Embedded Wavelet-Neural Network for ECG Analysis

Mahima Ghosh, Chi-Hsiang Wang, Yu-Sian Lin, and Wen-Ren Yang

***Abstract*— A signaling failure causes the heart to beat too quickly (tachycardia), too slowly (bradycardia), or irregularly (irregular beat). There have been a lot of classification methods based on feature extraction for classifying heartbeats and signals. While the limitations of current techniques include difficult signal collection, large feature sizes, and slow identification rates. In this work, a non-linear and non-stationary transform (Hilbert-Huang Transform) is coupled with a linear and non-stationary transform (Discrete Wavelet Transform) to distinguish between pathological and normal heartbeat signals. Using IoT devices, we gathered data from 10 subjects for a total of four minutes, or more than 2000 values. This project uses data gathered using laboratory-based IoT devices, in contrast to most projects in the field of heartbeat classifications that use the MIT-BIH dataset. Using IoT devices, we gathered data from 10 subjects for a total of more than 2000 values over the course of four minutes per subject. Unlike most projects in the field of heartbeat classification that use the MIT-BIH dataset, this one collects data using laboratory-based IoT devices. The proposed model was implemented in three stages: signal preprocessing, neural networks, and Arduino implementation. The MIT-BIH dataset was reviewed and evaluated using both convoluted neural networks (CNN) and artificial neural networks (ANN). The same model was then applied to the data that had been gathered for our dataset in order to conduct a similar analysis and validation for the purpose of evaluation. The trained model was then implemented on the Arduino Nano 33 BLE Sense. The model was transferred from the cloud to an embedded system for ECG. The implementation on an embedded system enables real-time ECG monitoring and analysis, making it a portable and cost-effective solution for healthcare applications. This approach also has the potential to improve patient outcomes by providing early detection of cardiac abnormalities.**

***Index Terms*— Adaptive signal processing, Artificial Neural Networks (ANN), Biomedical communication, Convoluted Neural Network (CNN), Discrete Wavelet Transform (DWT), ECG, Embedded systems, Hilbert Transform (HT), IIR Filters, Internet of Things (IoT), Notch Filter, R-Peaks Detection, Welch periodogram**

I. INTRODUCTION

In cardiac terms, a heartbeat is defined as a rhythmic contraction and relaxation of the heart muscles. A heart arrhythmia is an irregular heartbeat. Normal heart rhythm is often called normal sinus rhythm that ranges from 60 to 100 beats per minute in adults. The AV node (atrioventricular node) is a cluster of cells in the center of the heart between the atria and ventricles and acts like a gate that slows the electrical signal before it enters the ventricles. The various nodes in an ECG wave (fig 1) are the P,Q,R,S, and T. The P node symbolizes the atrial contraction when contractions in the atrial muscles increases pressure for the mitral valve to open. The QRS region depicts the ventricular depolarization (activation) and the ventricular repolarization is the region between the QRS complex to the end of the U wave.

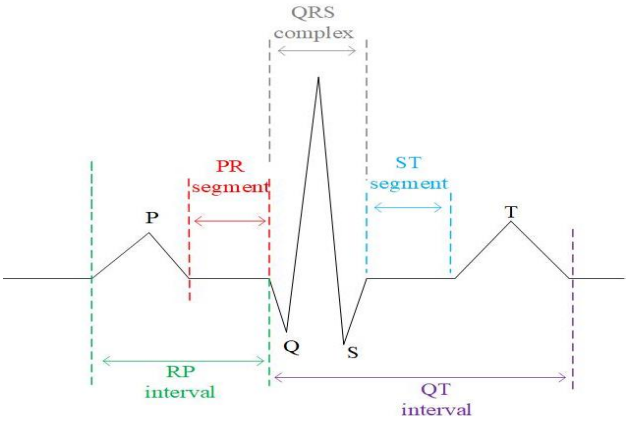


Fig.1. Electrical activity of the heart depicting the QRS complex

The raw ECG signals were collected (fig 8) using the Internet

The manuscript no. JBHI-00721-2023 submitted on 16 March 2023 to IEEE Journal of Biomedical and Health Informatics (JBHI). This research is sponsored by Artificial Neural Network Applications, TEEP (Taiwan Experience Education Program), Taiwan Ministry of Education, project no. 1112506208. Corresponding author: Mahima Ghosh

First A. Mahima Ghosh is a B. Tech-Computer Science and Engineering (CSE) student at Presidency University, Bangalore, India. She pursued her internship at National Changhua University of Education (NCUE), Taiwan, for a period of six months in the field of biomedical signal processing. She will be joining Imperial College London for her masters in Human and Biological Robotics this fall. (e-mail: mahimaghoshofficial@gmail.com).

Second B. Chi-Hsiang Wang and Third C. Yu-Sian Lin are students in the department of electrical engineering at National Changhua University (NCUE), Taiwan (email: S0952045@gm.ncue.edu.tw, s0952032@gm.ncue.edu.tw)

Fourth D. Wen-Ren Yang is an associate professor at the National Changhua University of Education (NCUE), Taiwan in the department of electrical engineering. He supervised the entire project and the team(e-mail: wry87c@cc.ncue.edu.tw)

This project uses ECG data collected from 10 volunteers at National Changhua University of Education (NCUE), Taiwan. All the proceedings and collection of data has been done according to the IEEE consent guidelines.

of Things (IoT) devices-Wemos D1 mini which is a microcontroller development board in the ESP-8266EX family having a flash of 4M bytes, AD8232 ECG-sensor (fig 8(a)) was used for measuring the electrical activity of the heart, a power source of 3.3V, a 3-pin electrode patch and jumper cables for connecting the modules. The software used was Arduino IDE. The electrodes were placed in specific positions as shown in (fig 2). The red electrode was placed on the right arm (RA), the yellow electrode was placed on the left arm(LA) and on the left leg (LL) was the green electrode. In this work we will study the computerized analysis of the electrocardiogram and in real time. The data was collected from 10 subjects, each for 240 seconds using IoT devices. The experiment table was set up in a laboratory at room temperature. The potential subject was made aware of the procedures of the experiment and they verbally consented. The subject was seated in front of a laptop. The environment was set up with a D1 mini which was the microcontroller and acted as the interface between windows and the AD8232 ECG--sensor. The 3-pin electrodes patch were attached with reference to fig 2 for an optimizing result. The timer was set for 4 minutes (240 seconds). During the time, the subject was allowed to do any activity of their choice such as reading a book, watching movie, or engaging in an intellectual debate. This helped us record the electrical activity of the heart, taking into consideration the activities which affected it in real time. The reason for the 4-min time limit was due to enormous amounts of data that was being generated every minute. Hence, the 4-min time limit seemed to be adequate for the data collected. It is important to note that any other electrical activity or interference was resolved before proceeding with the collection of data. The data collected as show in fig 8 was now converted from electrical signals to data and then transferred to a .csv format for further evaluation that will be discussed at length.

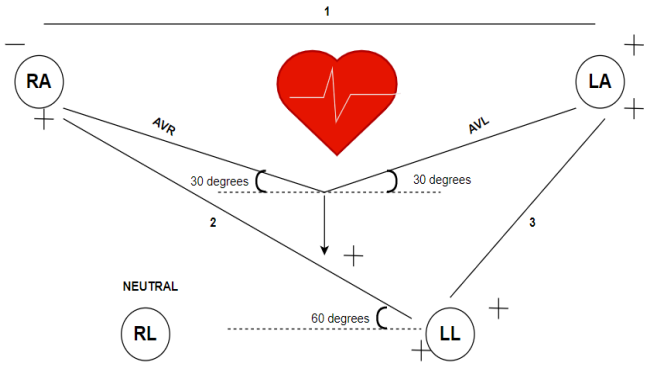


Fig.2. Specific placement of the 3-pin electrode patch for accurate and optimized ECG reading

II. System Diagram

The complete system plan involved the collection of data using IoT devices by filtering the signal, followed by data conversion, and then passing the output dataset into a CNN model. The trained model was then converted to a .h file (fig 5) and uploaded onto the Arduino Nano 33 BLE Sense (fig 6(b)). Since IoT promises the dream of anytime, and anywhere, we used TensorFlow lite for Arduino to achieve real time analysis of ECG. signals. This system is achieved in 3 major segments (fig 3)-

1.Signal Processing and analysis of the ECG
2.Training using neural networks
3. Implementation using Arduino

