for this research. The most accurate models used in [13] were the decision tree, neural network, and nearest neighbor clustering, which all had over 91%. However, the neural network had a training time of 2 hours, 10 min and the other two had times of over 5 minutes. A technique that had an average classification accuracy of 70% and a very low time was the Bayesian classifier, which made it a good candidate for future testing. The authors were able to classify 15 beat types from the database, in particular, the beats relating to ventricular flutter arrhythmia (labeled!, E, and F in the database), which is a condition that needs medical attention in less than three minutes to avoid fatality. A recent survey on ECG classification tasks outlines some of the common classification procedures, but has only one that classifies ECG beats on raw data. Using MLP and a nearest neighbor approach, the author was able to achieve 99% accuracy.

1.4 MIT Arrhythmia Database

There are several publicly available databases to use for arrhythmia classification tasks, including those on EDB, AHA, CU, and NSD. However, the most popular to use is the MIT Arrhythmia Database [24], since it has the best documentation and most beat types represented. The MIT Arrhythmia Database available on Physiobank contains 48 patient ECG records, each 30 minutes long. The heartbeats fall into five "super classes" –normal, supraventricular ectopic beats, ventricular ectopic beats, fusion beats, and unknown beats. The beats are further classified into 18 distinct types, each represented in the database records by a character. Figure

2 is an excerpt from one of the database's patient samples. From the "V" labels, it becomes clear that this patient suffers from a premature ventricular contraction, however, the "." Labels correspond to normal heartbeats. Many of the patient files contain more than one type of heartbeat. Table II outlines the basic symbols used to describe the beat classifications.

FIGURE 1.3

Example ECG Signal from MIT database

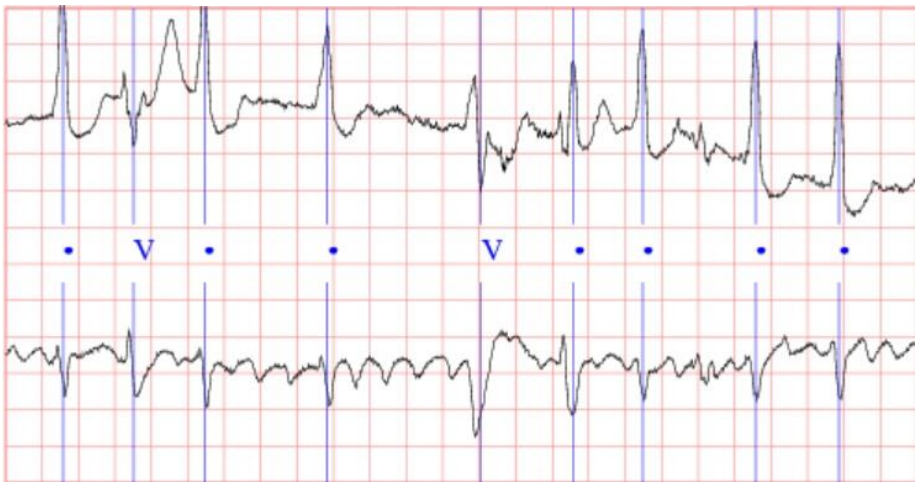


TABLE 1.2

Beat Annotation Symbols in Data

N	Normal Beat
L	Left bundle branch block beat
R	Right bundle branch block beat
A	Atrial premature beat
a	Aberrated atrial premature beat
J	Nodal premature beat
S	Supraventricular premature beat
V	Premature ventricular contraction
F	Fusion of ventricular and normal beat
!	Ventricular flutter wave
e	Atrial escape beat
j	Nodal escape beat
E	Ventricular escape beat
/	Paced beat
f	Fusion of paced and normal beat
x	Non-conducted P-wave (blocked APB)
Q	Unclassifiable beat
|	Isolated QRS-like artifact

1.5 Contributions

The contributions of this thesis are as follows:

- Survey of edge computing, with a focus on classification techniques used for edge device based medical applications.
- Meaningful selection of machine learning methods based on previous works on edge computing for healthcare applications, especially from [49] which provides an analysis, however slightly outdated, on many algorithms on the same dataset. This thesis uses state-of-the-art techniques to continue this work's analysis not only by comparing accuracy, but also runtime and CPU usage for optimal classification on a small device. In addition, this work compares four different techniques, which provides insight into the optimal one to use in a future 5G fog computing setup.
- Summary and mathematical theory behind machine learning techniques chosen for analysis
- Selection of a raw data input for the classifiers sets this project apart from previous works in ECG classification. A large dataset takes up storage and, when processed, memory that a small device does not have. Previous work uses a large number of features, so to test if sparse features can be used, extracted sparse raw data from the MIT database is chosen for the input to reduce the memory-related constraints as well as storage constraints.
- Previous work in ECG classification using deep learning techniques do not take into account energy efficiency and latency constraints for edge-based deployments. This work adds to these by providing an analysis in CPU usage, which is especially important for small devices that have less than 8 GB of memory, and latency of runtime, which is another requirement for fast medical devices.

- Raspberry Pi, the device on which algorithms are run in the experimental setup, is a representative machine for modeling a fog or edge-based scenario in the future. Raspberry Pi is 5G-compatible and can run Python code efficiently and faster when compared to a similar Arduino model.

- This work extends to a variation of beat types included in the machine learning techniques. In addition to showing results for 14 beat types, it also extends to abnormal versus normal beats with a focus on distinguishing between normal beats and the most urgent beat types that need diagnosis quickly (less than three minutes).

1.6 Articles Currently Under Review for Publication

2. M. Hartmann, U. Hashmi, C. Ge, A. Imran, "Edge Computing in Smart Health Care Systems: Review, Challenges and Research Directions" in ETT Special Issue (Submitted December 2018)
3. M. Hartmann and A. Imran. "Deep Learning based classification of different types of arrhythmia using ECG data through a low-cost low energy edge deployable device" (To be Submitted January 2019)

1.7 Organization

The first chapter gives additional background information on the topics of edge computing, medical applications of edge computing, and basic ECG information and previous works on classification of these signal types. Chapter two outlines the classification methods that will be used, as well as each one's mathematical background and information on why it was chosen for the comparison. Now with the preliminaries completed, the next chapters focus on the actual data processing. Chapter three explains the types of pre-processing methods, including how the data was obtained and features extracted. Chapters four, five, six, and seven contain the results

of the machine learning classification tasks. First, the results of each method are given their own chapter. Then, in chapter seven, the results are compared and final suggestion for optimal technique, based on energy efficiency, accuracy, and latency, is given. The tradeoffs between these three components is also discussed in chapter seven. Last, in chapter eight, is the conclusion, which discusses the comparison results, and future research directions in ECG classification for an edge computing scenario.

CHAPTER 2

Classification Methods

2.1 Classification Model Formulation

For qualitative tasks, where an output variable is in a non-numerical category, such as eye color or animal type, a specific technique called a classifier is used [26]. Classification is commonly used in medicine for many cases. For example, DNA sequence classification for diseases or image classification of tumors to be categorized as cancerous or non-cancerous. In classification, a set of input observations is given to the classifier. Logistic regression techniques cannot be used in these cases, since a natural ordering of the data is not always present. In this research involving ECG, there is no ordering for the types of beats. If a beat is encoded as a "1" and another as "2," this differentiation is not an ordering. One beat cannot be "greater" than another. For example, in this beat classification task, the outputs can be represented by

$$Y = \begin{cases} N, & Normal \\ A, & Atrial\ Premature\ Beat \\ e, & Atrial\ Escape\ Beat \\ & \dots \\ & \dots \end{cases}$$

The classification task has two main parts—model formulation and prediction. The prediction of a beat into a particular class relies on a representative model, which comes from the input variables. Each of the techniques' models are discussed in this chapter, along with a description of the relevant metrics used in the performance evaluations.

