SMART DIGITAL STETHOSCOPE FOR PULMONARY DISEASE DIAGNOSIS

A thesis submitted in partial fulfillment of the requirements for
Award of the degree of

B. Tech

In

INSTRUMENTATION AND CONTROL ENGINEERING

By

ADITYA DHAVALA (110117007)

ARPIT MOHAKUL (110117015)

ASIF AHMED (110117017)

B. YASASWINI (110117023)

P. SRIKAR (110117061)

PRADNYA MESHRAM (110117063)

RISHABH VERMA (110117069)

DEPARTMENT OF

INSTRUMENTATION AND CONTROL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY

TIRUCHIRAPPALLI – 620015

MAY - 2021

BONAFIDE CERTIFICATE

This is to certify that the project titled “SMART DIGITAL STETHOSCOPE FOR PULMONARY DISEASE DIAGNOSIS” is a bonafide record of the work done by

Aditya Davala (110117007)
Arpit Mohakul (110117015)
Asif Ahmed (110117017)
B. Yasaswini (110117023)
P. Srikar (110117061)
Pradnya Meshram (110117063)
Rishabh Verma (110117069)

in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Instrumentation and Control Engineering of the NATIONAL INSTITUTE OF TECHNOLOGY, TIRUCHIRAPPALLI, during the year 2020-2021.

[illegible]

Dr.R.Periyasamy	**Dr.G.Uma**
Project Guide	Head of Department

Project viva-voce held on 25/05/2021.

Internal Examiner	**External Examiner**

ABSTRACT

Observing pulmonary sounds is highly essential in examining the respiratory sounds for identifying abnormalities. While it is done simply by auscultation and palpation inspection, automated analysis of lung auscultation sounds can benefit the health systems in low-resource settings where there is a lack of skilled physicians. The primary objective is to separate the heart noise from the overall corrupted signal, to import the audio file as spectrographs for the neural network to understand and classify. The secondary objective lies in improving the network and make it more robust with minimal training data. To do this, we have to find the best combinations of hidden layers, convolution layers, and the number of both input and output neurons. The proposed scheme's performance is studied using a patient independent train-validation set from the publicly available ICBHI 2017 lung sound dataset. Later we compare the accuracies and confusion matrices of different models and report our findings.

Keywords: *Lung auscultation sound, Respiratory disease detection, Convolutional neural networks, Mel frequency cepstral coefficient, Empirical mode decomposition*

ACKNOWLEDGEMENTS

We would like to express our deepest gratitude to the following people for guiding us through this course and without whom this project and the results achieved from it would not have reached completion.

We thank **Dr. R. Periyasamy**, our project guide for helping us and guiding us through the course of this project.

We extend our gratitude to **Dr. G Uma**, the Head of the Department, Department of Instrumentation and Control Engineering, for allowing us to avail of the facilities at the department.
Our hearty thanks to the Department Project Evaluation Committee (DPEC), comprising Dr. Dhanlaksmi, Dr. V. Sridevi, and Dr. Shiraz Sohail, for their valuable suggestions during the reviews.

We are also thankful to staff members, B. Sangeetha(Ph.D. scholar), parents, and the Department of Instrumentation and Control engineering faculty, staff members, parents, and friends for their constant support and help.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Lung sounds

The respiratory system's internal sounds are synonymous with lung sounds. For a long time, our only means to listen to these sounds have been the Stethoscope. Stethoscopes perform auscultation (the medical term for listening in to internal bodily sounds). The functioning of the lung is an essential indicator of a variety of diseases, and Auscultation is still an integral part of a physical medical examination.

1.2 Need to classify lung sounds

However, these lung sounds are not accurately picked up by a regular stethoscope due to noises and device errors. The Doctor's human errors also might account for misidentification of the type of lung sound observed. These reasons are why there exists a need to advance the regular stethoscope. Our project includes detecting, processing, and classifying lung sounds into the related diseases.

1.3 Types of Lung sounds

Lung sounds are broadly classified into the following types

1. Breath Sounds
2. Voice Sounds
3. Adventitious Sounds

Breath sounds are associated with the upper respiratory tract; these are observed at the very beginning of inhalation and exhalation and are further classified as follows:

1. Normal breath (vesicular sounds)
2. Bronchial breathing
3. Soft breathing
4. Diminished breathing

Voice sounds are generated from the larynx and are generally used to check the resonance through the chest wall, where the apparatus is placed.