In course of the paper, we will elaborate on the major segments that helped achieve the wavelet-neural network in an embedded system. The system diagram consists of the workflow from the collection of raw ECG signal to the

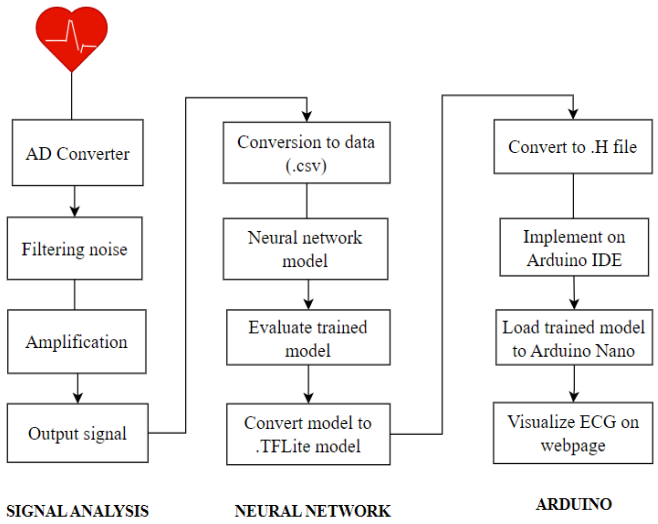


Fig. 3. System proposed workflow diagram

implementation of an embedded system using Arduino device- Arduino Nano 33 BLE Sense. The diagram has three phrases/segments of signal processing, neural network and implementation using Arduino.

III. Implementation

The implementation diagram(fig 4) depicts the third phrase of our proposed plan for implementation using Arduino. The three phrases of the project will now be discussed at length below. It is important to keep in mind that the data used for signal pre-processing and analysis in this paper was data collected using laboratory-based IoT devices and hence, there is a slight difference compared to medically acquired ECG signals (like the MIT-BIH dataset).

A. Model Conversion

The implementation part of the project, after the signal analysis and neural networks, was to load the trained model onto a IoT device that can be further used in real life implementation of analysing and an easy assessment of the electrical activity of the heart. This part involved the conversion of the keras model to a c code in to implement the same onto the Arduino board [14][15][16][17][18].

This conversion from a keras model to a .h file, that is compatible with Arduino IDE, involves the process of converting the file from keras to .TFLite model and then to

.h file. In the runtime, the model is viewed as a C code (fig 5). In general, any model saved as TFLite is converted to FlatBuffer at runtime. The converted model (.h file) is now uploaded onto the Arduino IDE. Due to limitations in the field of neural networks on microcontrollers, there are a limited boards by Arduino which allows us to load a fully trained CNN model onto the microcontroller. One such board is the Arduino Nano 33 BLE Sense[17].

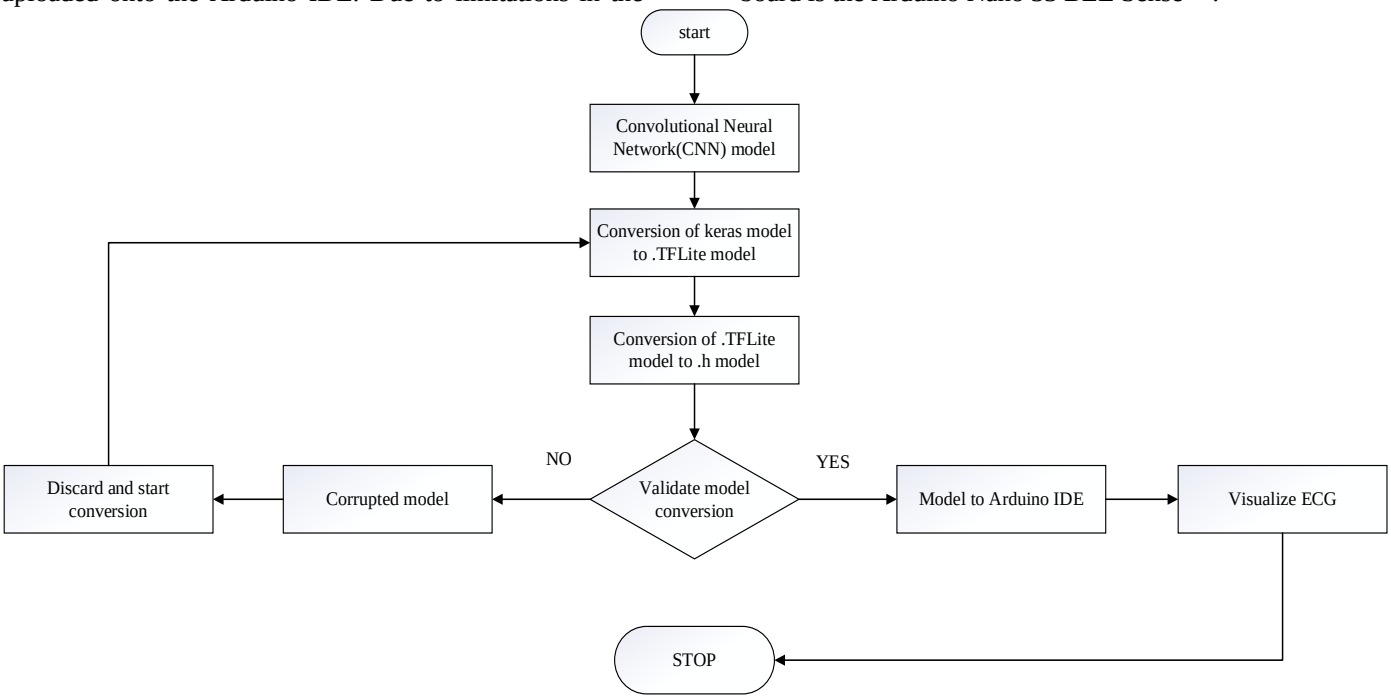
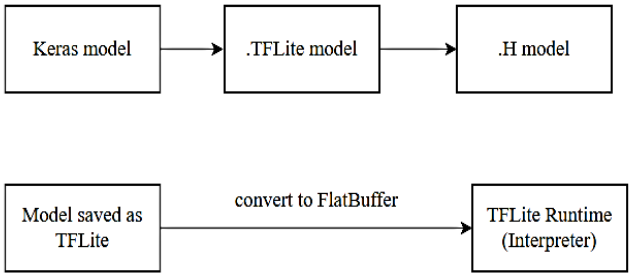


Fig. 4. Implementation using Arduino flow diagram

The Nano 33 (fig 6(b)) has a 32-bit ARM cortex M4 microcontroller with a CPU flash memory of 1MB, along with color, brightness, proximity and gesture sensor; a digital microphone and vibration sensor among others (table 1). Thus, making it a superior choice for loading CNN model for heart rate monitor.

In TensorFlow 2.0, it order to convert a .h5 file to .tflite, the saved keras file must uploaded and then converted using TFLiteConverter first. Then the final conversion from keras to .tflite model is successfully completed. Proceeding further, this .tflite model must now be made Arduino compatible. It is possible to directly build a neural network using C. But the is a very long process and has many ambiguities as the data to be uploaded on Arduino has limitations. While TensorFlow stores



Model saved as FlatBuffer

Fig.5. Conversion of keras model to .h file for Arduino implementation

models in the Protocol Buffer format, tflite stores models in the

Fig. 6. (a) AD8232 ECG Sensor

Fig. 6. (b) Arduino Nano 33 BLE Sense

Flat Buffer format, hence saving memory and enhancing a faster runtime.