2.2 **Classification Metrics**

2.2.1 Error Rate

The primary metric for assessing model accuracy measures how well predictions match the input data. For this, the mean squared error (MSE) is calculated, which is

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

where $\hat{f}$ represents the predicted class resulting from fitting the input data x_i and y_i is the actual value of the output class. However, since the predictions in the ECG dataset task are qualitative, MSE is not calculated. Instead an error rate is used [27]:

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

where I is an indicator variable that decides whether the argument is true or not. For example, if $y_i \neq \hat{y}_i$, then the observation was misclassified, and the argument will be equal one, whereas if $y_i = \hat{y}_i$, then the output will be a zero.

2.2.2 Accuracy

Accuracy, as used in all of the Python directories, is the number of correct predictions compared to the total number of predictions made [28]. In mathematical form, accuracy is the following:

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n}(1 - (y_i - threshold(f(x_i))))^2$$

However, accuracy alone does not give a true performance indication for some classifiers. For example, in the ECG arrhythmia dataset, the number of anomalies compared to normal values in one dataset is extremely low. This would mean that if the number of misclassified instances is equal to the total number of anomalous data points, the accuracy could still be extremely high without

the classifier truly achieving its goal, which is the classification of abnormal and life-threatening beat types.

2.2.3 Logarithmic Loss

Logarithmic loss, called cross-entropy in the python code, is a probability of the algorithm's confidence in a prediction. Loss increases as the probability of the predicted value diverges from its true value. For the portions of the task that involve binary classification (i.e. normal versus abnormal beats), cross entropy [29] is calculated as

$$-(y \log(p) + (1 - y) \log(1 - p))$$

where p represents the predicted probability that x_i belongs to class y_i and y is the binary indicator whether $\hat{y}$ is the correct classification for x.

For multi-class prediction tasks, cross-entropy is slightly different:

$$-\sum_{c=1}^{M} y_{x,c} \log(p_{x,c})$$

In this case, M is the number of classes or beat types and x represents the input observations associated with the data entry.

2.2.4 Confusion Matrix

This table is commonly used in classification analyses since it has an extremely readable format to determine the accuracy of the classifier. A confusion matrix contains a comparison of predictions and accurately predicted outcomes. For example, a binary beat classification may have the resulting confusion matrix:

$$\begin{bmatrix} 200 & 10 \\ 30 & 150 \end{bmatrix}$$

For this, the number of correct classification instances is are on the diagonal line of the matrix, or 200 and 150. The misclassified predictions are 30 and 10, which means that this classifier will have a relatively high accuracy according to this chart. More metrics can be obtained from the confusion matrix, such as precision and recall [30]. For example, in the above example, recall is the ratio of true positives (TP), which is represented by 200 abnormal types correctly classified, and the sum of true positives (TP) and false negatives (FN), which are the normal beats classified as abnormal beats. In this case,

$$recall = \frac{TP}{TP + FN} = \frac{200}{200 + 30}$$

Another similar metric is precision, which is the ratio of true positive (TP) cases to the sum of true positives (TP) and false positives (FP). This shows how well the classifier can predict relevant abnormalities. In the above case,

$$precision = \frac{TP}{TP + FP} = \frac{200}{200 + 10}$$

2.2.5 Classification Time

Because this ECG classification task involves time-sensitive information concerning a patient, the time for a model to be fitted or trained and predictions to be made needs to be low. To find the classification time, the timer function was used in python before the line that calls for model fitting, whereas for prediction time, the timer function is placed before the line that processes the test data for prediction.

2.2.6 Memory Usage

Another constraint for this task is low energy usage. The metric that is used in most publications for edge computing is memory usage, which quantifies the amount of random-access memory that

the program is using. If a CPU uses more memory than it has, the program will be run slowly, which is a problem for medical applications. The reason for this is the device type that the algorithms will be housed. The data amount that is used as the input for the algorithm has some influence on the memory usage, but the complexity of the algorithm is the main source of excess computing. A convolutional neural network, for example, results in a high accuracy for data classification, however, it also has a high memory requirement, which is not feasible for running on a small device at the edge of the network. This tradeoff between energy efficiency and accuracy will be discussed in the next chapter. The three compared techniques were chosen for their low complexity and low memory requirements.

2.3 Naïve Bayes Classifier

The first classification method I tested was the Naïve Bayes. The reason this technique was picked comes from the related work from [13], which shows the low latency and relatively high accuracy results for an edge device with low computational power. The Naïve Bayes classifier uses the concept of the Bayes theorem, which is a central idea in probability [31]. Given a set of variables

$$X = \{x_1, x_2, \dots, x_d\}$$

And a set of outcomes

$$C = \{c_1, c_2, \dots, c_d\}$$

Bayes' rule is calculated for each variable relating to each set of outcomes

$$p(C_j | x_1, x_2, \dots, x_d) \propto p\,(x_1, x_2, \dots, x_d |\, C_j)\, p(C_j)$$

Where the second half of the proportion is the posterior probability that any X belongs to a class C. This can be rewritten as a product of terms

$$p(X|C_j) \propto \prod_{k=1}^{d} p(x_k|C_j)$$

Once every input X is given a probability of belonging to each class C, the X is given a class based on which C has the highest probability. For example, if a beat was given a probability of .3 for normal, .7 for abnormal, the beat would be labeled in the outcome as abnormal. The inputs for this classification problem would be the electrical signal from two leads. For example, for one beat, the values might be

$$X = \{\text{-0.12, -0.08}\}$$

which represents the values of one QRS peak. Next, the probability of X belonging to each class will be assigned.

The specific type of Gaussian classifier used is Gaussian Naïve Bayes, which is a Python Scikit standard function called by the line of code GaussianNB [32]. This function assumes the likelihood of features is a normal distribution. In mathematical terms, this means that the normal density has the form

$$f_c(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2{\sigma_c}^2}(x-\mu_c)^2)$$

where μ and σ are the mean and variance of each class c. Combining this expression with the probability above, the probability that an input belongs to a class c becomes

$$p_c(x) = \frac{\pi_c \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_c)^2)}{\sum_{i=1}^{C}\pi_i \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_i)^2)}$$

where C is the total number of classes and π_c is the prior probability that an observation belongs to a class c. Although this technique relies on a very simple expression, it is extremely fast and works well with classification tasks, one famous example being spam filtering in emails. The simplicity

of the algorithm suggests that it would work well on a small device with limited computing capability.

2.4 Multilayer Perceptron (MLP)

Multilayer perceptron is a form of feedforward neural network, which forms a stacked regression model. The backbone of this algorithm is backpropagation, which comes from the essential knowledge of calculus [27]. MLP has three or more layers: one input layer, one output layer, and one or more hidden layers. For my project, I used 3 hidden layers with 100 neurons each. The model for this scenario is the following:

$$x_n \rightarrow h_1 \rightarrow h_2 \rightarrow h_3 \rightarrow y_n$$

For a classification task with K classes, the cross entropy is

$$J(\theta) = -\sum_{n}\sum_{k=1}^{K} \hat{y}_{nk} log \hat{y}_{nk}(\theta)$$

where θ is equal to the weight matrices of the model layers and n refers to the input number. The backpropagation comes in when the output layer for the model calculates the error and passes this error back through the model to compute the first layer error signal to minimize the total error.