Adventitious sounds originate in the lower respiratory tract and are the significant indicators for any underlying pathological conditions. These are classified as follows:

1. Crackles
   - Coarse/fine discontinuous and sporadic explosive sounds
2. Wheezing
   - Continuous high/low pitched sounds may be of singular/multiple tones
3. Stridor
   - High pitched loud sounds heard during inspiration
4. Mediastinal crunch
5. Pleural/Pericardial rub

1.4 Noises in lung sounds

The primary noise in detecting Lung sounds arises from the Circulatory System, i.e., the Heart sounds. It is imperative to use Signal Processing to separate the heart sounds from the lung sounds. Other noises arise from the surroundings where the recordings are taken in a busy or clustered environment, noises from fans, ambiance, etc. Most of these can be reduced by examining in a closed room.

1.5 Diseases that can be identified with lung sounds

The following diseases are being considered in this project :

- Bronchiectasis

  This is a condition where the bronchial tubes in the lungs get damaged, widened, and thickened. These damaged air passages allow bacteria and mucus to build up in the lungs. This results in frequent infections and blockages of the airways.
  Symptoms are a regular cough and discomfort in the chest
  This is a common condition and could be a symptom of various diseases.
- Bronchiolitis

  This is a condition where inflammation is present in the bronchioles, the smaller airways at the end of the respiratory tract. It is commonly seen in the younger population and children. Bronchiolitis is almost invariably caused by viruses like the common cold and can be treated with Analgesics and supportive care.
- Pneumonia

  Pneumonia is an infection that inflames the air sacs in one or both lungs. The air sacs may fill with fluid or pus (purulent material), causing cough with phlegm or pus, fever, chills, and difficulty breathing. A variety of organisms, including bacteria, viruses, and fungi, can cause pneumonia.

Symptoms include a cough with phlegm or pus, fever, chills, and difficulty breathing.
Antibiotics can treat many forms of pneumonia. Vaccines can prevent some forms of pneumonia.

- URTI

  An upper respiratory tract infection (URTI) is an illness caused by an acute infection, which involves the upper respiratory tract, including the nose, sinuses, pharynx, or larynx. This commonly includes nasal obstruction, sore throat, tonsillitis, pharyngitis, laryngitis, sinusitis, otitis media, and the common cold. URTIs usually resolve within seven to 10 days.
  Symptoms usually resolve within two weeks and include a scratchy or sore throat, sneezing, stuffy nose, and cough.
  Treatment includes rest and medication to relieve symptoms.
- COPD

  Chronic obstructive pulmonary disease (COPD) is a chronic inflammatory lung disease that causes obstructed airflow from the lungs. Symptoms include breathing difficulty, cough, mucus (sputum) production, and wheezing.
  Damage to the lungs from COPD can't be reversed.
  Symptoms include shortness of breath, wheezing, or a chronic cough.
  Rescue inhalers and inhaled or oral steroids can help control symptoms and minimize further damage.

CHAPTER 2
LITERATURE REVIEW

The list below is the papers that were referred to in the course of this project.

1. **Color enhancement aims the of lung sounds based on empirical mode decomposition and Fourier transform algorithm -** *Ashok Mondal,2017*

   During the re-recording of lung sound (LS) signals, there is always a heart sound (HS) signal interfering. This obscures the characteristics of LS signals and leads to misinterpretation about pulmonary disorders, assuming any exist. Based on the empirical mode decomposition (EMD) technology and prediction algorithm, Ashok Mondal et al. proposed decomposing the noisy lung sound into multiple components, i.e., intrinsic mode functions (IMFs), and removing heart sounds from each IMF instead of removing them from the entire lung sound signal. Following that, HS-included portions are located and discarded using heron's triangular criteria and removing peaks in adjacent regions. A new Fast Fourier transform (FFT) based prediction method predicts the missing values of the gap thus generated, and the time domain LS signal is reconstructed by obtaining an inverse FFT of the estimated missing values. For a 0 dB SNR value, their technique yields a signal to deviation ratio (SDR) of 9.8262 on their dataset.

2. **Lung sounds classification using convolutional neural networks-** *Dala Bardou, 2018*

   This paper proposed a neural network model with MFCC based feature extraction techniques. The lung sound data used in the paper is from the Respiratory acoustics laboratory environment. The RALE represents an educational program involving students, adults and allied health care workers. The various data augmentation techniques used are spectrogram cropping and Vocal tract length perturbation. The paper proposed and compared the performance of three different models. The models used in the study were Support vector machine, K nearest neighbors, and Gaudian mixture models in combination with a convolutional neural network model. The accuracies achieved for an individual CNN model were around 93% after 20000 iterations. Later the results from various models were compared for further improvements and discussions.