TABLE 1
SPECIFICATIONS OF ARDUINO NANO 33 BLE SENSE

Specifications	Value
Processor	nRF52840
Microcontroller	Cortex M4
Clock Speed	64 Mz
SRAM	264 kB

B. Load the converted model (.h file) to Arduino IDE

The fully converted model is now loaded to the Arduino IDE. The Arduino_TensorFlowLite library is used for implementing machine learning on Arduino. This library must be manually installed from the machine learning on microcontrollers website[28]. It is necessary to include the sub-libraries present in the Arduino_TensorFlowLite. The necessary operations for this program are: all_ops_resolver.h: interpreter to run the model; micro_error_reporter.h: outputs debug information; micro_interpreter.h: code to load and run models; and schema_generated.h :contains the schema for the TensorFlow Lite FlatBuffer model file format [16][17][18]

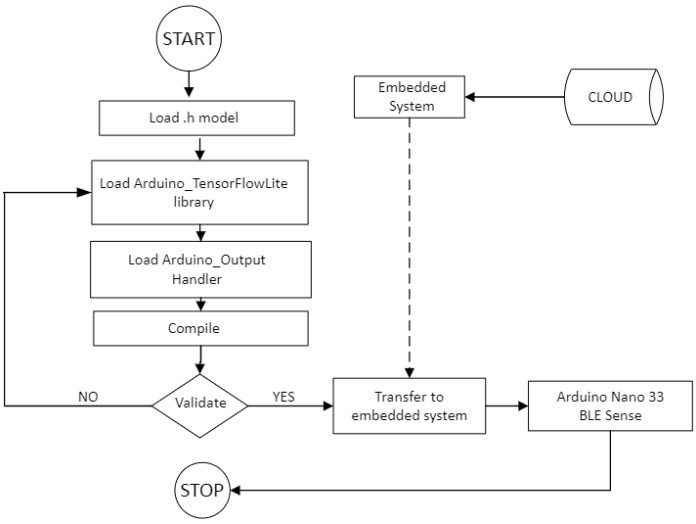


Fig.7. Loading the model to Arduino (transfer from cloud to embedded system)

The AllOpsResolver [28] operation is a necessary operation that loads all the different layers present in the CNN model. For large CNN models, it is advised to load only the necessary layers present in the CNN model because the AllOpsResolver uses a lot of memory. The different layers in the CNN model can be visualized using the netron app available on GitHub. The CNN model in this project included three Conv2D layers, three layers. The AllOpsResolver was used only for the layers present in our CNN model.

IV. Data Collection

The raw ECG data used for this project was procured from ten volunteers at the National Changhua University of Education (NCUE), Taiwan. The laboratory was set up using Wemos D1 mini- that is a microcontroller to collect the ECG data using Arduino IDE and Flag Blocks; AD8323 ECG Sensor- main equipment to sense the ECG; and jumper cables to connect to a 3.3V power source. Th entire setup then hooked to a PC (windows-8). The reading was taken using RNN on Wemos D1 mini. The waves forms were visualized on the serial plotter of the IDE (fig-8). The dataset collection was based on voluntary application of interested candidates. The volunteers were made aware of any risks (no risks involved) and the protection of their data. The candidate was seated in front of the system with the 3-pin electrodes attached in accordance to (fig-2). The participants were allowed to engage in any activity of their choice such as reading a book, watching a movie, or just relaxing, for a period of 4 minutes each. This helped us collect ECG data in real-time, keeping in mind the various variances in data if any.

The now collected data (in form of signals) was then converted to data (.csv format). This data was used for signal processing and analysis which will be discussed at length in the theory section.

V. Theory

A. Signal Processing

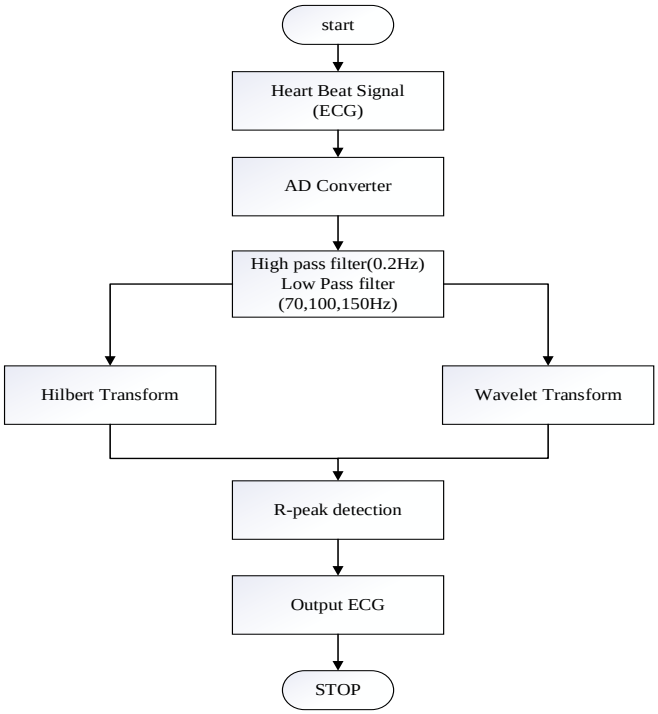


Fig. 9. The signal processing and analysis flow diagram used for the ECG signal

The signal processing focuses on the visualization of the "filtered signal" using the welch periodogram and the periodogram. We deploy the welch periodogram method (1) in this study [5][6]. In order to get rid of the noise as much as possible, mathematically, we must do averaging of the signal nodes.

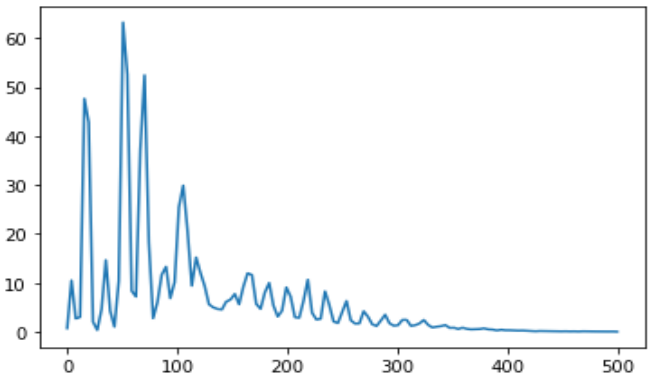


Fig.10. Welch periodogram for a stochastic ECG signal that has time varying mean and variance

Welch method helps us by slicing the original signal into several pieces and averages the total spectra of these. Unlike Fast Fourier Transform (FFT), where it converts the signals that are represented as a function of time to a function of frequency, we tend to overlook the fact that the signal is a sample of the stochastic process. The FFT works best for deterministic signal (sinα1 t + sinα2 t) and not a stochastic process (ECG). A stochastic process defines as a system that has observations at certain times and the outcome at each time is a random variable. Hence, by shortening the signals to which FFT is applied, and by averaging, the spectra are low-passed and thus the peaks are not so narrow [6]. When handling unequal variances, Welch's method (fig 10) is better as it averages the estimates of various segments of the signal to produce an estimate of the power spectral density. The Welch's method is carried out by the time signal into successive blocks and then averaging by forming the periodogram.

$$x_i(n) = x(iD + n)w(n) \quad (1)$$
$$\text{with } 0 \le n \le M-1, 0 \le i \le k-1$$

where, w(n)- window having length M

D- offset distance

K- no. of segments x[n] is divided into

TABLE 2

PARAMETERS FOR THE NOTCH FILTER USED

Variable	Type	Description
x	array	A one-dimensional array containing the signal to be filtered.
w_0	float	A scalar that must satisfy 0 < w0 < 1, w0 is the normalized frequency to remove from the filtered signal (w0 = 1 corresponds to half of the sample frequency).
δ	float	increment parameter. By default, δ=0.00125.

Next, the application of notch filter at 60Hz (fig 10) for the removal of unnecessary noise present in the signal. The same was done using the adaptive filter. The design of the IIR notch digital filter parameters: (table 2). The above (table 2) returns the following: b, a : ndarray, ndarray; Numerator (`b`) and denominator (`a`) polynomials of the IIR filter. The quality factor (Q factor) for the filter is related to the filter bandwidth by Q = w0/bw (for the default value of `ab`)[3]. The application of adaptive non-linear filter (fig 11) to signal [x] in order to remove the normalized frequency w0 parameters:

TABLE 3

PARAMETERS FOR THE ADAPTIVE FILTER USED

Variable	Type	Description
w_0	float	A scalar that must satisfy 0 < w0 < 1. For notch filters, w0 is the normalized frequency to remove from the filtered signal (w0 = 1 corresponds to half of the sample frequency)
b_w	float	Filter bandwidth.
a_b	float	Attenuation in decibels. By default, `ab` is 3. So, the attenuation is -3 dB or -10*log10(1/2) dB.