2.5 Distilled Deep Neural Network

Before the distilled neural network is discussed, it is essential to know the basics of a deep neural network. Most neural networks used for classification have multiple convolution and pooling layers, which add complexity to the model. The models used in this work are dense layers, which apply weights to inputs and connect this to an output. So, for an input such as ECG raw recordings, we have:

$$M = \begin{bmatrix} -.003 & -.88 \\ \vdots & \vdots \\ -.3 & -.01 \end{bmatrix}$$

And any input layer for the deep neural network can be represented as [33]

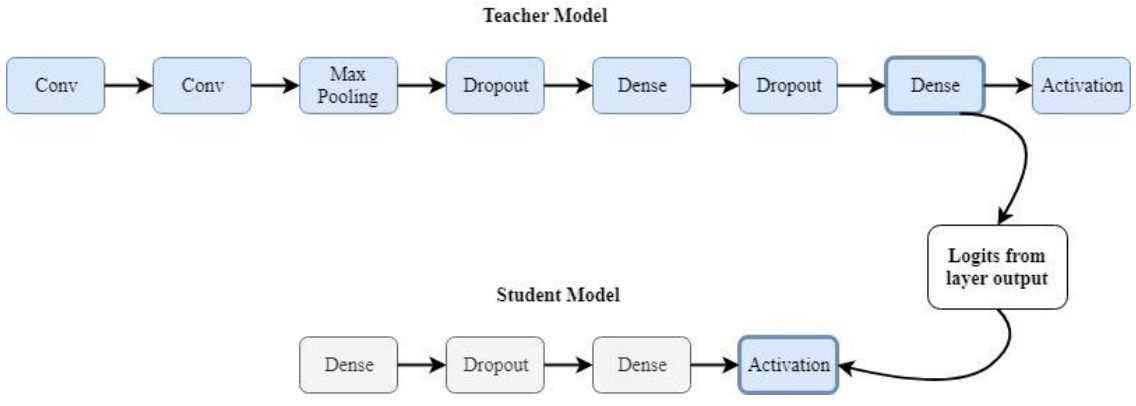
$$h^{in} = M * W + B$$

Where W is the weights and B represents biases from each of the neurons.

The newest state-of-the-art technique aims to gather essential knowledge from one large model or a number of medium-sized models to create a "student" or generalized model that is more energy efficient than its predecessors. This small size is ideal for a small edge device. This student model distills the knowledge from a large neural network using logits, which are the inputs to the teacher's final Softmax layer [34]. Figure III below shows the sequence of events when constructing a student model from a teacher model. The output of the teacher model's final layer is taken as an input to learn on the data at a fast rate. This in turn is used in the place of the Softmax activation layer of the student model.

FIGURE 2.1

Derivation of trained Student Model



2.6 Input Features

The input features chosen differ slightly from existing research discussed in the previous chapter. Using state-of-the-art classifiers appropriate for small devices, simple and fast classification is possible. Since these algorithms and datasets cannot exceed a certain memory usage amount, limited input observations need to be used to ensure this speed and efficiency. The input variables in this case are not the raw ECG signals, which exceed 15 MB for one patient's data. Instead, the raw signals from the amplitude of the QRS complex is taken from each beat type in 40 different patient files. This accounts for only 160 kB once extracted, which is small enough to be run from a small device.

CHAPTER 3

Data Pre-processing

The Pre-processing of MIT Arrhythmia Database files involved several steps:

- First, the files were downloaded from the Physiobank database for medical signals. The annotation files which contain the beat type labels for the duration of each patient's ECG, are separate from the raw signal files, which must be merged via an indexing program that takes the sample number from each annotation and inserts the label for the beat types in the raw signal file.
- Next, to select the features to be used from the data files, a filtering program was run to find the amplitude of each QRS complex, along with the raw signal information at the amplitude of both leads. These selected features were taken from multiple patient files and contained samples of 14 different beat types to analyze the robustness of the learning algorithms to classify beats for any patient.
- To run machine learning algorithms on the new merged datasets, the beat types must be assigned an integer, since the annotations use characters to represent different abnormal beats. Once they are integers, they must be converted to categorical data type via OneHotEncoder function from Python's library.

CHAPTER 4

Naïve Bayes Classifier Results

4.1 Classification of 14 beat types

The naïve Bayes classifier was used for its low memory requirement and simple algorithm, which could easily be run on a small device without lag. When run on 14 beat types, the algorithm was able to classify an average of 60.7% of beat types from the two parameters it was given. The fitting portion of the code was timed to be only 6.4ms, and prediction time only 3.8ms. The memory requirement is also quite low, only 166 MB, which can easily be run on an edge device. Table III outlines the classification report for 14 beat types. Clearly, some of the beat types need additional support for the algorithm, which is why the next step was to experiment with the beat types included in the study.

TABLE 4.1
Naïve Bayes Classification Report for 14 Beat Types

Beat Type	Precision	Recall	F1-Score	Support
A	.69	.94	.79	268
N	0	0	0	114
/	.54	.36	.43	126
f	0	0	0	13
V	.35	.62	.45	92
x	.76	.94	.84	443
F	0	0	0	0
L	.43	.55	.48	209
a	.45	.25	.32	166
J	0	0	0	15
R	0	0	0	1
j	0	0	0	37
S	.60	.35	.44	43
e	.20	.05	.08	20
Average	.53	.61	.55	1547

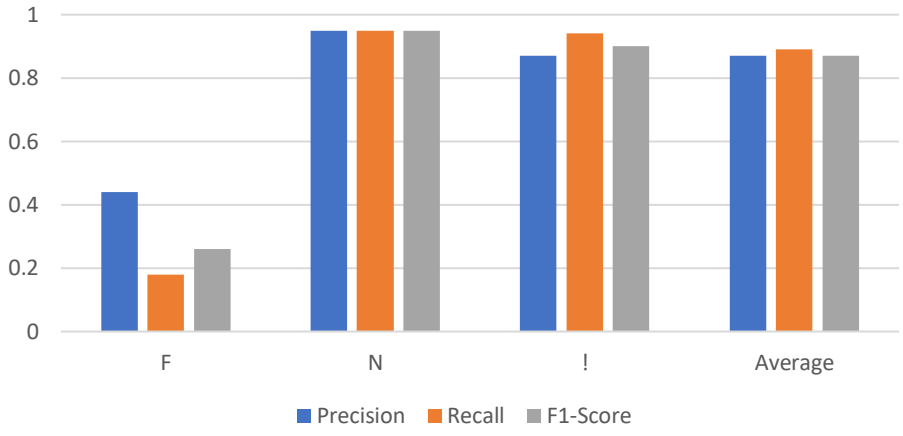
When the number of beat types was reduced to just three—normal and two of the most urgent beat types for treatment—the result was a much higher accuracy and lower runtime. Clearly, for sparse data inputs, the number of beats creates a tradeoff with the accuracy. The smaller dataset associated with less beat types does contribute to the overall runtime, but the data processes also play a role as well. The following is the confusion matrix for these three beat types:

$$\begin{bmatrix} \mathbf{4} & 1 & 17 \\ 1 & \mathbf{101} & 4 \\ 4 & 4 & \mathbf{137} \end{bmatrix}$$

The most urgent beat types are classified correctly at an accuracy of 88.6%, where the most common misclassification is confusing an F for ! beat classification, which are the anomalous and urgent types. Therefore, emergency care would be dispatched, since these are not misclassified as normal. Table IV contains the classification report summarizes the performance of the Naïve Bayes classifier for these urgent beat types. The decreased precision observed from the F category in part is due to the lack of available beat samples in this category but can be compensated in the future by creating more samples to be used. Using the Naïve Bayes for these three types results in the same memory usage as before (166 MB) but lower training time of 3ms and prediction time of .9ms. However, there is clear need to experiment with additional classifiers, since the Naïve Bayes is lacking in accuracy and precision.