3. **A Lightweight CNN Model for Detecting Respiratory Diseases from Lung Auscultation Sounds using EMD-CWT-based Hybrid Scalogram-** *Samiul Based Shuvo, 2020*

    The paper describes a methodology of classifying lung sounds into various diseases. A lightweight CNN model is proposed in the paper with a feature extraction method based on continuous wavelet transform and a scalogram approach. The journal trains a neural network model over ICBHI 2017 dataset. There is an imbalance in the ICBHI 2017 data set. To prevent model imbalance, various data augmentation techniques have been employed. The scalograms have been augmented using four different color mapping schemes. The four color mapping schemes used in Matlab 2020a are Parula, HSV, Jet, and Hot. Later the lightweight convolutional neural network was trained, and the results were compared to previous works. The model achieves an accuracy of 98%. A comparative study made between various other works also proves that the current model proposed in the paper is computationally more efficient and gets results faster.

CHAPTER 3
LUNG SOUND DATABASE

3.1 Data extracted

Pre-recorded pulmonary and cardiac audio samples from the RALE database and ICBHI 2017 are extracted. This data was present in the form of "*.wav* " files.

One of the sounds from the RALE Dataset has been used here for demonstration.

To load signal: function *wavfile.read()* is used from *scipy.signal()* Library.

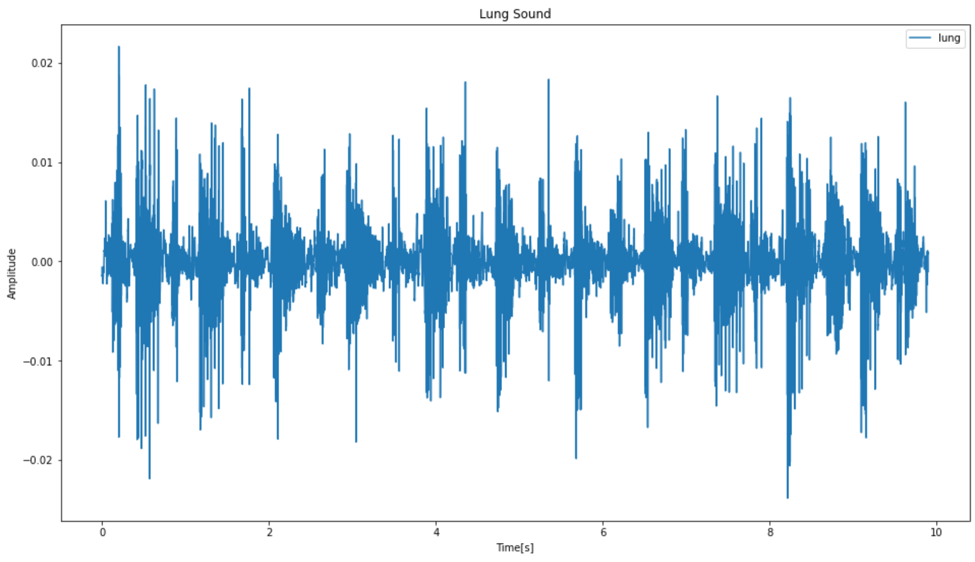

Figure 1: Lung Sound

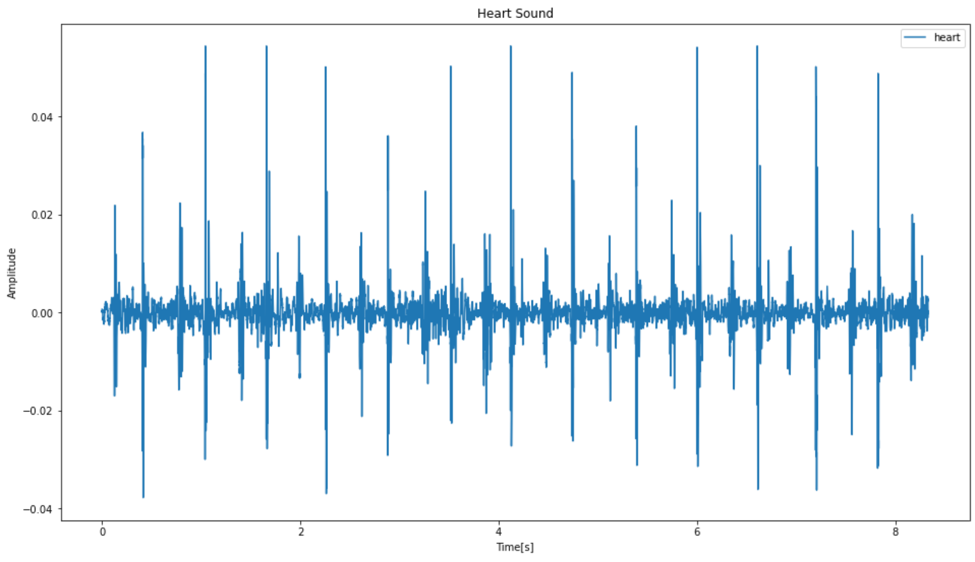

Figure 2: Heart Sound

The signals are about 10 seconds in duration. While lung sounds are 8KHz, the heart sounds are found to have a sampling rate of 4KHz.