The above (table 3) returns the following: y : array_like. The filtered output, an array with the same shape as x.[4]

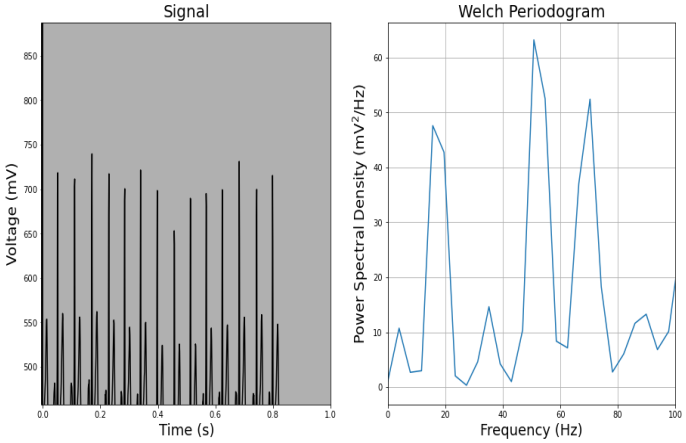


Fig.11. Application of notch filter on the ECG signal

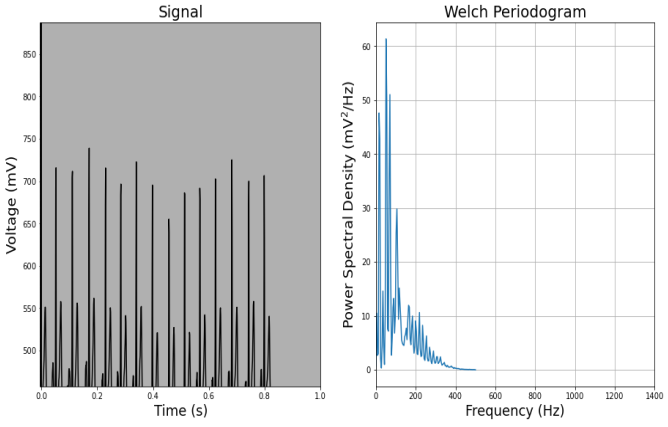


Fig. 12. Application of adaptive filter on the ECG signal

On comparison (fig 10,11), the result obtained by simulation seem to indicate that the use of notch filter can obtain a greater attenuation of the noise but according to [27] the adaptive filter would provide better transient responses. The low frequency fluctuation (baseline wander) is set using the high pass IIR filters. This was followed by the removal of low frequency.

B. Frequency Response

A high pass filter is a filter that attenuates low frequencies. As mentioned in the previous section, the low frequency fluctuations that contaminate the electrocardiogram. To filter these fluctuations an elliptical order filter will be used, it has the following specifications in table 4 [12] . It is important to note that bidirectional filtering (filtfilt function) was used. The filtfilt function performs zero-phrase digital filtering by processing the input signal in both forward and reverse directions.

$$y = filtfilt(b, a, x) \quad (2)$$

Where, x is the input signal.

TABLE 4
PARAMETERS FOR THE HIGH-PASS FILTER USED

Rejection Band	Frequencies below	0.4Hz
Pass Band	Frequencies above	0.2Hz
Attenuation in the rejection band	-	-4dB
Ripple in the pass band	-	0.1dB

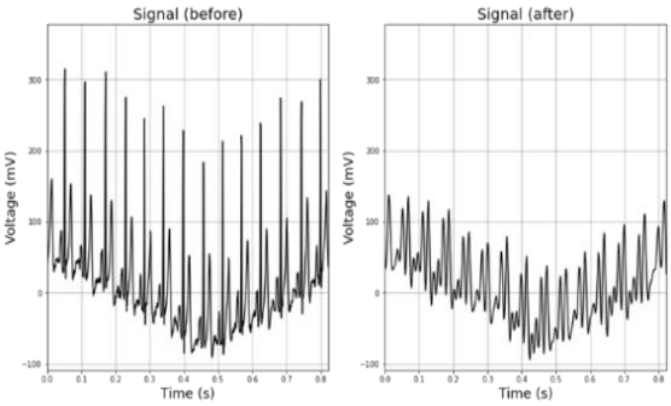


Fig. 13. Result of applying high pass filter to the signal. On the left side is the original signal and on the right side is the signal after all the lower frequencies have been removed. The pass band frequency is usually 0.4 Hz and rejection band of 0.8Hz.

This process helps preserve features exactly as they occur in the unfiltered signal. In an ECG signal, the QRS complex is the most important feature. The zero-phase filtering reduces noise as well as preserves the QRS complex. Both the filter() and filtfilt() remove high frequency noise but filter() has a significant phrase delay. This has two consequences: The attenuation obtained will be double that for which the filter was designed, since the filter is applied in the forward and reverse directions. There will be no phase distortion.

C. Low-pass Filter

To eliminate the high-frequency noise in the ECG signal, the application of a low-pass filter was studied and used. The IEC specification requires a bandwidth of for the electrocardiogram. For analysis purposes [12], we will compare the application of three different filters: 70 Hz (red), 100 Hz (green) and 150 Hz (black). Filtering, using each of three different cut-off frequencies, is shown below (fig 14). It is important to note that, again, bidirectional filtering was used to avoid phase distortion. Some comments on the signal filtered using the low-pass filter for the three specified cut-off frequencies: First, the lower the cut-off frequency, the greater the noise attenuation. It is easy to verify that the signal filtered with the cut-off frequency filter of 70Hz (in red) is significantly less "hairy" than the signal filtered with the filter of 150Hz (in black). Second, some signal characteristics of interest are attenuated when the cut-off frequency is less than 150Hz. For example, the peak of the R wave which is for the filtered signal with the filter of 150Hz (in black) has decreased to 1.36mV (a variation of approximately 3%) when we consider the other two signals in green and in red. In this case this effect is not significant, however, in other cases, components significant to the diagnosis may be hidden. Low-pass and high-pass filtering can be performed by a single band-pass filter. This was not done here because the objective was to analyse the different interference components in the ECG individually. The spectral content of the ECG is concentrated between 0.5Hz and 150Hz (fig 13). Note that in this frequency range, the gain grows almost linearly with frequency. In this way, the QRS complex, which has higher frequencies, will be enhanced while the other ECG waves are attenuated.

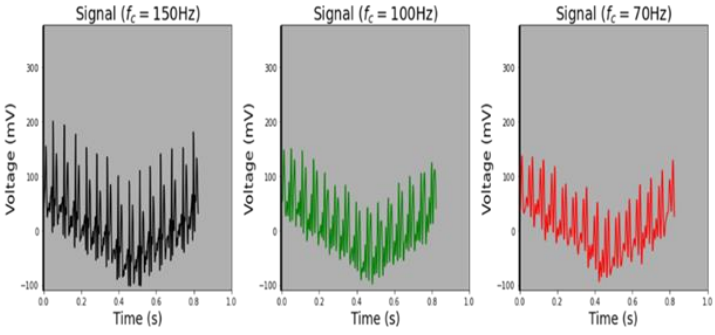


Fig. 14: Application of low pass filter for 70Hz, 100Hz and 150Hz respectively.

The spectral content of the ECG is concentrated between 0.5Hz and 150Hz (fig 14). Note that in this frequency range, the gain grows almost linearly with frequency. In this way, the QRS complex, which has higher frequencies, will be enhanced while the other ECG waves are attenuated.

D. Hilbert Transform

The Hilbert Transform is used to represent the band pass signals and to relate the gain and phase characteristics of the linear communication channels and the minimum phase type

filters [19][20] . The procedure for enhancing the QRS complex will be applied to a real electrocardiogram signal. Below are shown the sign x[n], the difference of this sign, d[n], and finally the result of the Hilbert transforms, y[n]. Note that for the signal y[n], the QRS complex is highlighted while the P wave is attenuated. This prevents the peak of the R wave from being confused with a high-amplitude P wave, for example:
Let a signal x(t) with Fourier transform X(ω). The Hilbert transform of x(t) is obtained by the convolution of x(t) and (1/πt), i.e.[20]

$$H(u)(t) = \frac{1}{\pi} p.v \int_{-\infty}^{+\infty} \frac{u(\tau)}{(t-\tau)}\, d\tau \qquad (3)$$

Where, p.v is Cauchy's principal value and the integral exists as a principal value.
Alternatively, by changing variables, the principal integral can we rewritten as:

$$H(u)(t) = \frac{2}{\pi} \lim_{\varepsilon \to 0} \int_{\varepsilon}^{\infty} \frac{[u(t+\tau)-u(t-\tau)]}{2\tau} d\tau \qquad (4)$$

According to the Bedrossian's theorem, the Hilbert transform of the product of a low-pass and a high-pass signal with non-overlapping spectra is given by the product of the low-pass signal and the Hilbert transform of the high-pass signal[8],

$$H\big(f_{LP}(t).f_{HP}(t)\big) = f_{LP}(t).H(f_{HP}(t)) \qquad (5)$$

Where, f_{HP} and f_{LP} are high pass and low pass signals respectively.