FIGURE 4.1

Nave Bayes Classification Results for Urgent Beat Types



CHAPTER 5

Multilayer Perceptron Classifier Results

5.1 Classification of 14 Beat Types

Multilayer Perception (MLP), an artificial neural network, is another method that has similar memory and runtime to the Naïve Bayes but is less simplistic. For both the 14 beat and 3 beat types, a three-layer neural network was chosen, with 100 neurons per layer. For 14 beat types, the training time was 4.89 seconds and prediction time was 20ms. Compared to the Naïve Bayes, the training time was significantly higher for the MLP model. However, the memory usage was similar (116 MB), as well as the accuracy (62.5%). The model loss is shown in Figure 5a. The classification report summarizes the performance of the MLP classifier, which slightly outperforms the Naïve Bayes classifier, but has weaknesses for different beat types. For example, the MLP has difficulty classifying the f, L, J, and R beat types, whereas the Naïve Bayes could not classify the normal beat types correctly, along with F and j types.

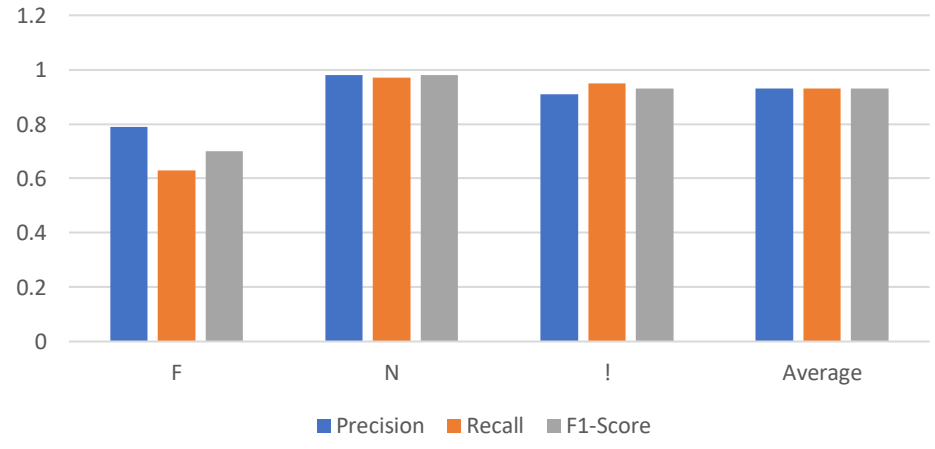
TABLE 5.1
MLP Classification Report for 14 Beat Types

Beat Type	Precision	Recall	F1-Score	Support
A	.89	.95	.92	302
N	.73	.35	.47	133
/	.73	.39	.51	148
f	0	0	0	19
V	.56	.46	.50	125
x	.93	.88	.91	558
F	.74	.75	.75	242
L	0	0	0	1
a	.8	.38	.51	229
J	0	0	0	25
R	0	0	0	2
j	.56	.08	.14	61
S	.40	.08	.14	49
e	.75	.23	.35	39
Average	.78	.64	.68	1933

5.2 Classification of Urgent Beat Types

The same urgent beat types as with Naïve Bayes were analyzed in the MLP model. The model performed with a 92.1% accuracy, which is 3% higher than the Naïve Bayes with the same memory requirement. The training and prediction time were only slightly higher than the Bayes, with the training time at 1.26 seconds and prediction time only 115ms. The most significant difference associated with less beat types was the decrease in model loss, as shown in comparison in Figure 5. When only three urgent beat types were analyzed, the loss was 40% lower than when using 14 beat types. Table 6 shows the classification report, which has high performance in all of the categories for these urgent beat types.

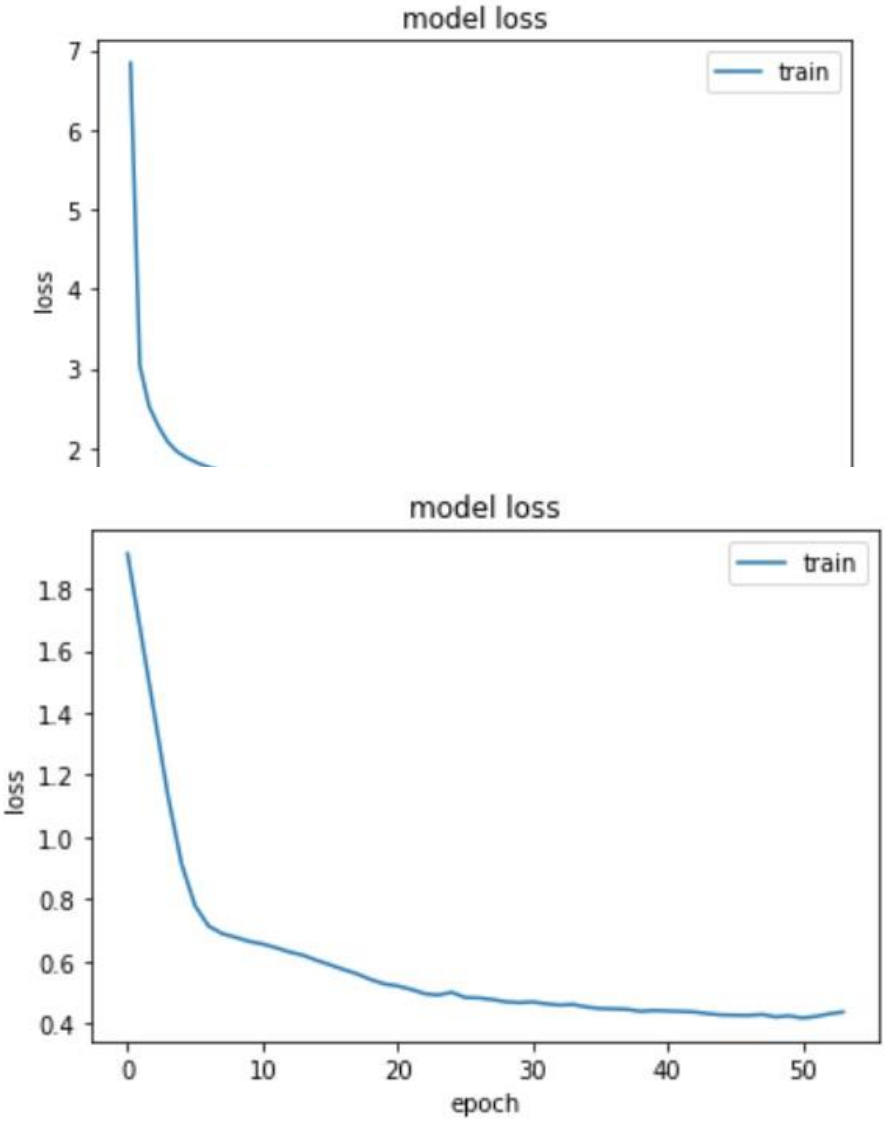
FIGURE 5.1
MLP Classification Results for Urgent Beat Types



As seen in the report, the MLP classifier has similar difficulty with the F beat type, which perhaps suggests similarities with the F and ! beat types that cause misclassification. However, this model outperforms the Naïve Bayes in accuracy and precision for these types, with the same memory and only slightly longer runtime.