3.2 Sampling Cycles

From the Heart and Lung Sounds extracted from the Database, we are required to sample a segment of each dataset, as only a few respiratory cycles should be enough for us to work with. As the respiratory cycles repeat, it is safe to assume that there are no data losses while sampling occurs.

It was imperative that both Heart and Lung Sounds Sampling rates should be matched. Hence, the signals were resampled to the lowest sampling rate, which is 4000Hz in this case. From the original Heart and Lung Sounds Dataset, 12000 Samples were obtained after this.

Method *resample()* from the *Scipy.signal()* Library was used to carry out this Sampling

The obtained Sampled signals are presented below :

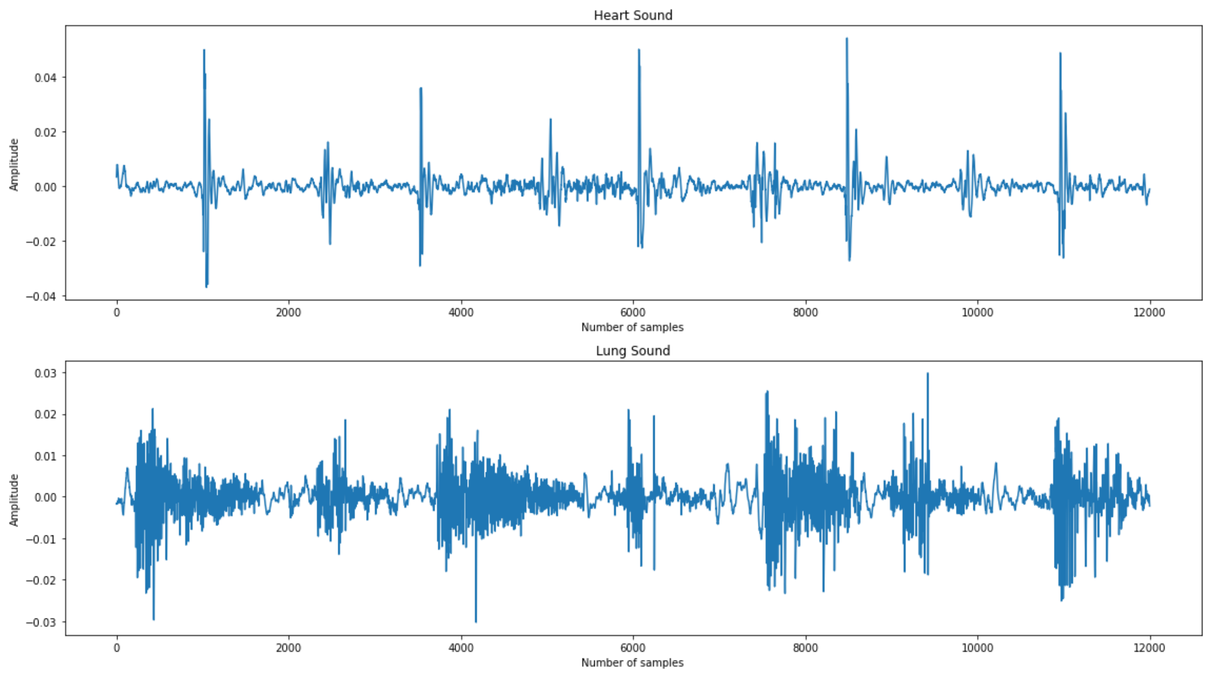

Figure 3: Obtained Sampled Signals (Heart and Lung Sounds)

3.3 Generating Mixed Heart + Lung Sound

We have sourced Heart & Lung sounds from the database, but our problem statement dictates that the input signal is an overlap of both Heart and Lung Sounds, as our Stethoscope would pick up both the sounds, and it is up to us to distinguish the Lung sounds from the mix.

Hence, we need to create our input signal (which would include both Heart and Lung Sounds together, as the Digital Stethoscope shall measure sounds from the chest wall) by mixing the independently sourced Heart and Lung Sounds from the Dataset.

This overlap gives us the following signal, which we shall use for the remainder of this project. Our Signal processing goal is to separate the Heart sounds and reobtain just the lung sounds for further classification.