The Hilbert Transform can be understood in terms of a function pair f(x) and g(x):

$$F(x) = f(x) + ig(x) \qquad (6)$$

This is the boundary value of a homomorphic function[10]. Now, using the conjugate function:

$$u_a(t) \triangleq u(t) + iH(u)(t) \qquad (7)$$

This is known as the analytic representation of u(t).
According to[22],

$$u(t) = A\cos\left(ut + \emptyset_m(t)\right) \qquad (8)$$

This includes phrase modulation and frequency modulation. The instantaneous frequency is defined as:

$$\omega + \emptyset'(t) \qquad (9)$$

For a sufficiently large ω, compared to $\emptyset'$,

$$H(u)(t) \approx A.\sin\left(\omega t + \emptyset_m(t)\right) \qquad (10)$$

and,

$$u_a(t) \approx A.e^{i(\omega t + \emptyset_m t)} \qquad (11)$$

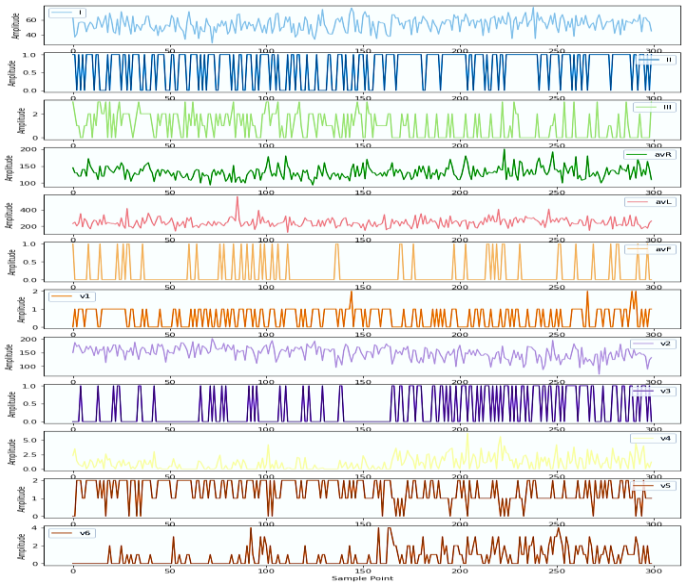


Fig. 15. Visualization of ECG using the 12 lead electrodes from MIT-BIH data

E. Discrete Wavelet Transform

Like FT, Discrete Wavelet Transform (DWT), decomposes signals into predefined elementary building blocks (atomic functions) called wavelets [3][4]. Wavelets are functions that are obtained by translation and dilatation of a predefined function called "mother wavelet." DWT can be represented as:

$$X(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} \varphi \left[\frac{t-b}{a}\right] x\,(t)dt \qquad (12)$$

Where, a is scaling and b is time shift factor.

The DWT of a particular signal (x) is first calculated by passing through a series of low pass filters with an impulse (δ(t)), producing a convolution:

$$y[n] = (x * g)[n] \qquad (13)$$

Where, g is the impulse response function of a dynamic system output

This further produces the equation:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]g[n-k] \qquad (14)$$

The signal is also simultaneously passed through a high pass filter. The output is a detailed coefficient (from the high-pass filter) and approximate coefficient (from the low pass filter) (fig 16,17). With respect to Nyquist's rule, since half the frequency samples have been removed, half the samples can be discarded.

$$y_{low}[n] = \sum_{-\infty}^{\infty} x[k]g[2n-k] \qquad (15)$$

$$y_{high}[n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-k] \qquad (16)$$

Where, g- new low pass filter
h- high pass filter with half cut-off frequency

Compared to Fourier Transform, the wavelet transform decomposes a function into sets of wavelets. As a result, this works best on electrocardiography (ECG) signals which have short intervals of characteristic collisions. Wavelets have two basic properties: Scale- defines frequency on how "squished" or "stretched" a wavelet is; Location- the position of the wavelet with respect to time.

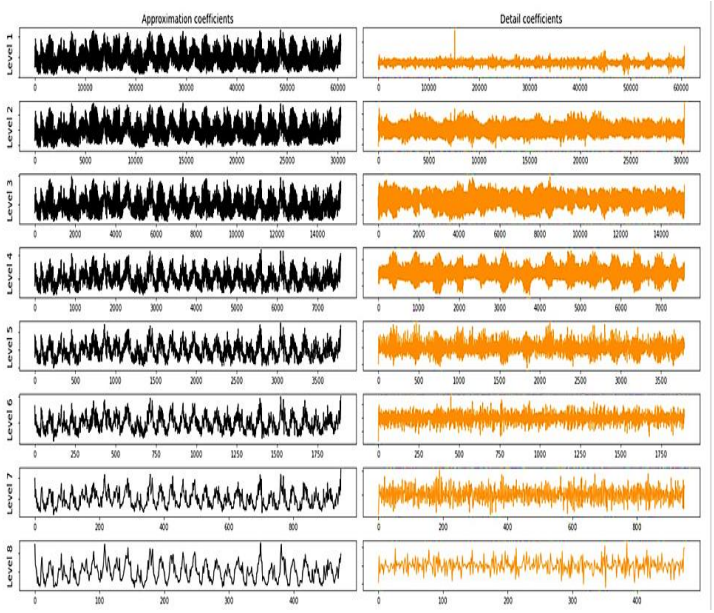


Fig. 16. Wavelet decomposition on the ECG signal at level 1

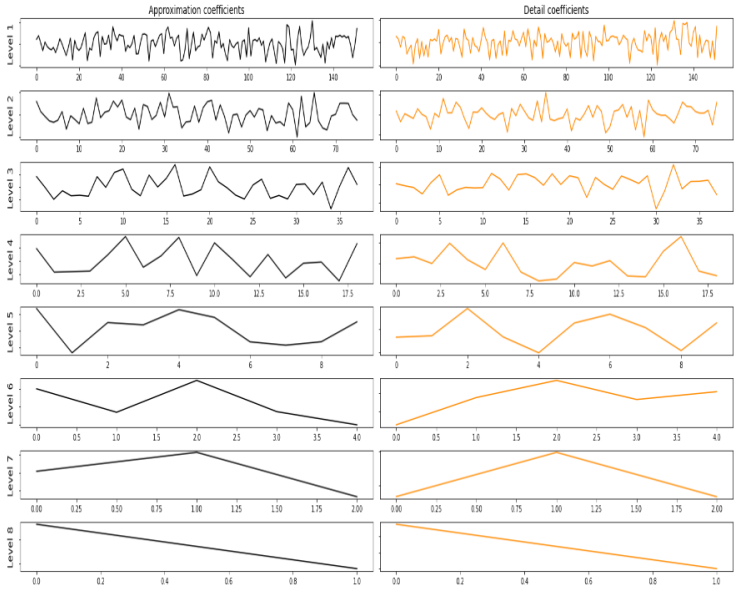


Fig. 17. Wavelet decomposition on the ECG signal of level 1 at level 2

The important part to remember when using wavelet transform on known, non-sinusoidal signals such as heartbeats, the application of matched wavelets can outperform Morlet analysis. The maximal overlap discrete wavelet transform (MODWT) helps in the process for locating the R-peaks present in the ECG signal. The symlet wavelet, which has near symmetrical properties, was used at 8 different levels for visualizing and understanding the application of wavelet transform on the ECG signal(fig 16,17). The presented discretization ensures that narrow wavelets with high frequency content are translated by small, and wide wavelets with low-frequency content by larger time steps. Consequently, DWT has better time resolution for high and better frequency resolution for low frequency components [7] .

F. Comparative study of Hilbert Transform and Wavelet Transform

The Hilbert-Huang Transform (HHT) (4) and Wavelet Transform (WT) (12) are conceptually and mathematically quite like each other and yet they have certain diverse applications. They both split signals into a frequency band[20]. But their other attributes vary. The HHT provides information about certain frequencies, whereas the WT provides an overview of spectral changes in power over a time domain. The WT convolves a signal using a pre-defined mother wavelet to decompose a signal. The HHT does not require any pre-defined function for convolution of the signal. One of the major advantages of HHT is that its ability to recognize subtle changes in frequencies whereas, a wavelet transform assumes frequency stationarity during the time span of the wavelet.