FIGURE 5.2

Loss Comparisons for **(top)** 14 beat types and **(bottom)** urgent beat types

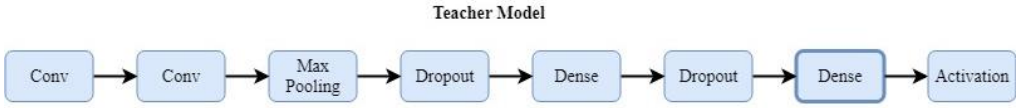


CHAPTER 6

Deep Neural Network Results

6.1 Deep Neural Network Teacher Model

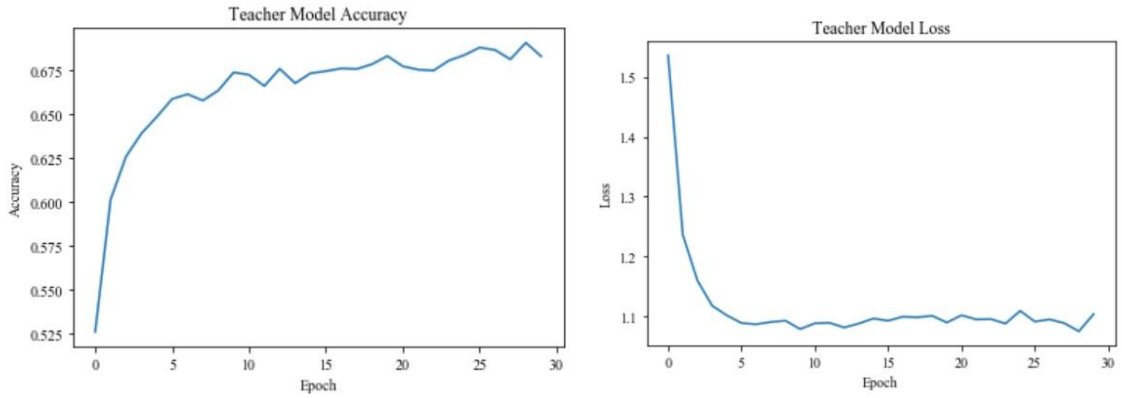
FIGURE 6.1



The previous methods involved simpler mathematics for the classification of the ECG beat types. However, these models fail to give a high accuracy in the case of 14 beat types. The initial full, or teacher, model was trained first on the 14 types and took 117 seconds for this task, which is considerably longer than the Naïve Bayes and MLP methods. The memory usage was also twice that of the other two. The accuracy after 30 epochs was 68.3%, which outperforms both of the previous models. The loss for this model is similar to that of the MLP classifier but will be improved

FIGURE 6.2

Teacher Model Accuracy and Loss
graphs for 14 beat types classified

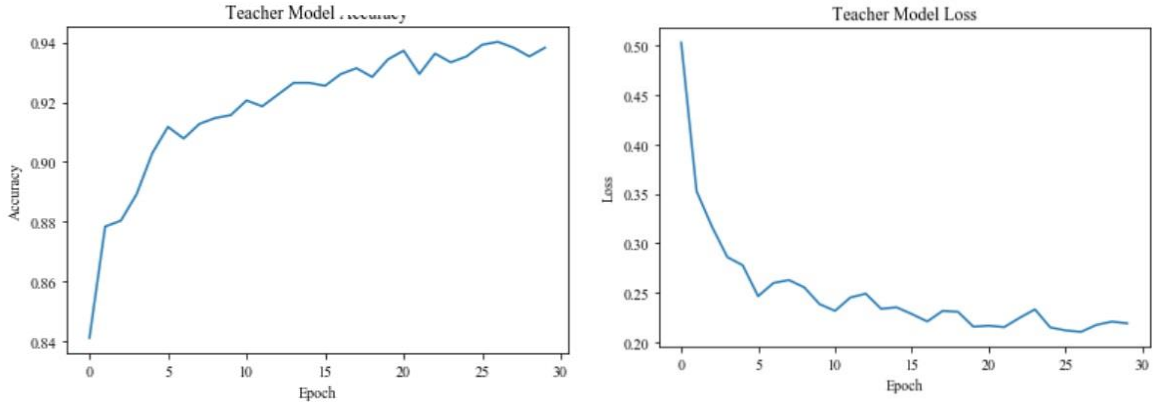


upon with the trained student model. Seen below in figure 6.1, the model loss decreases to 1.2 at the end of 15 epochs.

The deep neural network teacher model was then trained to distinguish between the beat types deemed most urgent for treatment (see Figure 6.2). This time, the training only took 42 seconds and 244 MB of memory. It achieved an accuracy of 94%, and after the 20th epoch, consistently achieved accuracies above 90%. Clearly a common trend among the neural network models is the increase in accuracy and precision with the inclusion of fewer beat types.

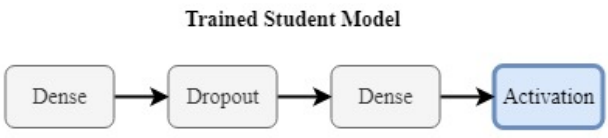
FIGURE 6.3

Teacher Model Accuracy and Loss
graphs for Urgent beat types
classified



6.2 Trained Student Model

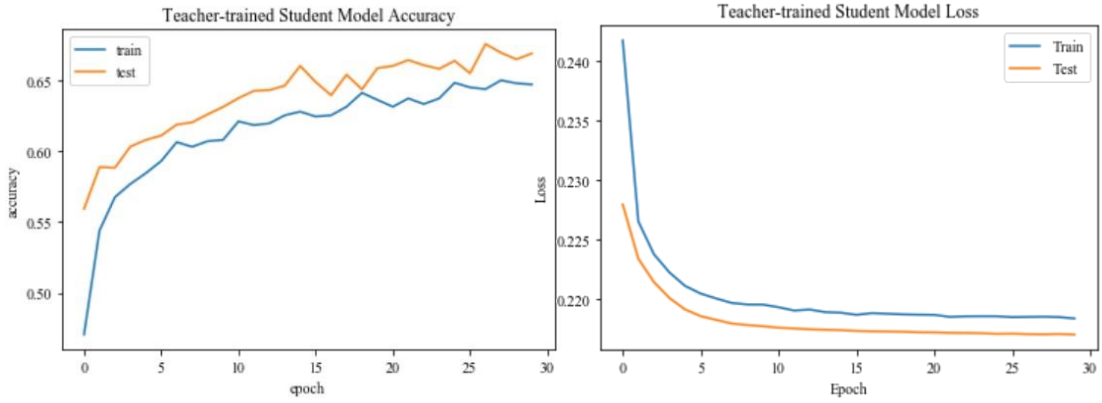
FIGURE 6.4



The student model obtained from training on the last layer of the teacher model's logits aims to keep the accuracy at a similar level while decreasing the loss and size of the teacher. After 30 epochs, the student was able to achieve greater than 65% accuracy, which is only slightly lower than the teacher model. The student model takes 115 seconds to train, since some inputs from the teacher are modified before inserting into the model, which increases the runtime. As seen from Figure 6.3, the model loss for the trained student model was slightly less than the teacher model.