The mixed input sound is represented below :

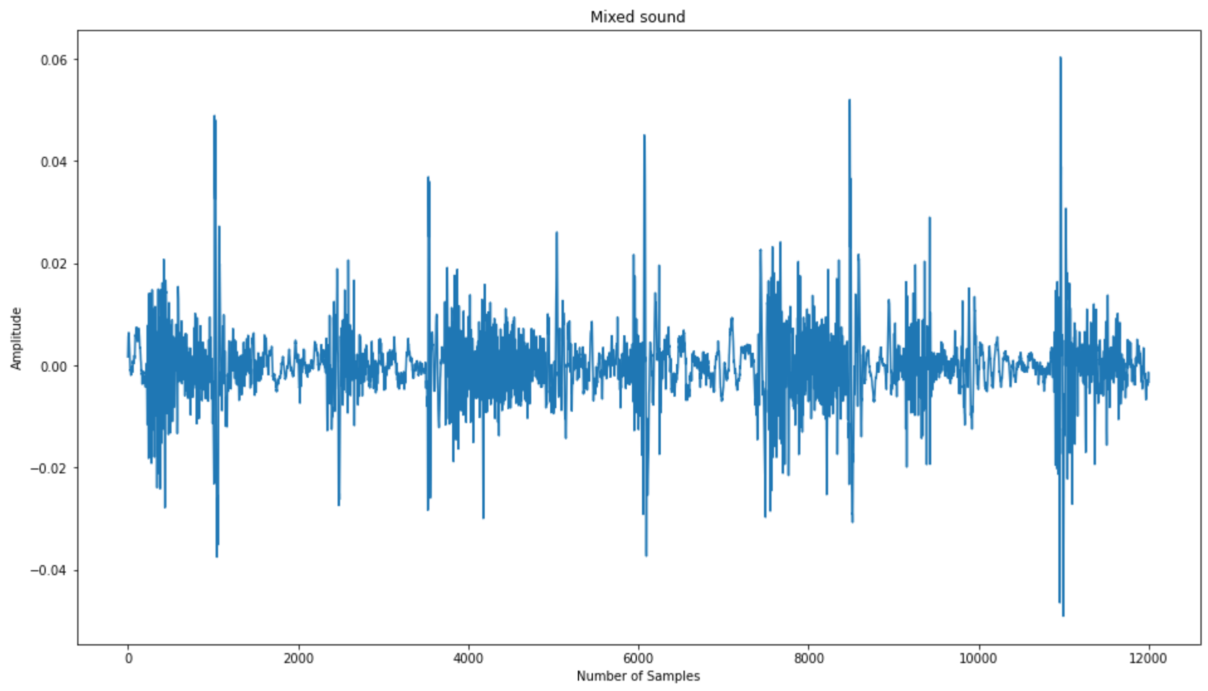

Figure 4: Mixed Heart and Lung Sound Input

CHAPTER 4
SIGNAL PROCESSING

The aim of the Signal Processing portion of this project is to separate the Heart sounds from the mix of Heart and Lung sounds. We are to obtain sole Lung sounds for classification.

The subsequent subsections deal with the same. We identify the heart sounds and separate them from the mix of Heart and Lung Sounds. This is done in the following broad steps:

- Decomposing the mixed noisy signal into Intrinsic Mode Functions (IMFs).
- Localizing the Heart Sound Peaks, as the heart sounds are more extensive in amplitude.
- Estimating the boundary of the region dominated by heart sounds.
- Removing the Heart peaks appropriately with minimal loss of Lung sound data.
- Predicting the missing samples that were removed, according to the existing lung sound data.
- Reconstruction of the Lung signal.

A crucial step to perform before this is to decompose the Input Signal into smaller "signals" that are easier to work and operate on. This is done by the Empirical Mode Decomposition (EMD) algorithm explained in the following section.

4.1 Empirical Mode Decomposition (EMD)

EMD is a method of breaking down a signal without leaving the time domain. It can be compared to other analysis methods like Fourier Transforms and wavelet decomposition. The process helps analyze natural signals, which are most often non-linear and non-stationary.[1]

EMD filters out functions that form a complete and nearly orthogonal basis for the original signal. Completeness is based on the method of the EMD; the way it is decomposed implies completeness. The functions, known as *Intrinsic Mode Functions (IMFs)*, are therefore sufficient to describe the signal, even though they are not necessarily orthogonal.[2]

The essence of the method is to empirically identify these intrinsic oscillatory modes by their characteristic time scales in the data and then decompose the data accordingly. Through a process called *sifting*, most of the riding waves, i.e., oscillations with no zero-crossing between extrema, can be eliminated. THUS, the EMD algorithm considers signal oscillations at a very local level and separates the data into locally non-overlapping time scale components.

It breaks down a signal x(t)into its component IMFs obeying two properties:[3]

1. An IMF has only one extremum between two subsequent zero crossings, i.e., the number of local minima and maxima differs at most by one.
2. An IMF has a mean value of zero.