In conclusion, depending on the type of signal available, HHT and WT can be used[19]. In case of a a-priori statistical information, it would be advised to use WT. And, if the data is completely random and difficult to categorize, HHT would provide a better solution.

G. Peak detection

The presence of random noise in real experimental signal will cause many false zero-crossing simply due to the noise. To avoid this problem, the technique described here first smooths the first derivative of the signal, before looking for downward-going zero-crossings, and then it takes only those zero crossings whose slope exceeds a certain predetermined minimum (called the "slope threshold") at a point where the original signal exceeds a certain minimum (called the

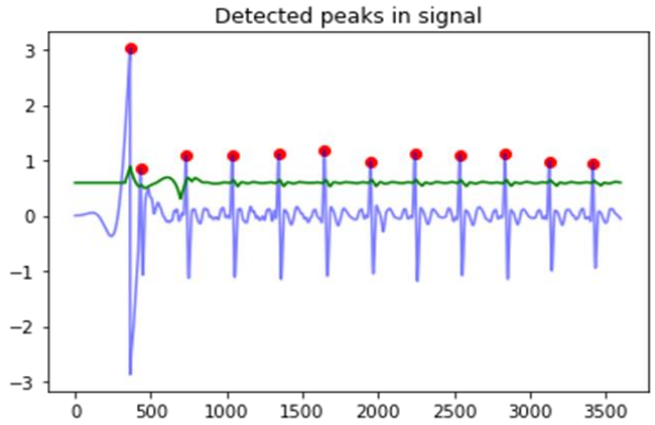


Fig. 18: Peak detection.
Average heartbeat=72.0
Number of peaks in sample=12

"amplitude threshold")[12]. By carefully adjusting the smooth width, slope threshold, and amplitude threshold, it is possible to detect only the desired peaks and ignore peaks that are too small, too wide, or too narrow.

Moreover, this technique can be extended to estimate the position, height, and width of each peak by least-squares curve-fitting of a segment of the original unsmoothed signal in the vicinity of the zero-crossing. Thus, even if heavy smoothing of the first derivative is necessary to provide reliable discrimination against noise peaks, the peak parameters extracted by curve fitting are not distorted by the smoothing, and the effect of random noise in the signal is reduced by curve fitting over multiple data points in the peak.

H. R-peak Detection

The signal y[n], where the QRS complex is highlighted, will be used to help detect the R peaks. The procedure used follows these steps (table 5) [12]:

Step One: Find the peaks in the function y[n]. These peaks must be greater than a minimum amplitude 'm' and must respect a minimum distance 'l' between them.

Step Two: The region around the identified peaks of y[n] is the likely region for the peaks of R. the moment of occurrence of a peak at y[n]. The peak of R in will be contained in the interval ($n\_0-\varepsilon \leq n \leq n\_0+\varepsilon$). The peak of R will be the largest peak in that range.

With reference to table 6, the variables l,m and epsilon (ε) depicts the minimum distance between peaks in y[n], minimum peak height and minimum distance between maximum peaks between x[n] and y[n] respectively.

TABLE 5

PARAMETERS FOR R-PEAK DETECTION

Variables	Type	Description
x	array	One dimensional array containing the original ECG signal.
y	array	One dimensional array containing the signal with enhanced QRS complex
min_peak_height	float	Minimum height the signal [y] should have to be considered a R peak
min_peak_dist	float	Minimum distance between R peaks
interval	float	Maximum distance to look for a R peak (on x) near a peak an y peak

TABLE 6

VALUES USED FOR R-PEAK DETECTION

sign	type	Value
l	Minimum distance between peaks in y[n]	0.2 fs
m	Minimum peak height in y[n]	0.1
ε	Minimum distance between maximum peaks in x[n] and y[n]	0.03 fs

The procedure described above is illustrated below in two figures (fig 19). The first one shows the peaks detected for the signal y[n]. The following figure shows the R peaks detected in x[n]. The parameters used are written (table 6) below as a function of the sampling frequency (fs=1000Hz). This procedure proposed by [25] is much more robust than simply looking for the peaks of the R wave in the original signal. It avoids confusing the R wave with high amplitude P and T waves. High frequency noise will also be amplified by this method along with the QRS complex. So, if the electrocardiogram signal is too corrupted with high frequency noise this method may have problems. Despite this, for a well-done ECG, high frequency noise levels will not be an issue. In the book, the author of the procedure reports a very high success rate in the MIT-BIH database, which many ECG feature extraction articles use as a benchmark.

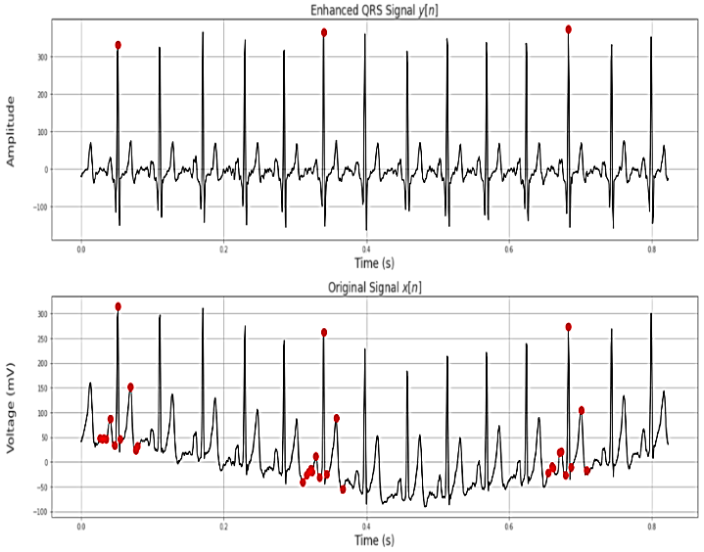


Fig. 19. R-peak detection

Due to locally collected data, unlike the MIT-BIH data, the R-peaks have a slight difference compared to medical ECG signals.

H. Neural Network

The signal collected after analysis in part-I was now passed through a neural network. In this project, both Artificial Neural Network (ANN) and Convolutional Neural Network (CNN) were used to study the model. Although the final part of the project was implemented through the CNN model. The data used here for training was the well-known MIT-BIH dataset[21]. The reason being that compared to locally collected data, the MIT-BIH dataset gave us a thorough and extensive foundation in recognizing the neural network needed to understand our dataset.

The architecture of the model includes a 1D 12-layer convoluted neural network with input layer, convolutional layer, pooling layer, fully-connected layer, and output layer. Since our project did not require any image processing, we used a 1D model. The convolutional layer of the neural network contains the following parameters:

1D ECG ⟶ 1D Convolutional Kernel

$$h_i^{l,k} = f(b_i^{l,k} + \sum_{n=1}^{N} W_{n,i}^{l,k} * x_{i+n-1}^{l-1,k}) \quad (17)$$

where, $h_i^{l,k}$ - output of the i^{th} neuron in layer 'l'
f()- activation function
$b_i^{l,k}$ - offset of the neuron in layer 'l'
$x_{i+n-1}^{l-1,k}$ - output of neuron in layer 'l-1'
$W_{n,i}^{l,k}$ – k^{th} convolutional kernel in 'l^{th}' layer

The pooling layer of the neural network focuses on reducing the dimensions of the convolutional layer output data while keeping in mind network complexity.

$$o_i^{l,k} = f(\alpha_i^{l,k}\, pool(x_i^{l-1,k}) + b_i^{l,k}) \quad (18)$$

where, $o_i^{l,k}$ - output of the i^{th} neuron in 'l' layer
f()- activation function
$b_i^{l,k}$ - offset of neurons in 'l' layer
$\alpha_i^{l,k}$ - sampling weight coefficient
$x_i^{l-1,k}$ – output of neuron in (l-1) layer

VI. Results and Discussions

The project consists of three segments: Signal Processing, Training using Neural Networks, and Implementation from cloud to edge (fig 3). The signal processing phrase involved the collection of raw ECG signal from subjects for a time interval of 4 minutes. The volunteers were allowed to engage in any activity of choice during the signal collection part. This collected signal was then converted from analog signal to data and saved in a .csv format. This was done by using Arduino tools such as serial monitor and serial plotter. The now converted data is used for the signal analysis and pre-processing part of the project. This was followed by understanding and applying the various methods in terms of ECG signal processing such as Welch's method, Hilbert transform, wavelet transform, high pass and low pass filters and finally R-peak detection as discussed in the paper. The processed signal was now passed through a CNN model. The model was trained using MIT-BIH data. This was because unlike locally collected data, the MIT-BIH dataset is a standardized and widely used for study purposes. We tried to understand the requirements that our data needed using the MIT-BIH dataset. The model trained a total 118,341 parameters. The same was passed through the CNN model. Having an accuracy of 98.82% the MIT-BIH data was used for training the CNN model (table 7). The same model was used to understand the accuracy of data collected by us.