FIGURE 6.5

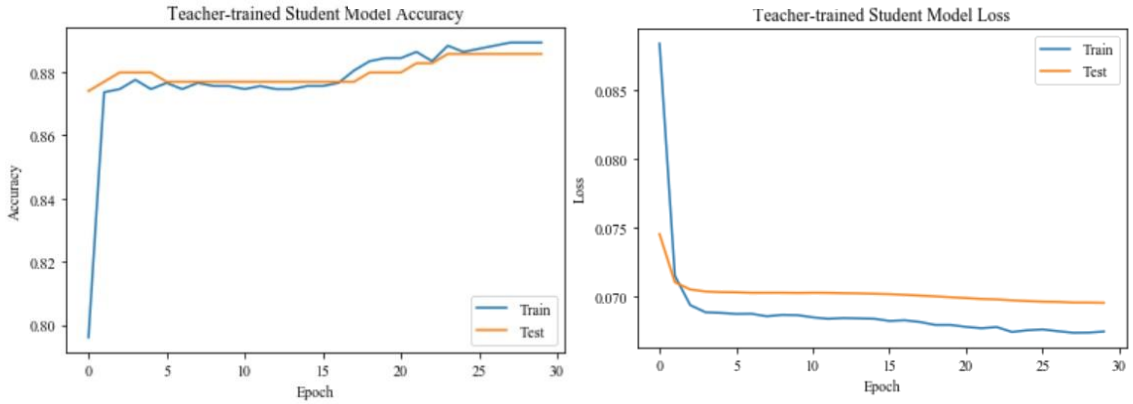
Trained Student Model Accuracy and
Loss graphs for 14 beat types classified



For the three urgent beat types, the student model trained on the teacher model performed similar to the 14-beat case. Again, the accuracy was slightly decreased (88.9%) compared to the teacher model after the 25th epoch and took 48 seconds to fully train. The loss for this case was much lower than the 14-beat classification task, and, in addition, has the lowest loss of all the deep neural network models trained. Because of the low loss and relatively high accuracy, this technique has the optimal traits needed for deployment in a medical edge device scenario out of all the deep learning models included.

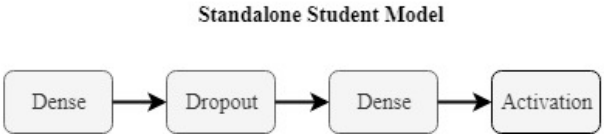
FIGURE 6.6

Trained Student Model Accuracy and
Loss graphs for Urgent beat types
classified



6.3 Standalone Student Model

FIGURE 6.7

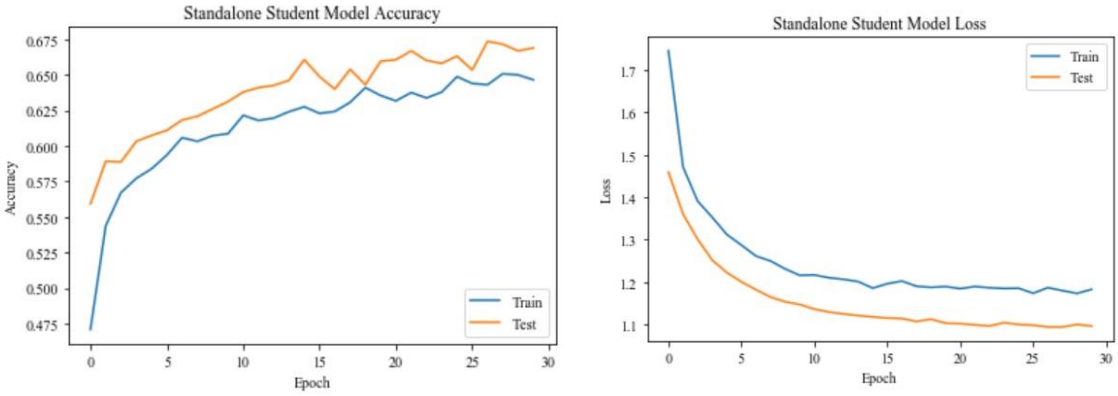


For comparison with the teacher-trained model, a standalone smaller model was created. This standalone model has the same structure as the trained student model, except for the activation layer which is separate from the teacher model. Thus, this entire model was trained independently from the teacher. This enables testing of the distillation compared to an initial simpler deep neural network. The training time for this model was 107 seconds, which is only 5 seconds longer than the previous deep neural network training times. For the 14 types, the accuracy and loss were

similar in the standalone model to the teacher model, which suggests that a smaller model may be appropriate for this specific ECG case.

FIGURE 6.8

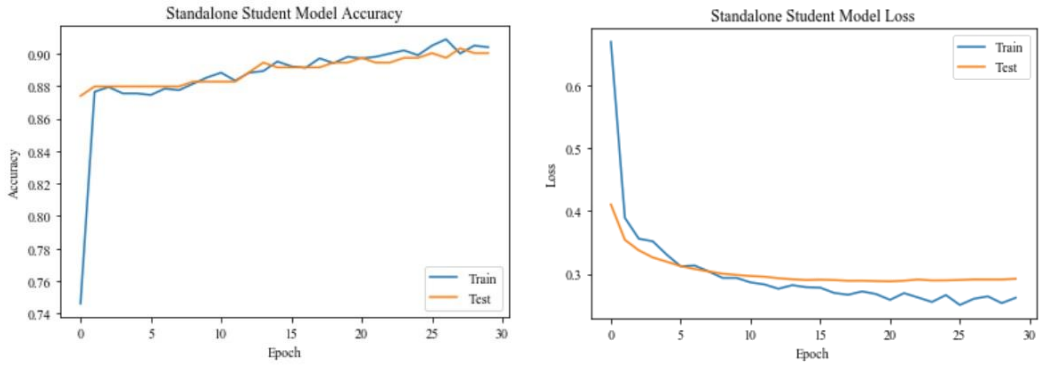
Standalone Student Model Accuracy
and Loss graphs for 14 beat types
classified



After decreasing the dataset to include only three beat types, the standalone student model achieved comparable results to the teacher-trained student model, however, the standalone had significant model loss as compared to the teacher-trained student model. Figure 6.6 illustrates this standalone model achieved a 90% accuracy and ran its training in 36.6 seconds, much lower than the other deep neural network models. So, although there was a lower runtime involved with this model, the accuracy and loss parameters are not as ideal as for the trained student model. For the next chapter, these and the other machine learning techniques will be compared in their energy efficiency, latency, and classification ability.