The EMD Algorithm (Sifting Process) can be broken down into the following steps :

- The sifting process is what EMD uses to decompose the signal into IMFs.

  The sifting process is as follows:
  For a signal X(t), let m_1 be the mean of its upper and lower envelopes as determined from a cubic-spline interpolation of local maxima and minima. An arbitrary parameter determines the locality.; the calculation time and the effectiveness of the EMD depends significantly on such a parameter.

- The first component h_1 is computed:
  $$h_1=X(t)-m_1$$
- In the second sifting process, h1 is treated as the data, and m_{11} is the mean of h_1s upper and lower envelopes:
  $$h_{11}=h_1-m_{11}$$
- This sifting procedure is repeated k times, until h_{1k} is an IMF, that is:
  $$h_{1(k-1)}-m_{1k}=h_{1k}$$
- Then it is designated as $c_1=h_{1k}$, the first IMF component from the data, which contains the shortest period component of the signal. We separate it from the rest of the data: $X(t)-c_1 = r_1$
  The procedure is repeated on r_j: $r_1-c_2 = r_2, \ldots, r_{n-1} - c_n = r_n$.

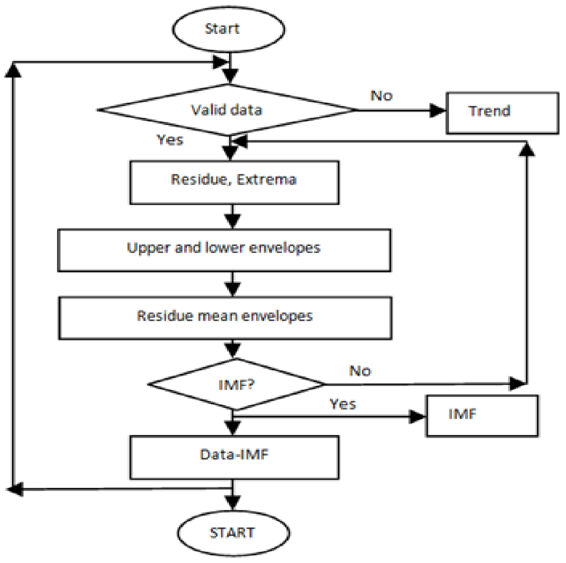

Figure 5: Empirical Mode Decomposition [courtesy (Wang et al. [26])

4.2 Ensemble Empirical Mode Decomposition(EEMD)

EEMD (Ensemble EMD) is a noise assisted data analysis method. EEMD consists of "sifting" an ensemble of white noise-added signals. EEMD can separate scales naturally without any a priori subjective criterion selection as in the intermittence test for the original EMD algorithm.
EEMD has been recently proposed to eliminate the mode mixing problem of the EMD technique. Essentially, the EEMD repeatedly decomposes the original signal with added white noise into a series of IMFs, by applying the actual EMD process. This means the corresponding IMFs during the repetitive process are considered as the final EEMD decomposition result. Since white noise is added throughout the entire signal decomposition process, mode mixing is effectively eliminated. The EEMD process has been used already to detect rotating machine faults such as defective bearings and gears in the past few years.

Application in the project

The EEMD algorithm is applied to the mixed signal to decompose it. This was done using the *PyEMD* package available to install from the command window using pip install EEMD-signal.

The EEMD Function creates an EEMD Object and, when our input signal is passed into it, makes the following IMFs :