	Predicted labels		
True labels	M_{11}	M_{12}	M_{13}
	M_{21}	M_{22}	M_{23}
	M_{31}	M_{32}	M_{33}

Fig. 20. Confusion matrix template

In this paper, mean squared error (MSE), mean absolute error (MAE) and confusion matrices were used for the result analysis. As shown in (fig 20) , we can calculate the precision, recall, and mean accuracy based on an example of 3-by-3 confusion matrix. In terms of a N-by-N confusion matrix, assessment criteria are defined as follows.

$$Recall \mid Sensitivity = \frac{True\ Positive}{True\ Positive + False\ Negative} = \frac{M_{ii}}{M_{ii} + \sum_{n \neq i}^{N} M_{in}} \quad (19)$$

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} = \frac{M_{jj}}{M_{jj} + \sum_{m \neq j}^{N} M_{mj}} \quad (20)$$

(21)

$$Mean\ Accuracy = \frac{\sum_m^N M_{mm}}{\sum_m^N \sum_n^N M_{mn}} \quad (21)$$

where i,j= 1,2,3…N

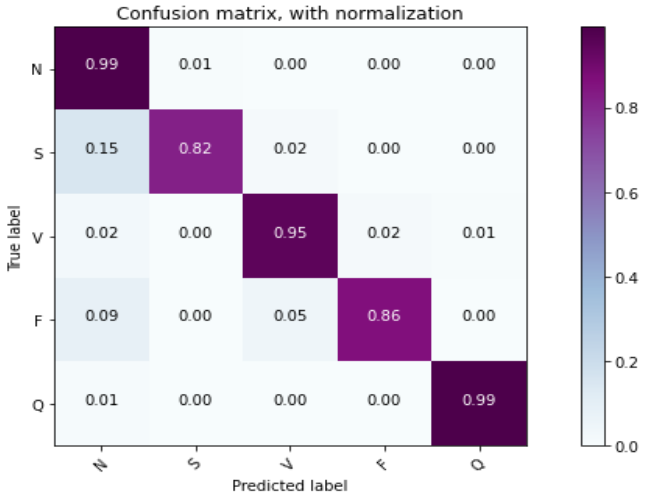


Fig. 21. Wavelet-Neural Network confusion matrix for ECG analysis with normalization

Simulation results are represented with the confusion matrix. Figure 21, we can observe that the wavelet-based neural network model has better performance than referenced models, which was indicated by dark pink blocks located at the diagonal. The confusion matrix helps us evaluate the performance of the classification model. The diagonal entries reflect the percentages of successfully categorized classes, while entries off the diagonal reflect improper categorization. The x-axis and y-axis represent the predicted labels and actual labels, respectively.Although the existing models had relatively good performance for classification of the type Q, N and V, both tended to sort type S to N and identify class A inaccurately to class S and N.

The MSE and MAE of the model was also calculated in order to understand the performance of the system. The mean square error calculates the average of the squares of errors, i.e.; the average squared difference between the predicted values and the actual values. Figure 22 depicts the MSE of the model.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(a_i - a_i')^2 \qquad (22)$$

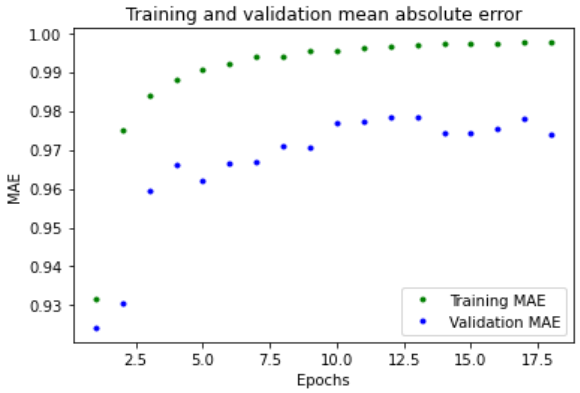


Fig. 22. Mean squared error of the CNN model for ECG analysis

where a_i – observed value

a'_i – predicted value

The mean absolute error or MAE is the average of all absolute errors. It is calculated by taking summation of the absolute difference between the actual and calculated values observed over the entire data and then dividing it with the number of observations.

$$MAE = \sum_{i=1}^{n}\frac{|b_i - a_i|}{n} \qquad (23)$$

where, bi – prediction

ai – true value

n- total data points

Due to laboratory-based collection of raw ECG signals using Wemos D1 mini and AD8232, also keeping in mind that the signal was collected in real time where the volunteers were allowed to engage in either reading, debating, or watching a movie, the model had an accuracy of 86.17% (table 7). The F1 score of the MIT-BIH dataset using CNN is 0.91 whereas for the locally collected dataset has a F1 score of 0.86, using conventional IoT devices for data procurement. This helped us understand the various variables in the ECG dataset. Similarly, the accuracy for both the MIT dataset and IoT data is 0.98 and 0.86. This calculation has been done keeping in mind that the dataset noise and frequency calibrations. The main purpose of this model was to depict and predict the heart beat classification using IoT devices and making a portable heart rate monitoring device. The model unlike the existing CNN models trained using the MIT-BIH data, can collect raw ECG signals, and predict the classification. The model was built using open source jupyter notebook and Arduino IDE. The model was then loaded on Arduino Nano 33 BLE Sense, thus enabling a portable heart beat classification embedded system working in real time. The ECG waves can be viewed using the serial plotter in the IDE or the serial monitor for data reading.

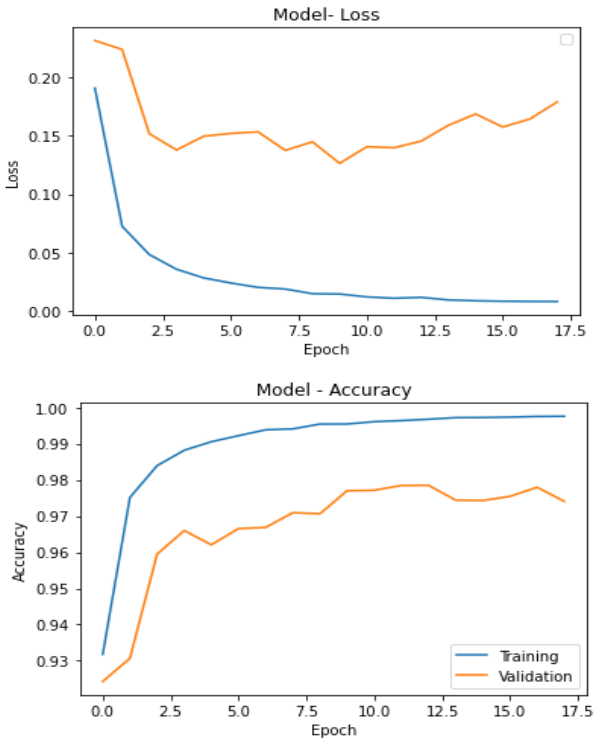


Fig. 23. Model accuracy and model loss

TABLE 7

RESULTS AND COMPARISON BETWEEN THE DATA USED

Data used	F1 score	Accuracy	Loss
MIT-BIH	0.91	0.988	0.0371
IoT Data	0.86	0.8617	0.0421

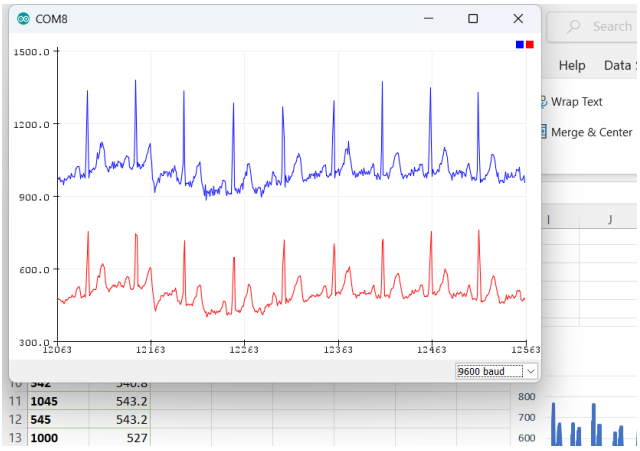


Fig. 24. Final real-time detection and ECG analysis using the embedded wavelet-neural network

The now fully trained wavelet-neural network was loaded onto the Arduino Nano 33 BLE Sense (fig 6(b)) in order to

facilitate a real-time ECG analysis embedded system. The real-time embedded wavelet-neural network for ECG analysis can be visualized using the serial monitor of the Arduino IDE. The implementation and conversion of the keras model to a .h file for Arduino compatibility and working has been discussed at length in section III of the paper. The final ECG depiction (fig 24).