FIGURE 6.9

Standalone Student Model Accuracy
and Loss graphs for 3 beat types
classified



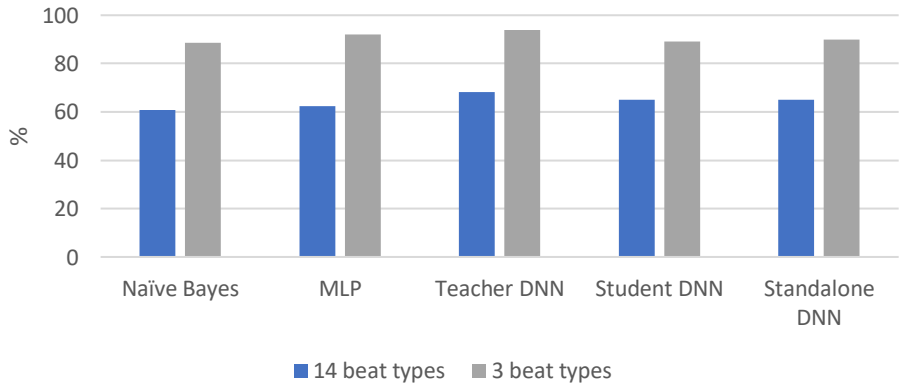
CHAPTER 7

Comparison of Machine Learning Techniques

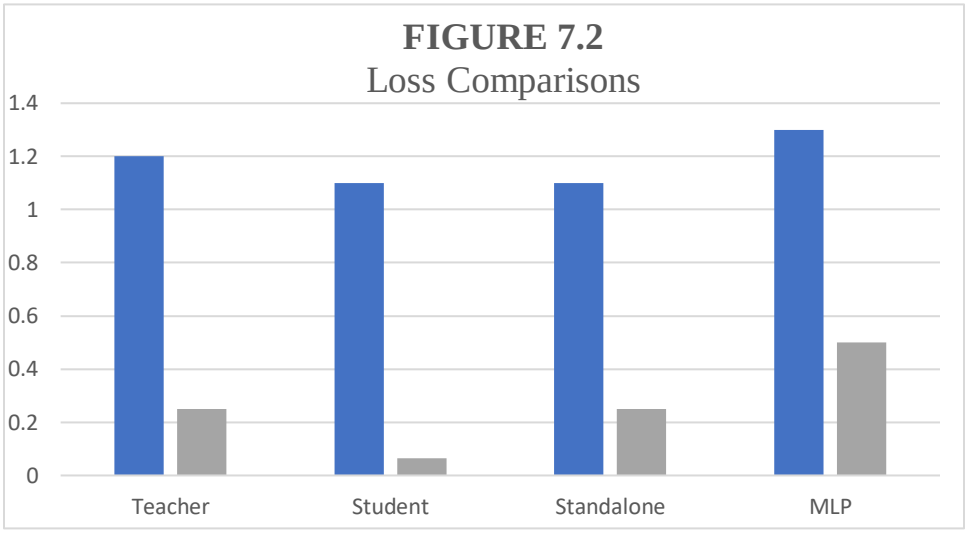
7.1 Accuracy Comparison

Each of the classifiers chosen achieve similar accuracy results when compared, however, the teacher DNN model has a slight edge over the others. This edge comes at a cost, since it has a high runtime and the highest memory requirement out of all the classifiers. The Naïve Bayes performs well, even though it is a relatively simple algorithms compared to the artificial neural network and deep networks. Accuracy is only one metric that is used to compare the performance, which is why precision and loss were included in the study. For the Naïve Bayes and MLP classifiers, the accuracy results were comparable to the precision, which is discussed in Chapters 4 and 5. For the neural network classifiers, loss comparisons are discussed in the next section.

FIGURE 7.1
Accuracy Comparison for Machine Learning Techniques



7.4 Loss Comparisons

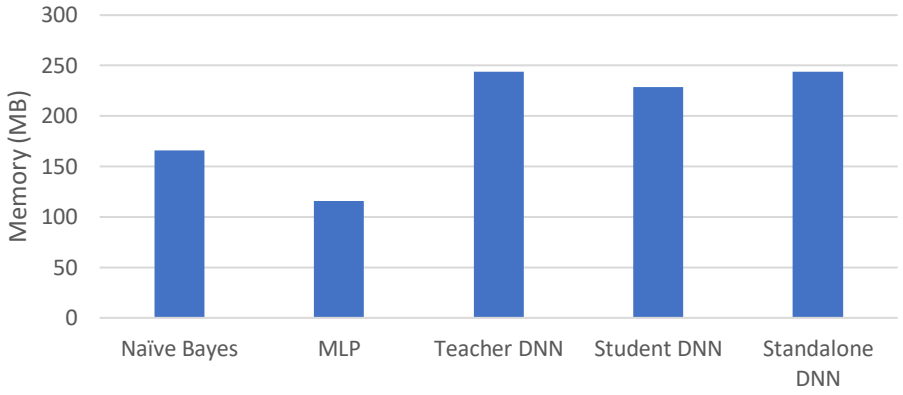


The neural networks tested—the artificial neural network (MLP) and three deep neural networks were all assessed for their model loss. As seen from Figure 7.2, the model loss for the MLP was much higher than the model losses for the deep neural networks. So, although the MLP model has a relatively high accuracy that compares to the other models, it has a high loss, which cannot be overlooked when choosing an ideal classification algorithm for an edge device.

7.2 Energy Efficiency Comparison

The third metric important for deployment on edge devices is the low CPU Usage requirement, a comparison of which is shown in Figure 7.3. Since the Naïve Bayes and MLP methods are not deep learning models, they consume less energy when performed on a dataset, especially the Naïve Bayes, which uses half of the memory that the deep neural networks require. Although the MLP does have a higher loss, it is extremely close to the teacher model in this respect and uses almost half as less energy as the teacher model.

FIGURE 7.3
Memory Requirements for Machine Learning Techniques

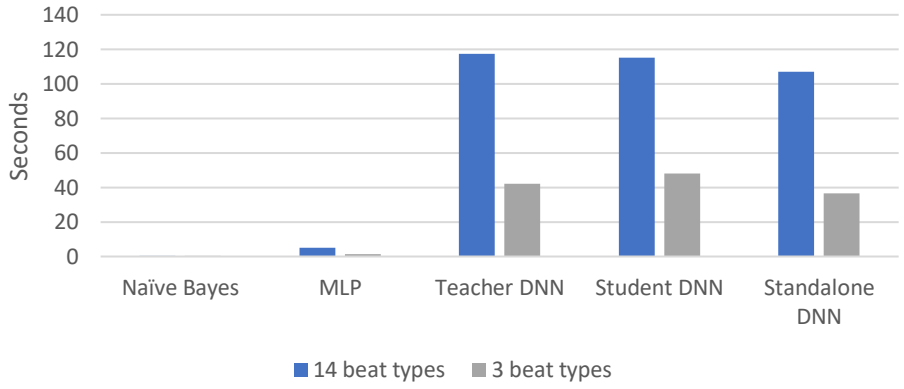


7.3 Latency Comparison

The final metric involved in an ideal edge algorithm selection is latency. There seems to be a direct correlation between latency and memory requirements, shown in Figure 7.4. The Naïve Bayes has nearly no runtime lag, while the deep neural networks, and, to an extent, MLP model, have higher latency of up to 117 seconds. However, if a dataset is run on a pre-trained model that can account for a variety of beat types in a variety of patients, the model might not have to be fitted to each patient's ECG. Further testing is required to test this hypothesis.

FIGURE 7.4

Training Latency Comparison of Machine Learning Techniques



CHAPTER 8

Conclusion and Future Work

This thesis provides a thorough survey of edge computing classification techniques which aided in a comparison on promising methods for deployment on a medical device at the edge of the network for low latency, energy efficient, and accurate diagnosis of raw ECG signals. It further provided an analysis for the best machine learning techniques to choose in a scenario involving sparse data to decrease the energy consumption on the part of the complete system.

The analysis of machine learning techniques included a state-of-the-art method called distilled knowledge learning, which uses a large teacher model for training a smaller, edge-friendly classifier. The results of the analysis proved the usefulness of the distilled neural network, which performs with the lowest loss among the classification techniques, with only a small drop in accuracy in comparison to the large model. This distilled model was compared to some simpler methods, namely Naïve Bayes and MLP. These smaller models are legitimate options for small devices with a larger number of features and low number of ECG beat types, which would help to decrease the loss which exists under this work's conditions. Adding more features does require more data pre-processing, which could add to the total runtime of the diagnosis program.

Based on the analysis, it is clear that using one data feature does limit the accuracy and precision of a system but works relatively well when the number of total classes is reduced. If more features are added, it is suggested that a smaller model such as Naïve Bayes is used for deployment in a case where low latency is the priority, above accuracy, such as an emergency diagnosis scenario. However, when all of the metrics are weighed, it is clear that the student model trained on the large deep neural network has the best tradeoff between accuracy, loss, and runtime.

Future work would include a study on best extracted features to use in a low latency scenario. For example, comparing the total runtimes of extraction plus the learning algorithm for several different

features. Adding additional data, such as heart rate or blood pressure could also add to the diagnosis aspect of a medical system. Another area of research would be to compare performance of machine learning algorithms best for specific beat types and then create specialty models to then distill into a neural network.

References

[1] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. 2012. Fog computing and its role in the internet of things. In Proceedings of the first edition of the MCC workshop on Mobile cloud computing (MCC '12). ACM, New York, NY, USA, 13-16.