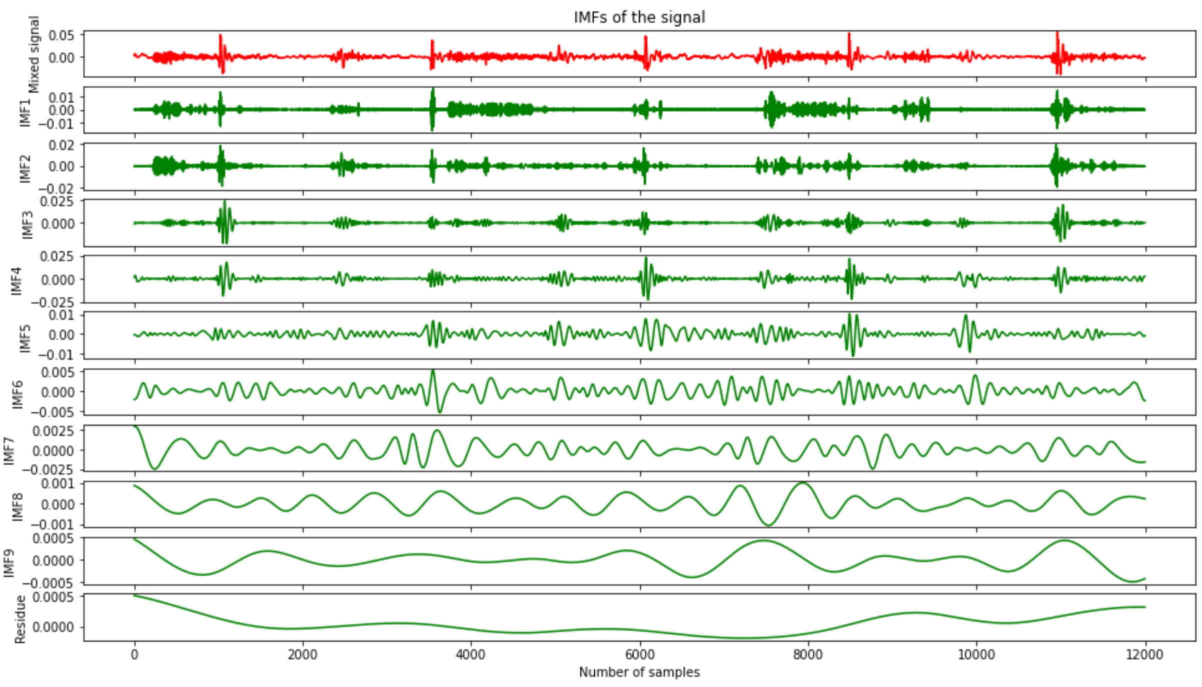

Figure 6: Output Intrinsic Mode Functions (IMFs) created by EEMD Function

The decomposition of the Input Signal into these IMFs simplifies the signal processing steps. Instead of working on the entire signal, working on the IMFs separately and then reconstructing them, in the end, shall lead us to accurate end results.

In the following steps, we individually performed operations (to separate heart sounds) on each of the IMFs. In the end, we were left with a new set of IMFs, which were then reconstructed into our required signal (Lung sound).

4.3 Heart Sound Peak Detection

To detect the heart peaks appropriately, we perform the following operations :

- Passing through a Butterworth Filter
- Creating and Smoothening a Hilbert Envelope
- Identification of Peaks, considering the thresholds

These steps were arrived at from the literature surveys that we conducted. These steps are generally taken as a standard in peak identification.

4.3.1 Butterworth Filter

The Butterworth low-pass filter is chosen because it gives out a flat frequency response in the passband. Higher-order Butterworth filters can give near ideal responses. Here, we used a 10th order Butterworth low pass filter. The Cutoff frequency was set at 150Hz, as this is the general average frequency of heart sounds.

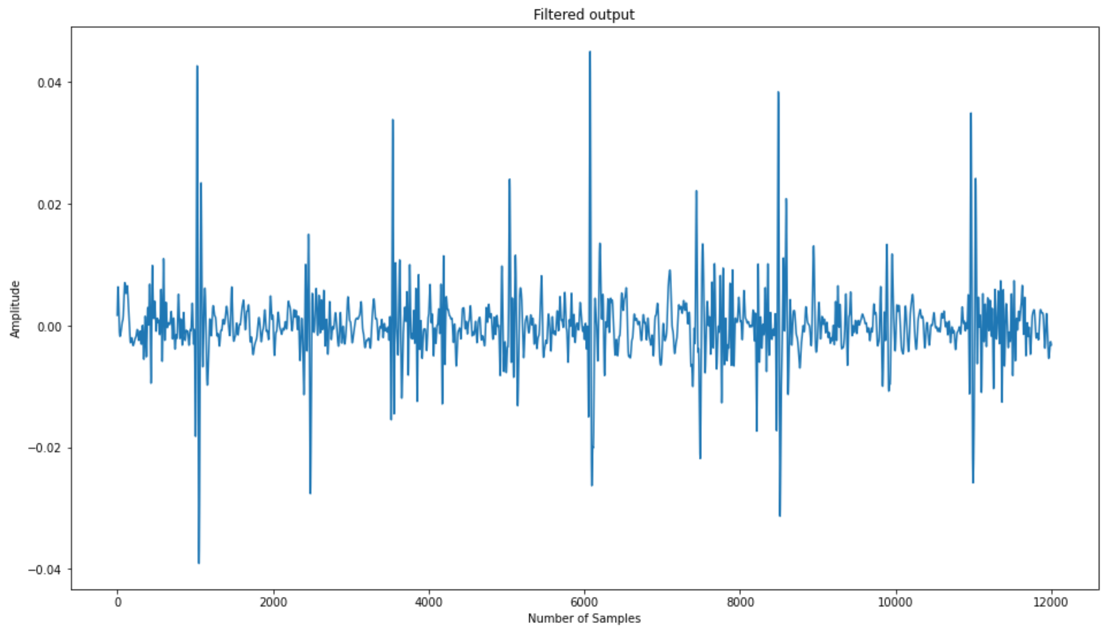

Figure 7: Butterworth Filter Output

4.3.2 Hilbert Envelope

The Hilbert Envelope takes the input signal and forms a close approximation of the high amplitude values of a given signal. Using a Hilbert Envelope to detect peaks simplifies the concern to consider the low amplitude signals. The Hilbert envelope also causes the least loss of the original signal compared to other signal envelopes, as demonstrated in the following image.