VII. Conclusion And Further Work

This paper used a three-lead ECG electrode patch to record heat beats and evaluated model using existing MIT-BIH dataset. In addition to this, we proposed a Embedded Wavelet-Neural Network for ECG analysis keeping in mind the performance improvement and accessibility using locally (IoT) collected dataset. Comparing with the MIT-BIH data models, the proposed architecture had a better performance and is easily accessible. Considering practicality, overall performance and computation time, the proposed embedded wavelet-neural network model works in real-time and has data transferred from cloud to the embedded system. Compared with referenced MIT-BIH data, the model works in real time using discrete wavelet transform and can be easily industrialized using Internet of Things devices. Although the model has a good performance, there were still some drawbacks. The model is based on dataset that has been collected separately and then pre-processed. After the initial filtering, the data was passed through a wavelet-neural network, thus making it a tedious process. Moreover, the collected data is uploaded to the cloud and analyzed offline by using discrete wavelet transform for noise reduction and then implemented in an embedded system. However, it is not the end of the research. The importance of a real-time interaction between engineering and biomedical signals without uploading data into the cloud and enhancing the availability of the embedded system is also our further work, which relates to edge computing technology using embedded systems.

References

[1] *Mayo Clinic*," heart arrhythmia"
Available:mayoclinic.org/diseases-conditions/heart-arrhythmia/symptoms-causes/syc-20350668 [Accessed September 12,2007]

[2] Xu Yuefan ,Zhang Sen, Cao Zhengtao, Chen Qinqin, Xiao Wendong, "Extreme learning machine for heartbeat classification with hybrid time-domain and wavelet time-frequency features," *journal of healthcare engineering*, Vol. 2021, Article ID:6674695. [Online serial].
Available:Hindawi,hindawi.com/journals/jhe/2021/6674695

[3] Sophocles J. Orfanidis, "Introduction to Signal Processing", Prentice-Hall, 1996

[4] Mortara, David W, "Digital filters for ECG signals." *Computers in Cardiology* (1977): 511-514

[5] Macfarlane, Peter W., and Thomas Davidson Veitch Lawrie, "Comprehensive electrocardiology: theory and practice in health and disease" Vol. 4, Cap. 37, *Pergamon*, 1989.

[6] Adam Gacek, Witold Pedrycz , "ECG signal processing, classification, and interpretation- A comprehensive framework of computational intelligence" Heidelberg , DE : *Springer*, 2014.

[7] Allen B. Downey, "Think DSP: Digital Signal Processing in Python" *Sebastopol*, US: O'Reilly Media, 2016.

[8] Zhishuai Liu, Guihua Yao, Qing Zhang, Junpu Zhang, Xueying Zeng, "Wavelet Scattering Transform for ECG Beat Classification", *Computational and Mathematical Methods in Medicine*, vol. 2020, Article ID 3215681, 11 pages, 2020. [online serial]. Available: Hindawi, https://doi.org/10.1155/2020/3215681

[9] Schreier, P.; Scharf, L. (2010). *Statistical signal processing of complex-valued data: The theory of improper and noncircular signals.* Cambridge, UK: Cambridge University Press.

[10] Martin K. Stiles , Clifton D, Grubb NR, Watson JN, Addison PS, "Wavelet-based analysis of heart-rate-dependent ECG features." *Annuals Noninvasive Electrocardiol*,vol.9,no.4, October,2004. Available: https://ncbi.nlm.nih.gov/pmc/articles/PMC6932565

[11] Titchmarsh, E. (1986) [1948]. *Introduction to the theory of Fourier integrals* (2nd ed.). Oxford, UK: Clarendon Press. ISBN 978-0-8284-0324-5.

[12] Yuefan Xu, Sen Zhang, Zhengtao Cao , Qinqin Chen, Wendong Xiao, "Extreme Learning Machine for Heartbeat Classification with Hybrid Time-Domain and Wavelet Time-Frequency Features," *J Healthc Eng*., Jan.11, 2021. Available:- PubMed (nih.gov)

[13] Antonio Horta Ribeiro (2020), "Automatic ECG diagnosis using a deep neural network"
Available:https://github.com/antonior92/ECG-jupyter-notebook/blob/master/ECG.ipynb

[14] Hari Mohan Rai, Anurag Trivedi, Shailja Shukla, "Ecg signal processing for abnormalities detection using multi-resolution wavelet transform and artificial neural network classifier," *measurement*, vol.46, issue 9, Pages 3238-3246, 2013 Available: ScienceDirect, https://www.sciencedirect.com/science/article/pii/S0263224113002133

[15] TensorFlow, "Model conversion," *TensorFlow*, 2022.
Available: tensorflow.org/lite/models/convert/

[16] TensorFlow," Build and convert models," *TensorFlow*, 2022
Available: tensorflow.org/lite/microcontroller/build_convert

[17] Advait Jain *et al.*, Apache version 2.0 (Oct 13, 2022) *TensorFlow tflite-micro*. Available:github.com/tensorflow/tflite-micro/blob/main/tensorflow/lite/micro/examples/hello_world

[18] TensorFlow," Get started with microcontroller" *TensorFlow*, 2022.
Available:
https://www.tensorflow.org/lite/microcontrollers/get_started_low_level

[19] Arduino documentation, "Get started with machine learning on Arduino" *Arduino documentation*,2022.
Available: https://docs.arduino.cc/tutorials/nano-33-ble-sense/get-started-with-machine-learning [online]

[20] M. Feldman, "Hilbert transforms," in *Encyclopaedia of Vibration*, Pages 642-648, Elsevier, [online document],2001. Available: ScienceDirect, https://doi.org/10.1006/rwvb.2001.0057.

[21] M.k.Saini "Signals & systems- what is Hilbert transform," *tutorialspoint.com*, Dec. 15,2021.[Online].
Available: www.tutorialspoint.com/signals-and-systems-what-is-hilbert-transform

[22] Ramazan Cetin, Selen Gecgel, Gunes Karabulut Kurt, Senior Member, IEEE, and Faik Baskaya, Member, IEEE, "Convolutional Neural Network-Based Signal Classification in Real Time",
IEEE Embedded Systems Letters, Vol. 13, No. 4, December 2021

[23] Osgood, Brad, "*The Fourier Transform and its Applications"* (PDF), Stanford University, retrieved 2021-04-30

[24] Yuksel. Ozbay, Rahime. Ceylan, Bekir. Karlik, "Integration of type-2 fuzzy clustering and wavelet transform in a neural network-based ECG classifier", *Expert Syst. Appl*., 38 (2011), pp. 1004-1010

[25] B. Devine & P. W. Macfarlane
"Detection of electrocardiographic 'left ventricular strain' using neural nets", *Medical and Biological Engineering and Computing volume* 31, pages343–348 (1993)

[26] D. Benitez *et al*.,
"The use of the Hilbert transforms in ECG signal analysis", *Computers in Biology and Medicine* (2001): 399-406"

[27] Zumray. Dokur, Tamer. Olmez
"ECG beat classification by a novel hybrid neural network," *Comput. Methods Prog. Biomedicine*, 66 (2001), pp. 167-181

[28] Hari Mohan Rai, Anurag Trivedi
"De-noising of ECG waveforms using multiresolution wavelet transform", *International Journal of Computer Application*, 45 (18) (2012)

[29] TensorFlow tflite-micro, GitHub.
Available:github.com/tensorflow/tflite-micro/blob/main/tensorflow/lite/micro/examples/hello_world.