[2] M. Hartmann, U. Hashmi, C. Ge, A. Imran, "Edge Computing in Smart Health Care Systems: Review, Challenges and Research Directions" (To be Submitted)

[3] K. Bhargava and S. Ivanov, "A fog computing approach for localization in WSN," 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Montreal, QC, 2017, pp. 1-7.

[4] Yu Cao, Songqing Chen, Peng Hou and D. Brown, "FAST: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation," 2015 IEEE International Conference on Networking, Architecture and Storage (NAS), Boston, MA, 2015, pp. 2-11.

[5] Rui Hu, Hieu Pham, Philipp Buluschek, and Daniel Gatica-Perez. 2017. Elderly People Living Alone: Detecting Home Visits with Ambient and Wearable Sensing. In Proceedings of the 2nd International Workshop on Multimedia for Personal Health and Health Care (MMHealth '17). ACM, New York, NY, USA, 85-88.

[6] Bhatia, M. & Sood, S.K. Mobile Netw Appl (2018). Springer.

[7] P. Verma and S. K. Sood, "Fog Assisted-IoT Enabled Patient Health Monitoring in Smart Homes," in IEEE Internet of Things Journal, vol. 5, no. 3, pp. 1789-1796, June 2018.

[8] S. S. Bhunia, S. K. Dhar and N. Mukherjee, "iHealth: A fuzzy approach for provisioning intelligent health-care system in smart city," 2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Larnaca, 2014, pp. 187-193.

[9] S. K. Sood and I. Mahajan, "A Fog-Based Healthcare Framework for Chikungunya," in IEEE Internet of Things Journal, vol. 5, no. 2, pp. 794-801, April 2018.

[10] D. C. Yacchirema, D. Sarabia-Jácome, C. E. Palau and M. Esteve, "A Smart System for Sleep Monitoring by Integrating IoT With Big Data Analytics," in IEEE Access, vol. 6, pp. 35988-36001, 2018.

[11] M. Hosseini, T. X. Tran, D. Pompili, K. Elisevich and H. Soltanian-Zadeh, "Deep Learning with Edge Computing for Localization of Epileptogenicity Using Multimodal rs-fMRI and

EEG Big Data," 2017 IEEE International Conference on Autonomic Computing (ICAC), Columbus, OH, 2017, pp. 83-92.

[12] Ahmed Nait Aicha, Gwenn Englebienne, and Ben Kröse. 2014. Modeling visit behaviour in smart homes using unsupervised learning. In Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication (UbiComp '14 Adjunct). ACM, New York, NY, USA, 1193-1200.

[13] J. Rodriguez, A. Goni and A. Illarramendi, "Real-time classification of ECGs on a PDA," in IEEE Transactions on Information Technology in Biomedicine, vol. 9, no. 1, pp. 23-34, March 2005.

[14] Daniel Castro, Steven Hickson, Vinay Bettadapura, Edison Thomaz, Gregory Abowd, Henrik Christensen, and Irfan Essa. 2015. Predicting daily activities from egocentric images using deep learning. In Proceedings of the 2015 ACM International Symposium on Wearable Computers (ISWC '15). ACM, New York, NY, USA, 75-82.

[15] H. Dubey, J. Yang, N. Constant, A. M. Amiri, Q. Yang, K. Makodiya, “Fog Data: Enhancing Telehealth Big Data Through Fog Computing,” Proceedings of the ASE Big Data & Social Informatics 2015, ACM, NY.

[16] D. Borthakur, H. Dubey, N. Constant, L. Mahler and K. Mankodiya, "Smart fog: Fog computing framework for unsupervised clustering analytics in wearable Internet of Things," 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Montreal, QC, 2017, pp. 472-476.

[17] "Electrocardiogram (ECG or EKG)", www.heart.org, 2018. [Online]. Available: http://www.heart.org/en/health-topics/heart-attack/diagnosing-a-heart-attack/electrocardiogram-ecg-or-ekg. [Accessed: 28- Sep- 2018].

[18] Orestis Akrivopoulos, Dimitrios Amaxilatis, Athanasios Antoniou, and Ioannis Chatzigiannakis. 2017. Design and Evaluation of a Person-Centric Heart Monitoring System over Fog Computing Infrastructure. In Proceedings of the First International Workshop on Human-centered Sensing, Networking, and Systems (HumanSys'17), Rasit Eskicioglu (Ed.). ACM, New York, NY, USA, 25-30.

[19] S. S. Bhunia, S. K. Dhar and N. Mukherjee, "iHealth: A fuzzy approach for provisioning intelligent health-care system in smart city," 2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Larnaca, 2014, pp. 187-193.

[20] C. Wang, Q. Wang and S. Shi, "A distributed wireless body area network for medical supervision," 2012 IEEE International Instrumentation and Measurement Technology Conference Proceedings, Graz, 2012, pp. 2612-2616.

[21] Z. Lv, F. Xia, G. Wu, L. Yao and Z. Chen, "iCare: A Mobile Health Monitoring System for the Elderly," 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing, Hangzhou, 2010, pp. 699-705.

[22] Iman Azimi, Arman Anzanpour, Amir M. Rahmani, Tapio Pahikkala, Marco Levorato, Pasi Liljeberg, and Nikil Dutt. 2017. HiCH: Hierarchical Fog-Assisted Computing Architecture for Healthcare IoT. ACM Trans. Embed. Comput. Syst. 16, 5s, Article 174 (September 2017), 20 pages.

[23] J. Jusak, H. Pratikno and V. H. Putra, "Internet of Medical Things for cardiac monitoring: Paving the way to 5G mobile networks," 2016 IEEE International Conference on Communication, Networks and Satellite (COMNETSAT), Surabaya, 2016, pp. 75-79.

[24] Moody GB, Mark RG. The impact of the MIT-BIH Arrhythmia Database. *IEEE Eng in Med and Biol* 20(3):45-50 (May-June 2001).

[25] Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng C-K, Stanley HE. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. Circulation 101(23):e215-e220 [Circulation Electronic Pages; http://circ.ahajournals.org/content/101/23/e215.full]; 2000 (June 13).

[26] G. James, D. Witten, T. Hastie, R. Tibshirani. *Introduction to Statistical Learning with Applications in R*, 1st ed. 2013.

[27] K.P. Murphy. *Machine Learning: A Probabilistic Perspective*, 1st ed. Massachusetts Institute of Technology, 2012.

[28] Metrics to Evaluate Machine Learning Algorithms in Python https://machinelearningmastery.com/metrics-evaluate-machine-learning-algorithms-python/

[29] ML Cheatsheet “Loss Function” https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html

[30] W. Koehrsen. “Beyond Accuracy: Precision and Recall.” [online] *Towards Data Science,* 2018. Available: https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c [Accessed 1 Nov 2018]

[31] H. Zhang (2004). The optimality of Naive Bayes. Proc. FLAIRS.

[32] Scikit-learn.org. *1.9. Naive Bayes — scikit-learn 0.20.1 documentation.* [online] Available: https://scikit-learn.org/stable/modules/naive_bayes.html [Accessed 1 Nov. 2018].

[33] A. Sakryukin.. “Under the Hood of Neural Networks. Part 1: Fully Connected.” [online] *Towards Data Science*, 2018. Available: https://towardsdatascience.com/under-the-hood-of-neural-networks-part-1-fully-connected-5223b7f78528

[34] U. Upadhyay. “Knowledge Distillation.” [online] *Towards Data* Science, 2018. Available: https://medium.com/neural-machines/knowledge-distillation-dc241d7c2322