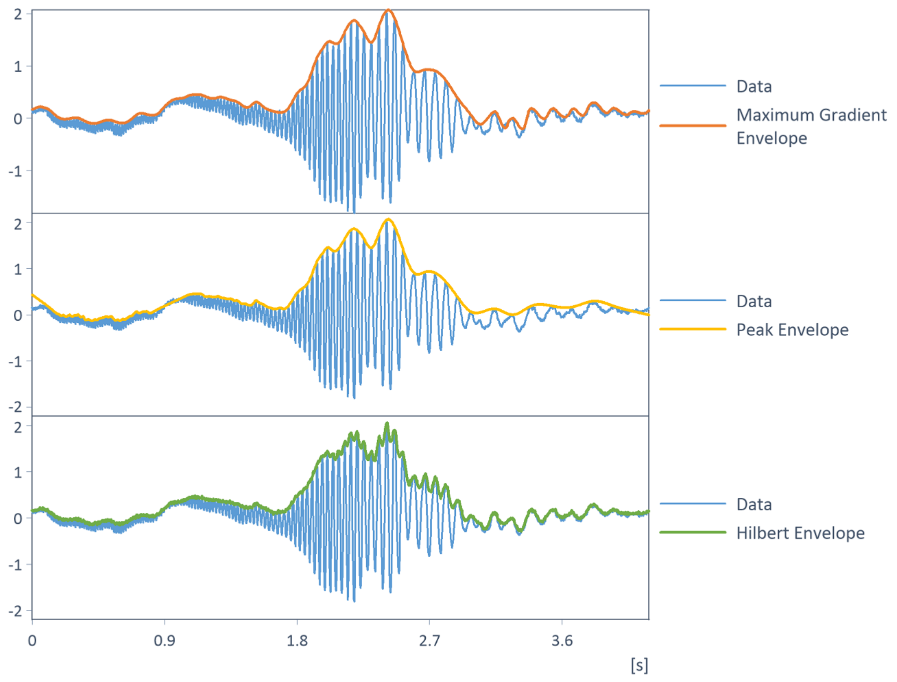

Figure 8: Example image showing least loss of signal in Hilbert Envelope taken from [Weisang documentation]

The Butterworth Filtered Output was subjected to a Hilbert Transform.ToThe Hilbert Envelope was developed and is presented below :

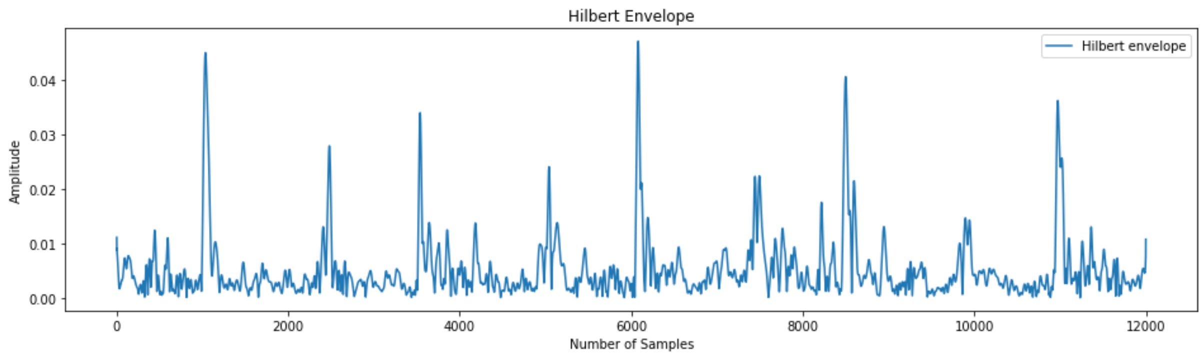

Figure 9: Developed Hilbert Envelope

4.3.3 Smoothened Hilbert Envelope

The Hilbert envelope is further smoothened using a Butterworth Low Pass Filter of order 5. The cutoff frequency varies from 7-25Hz.

This is done so as to further narrow down the peaks which are to be detected.

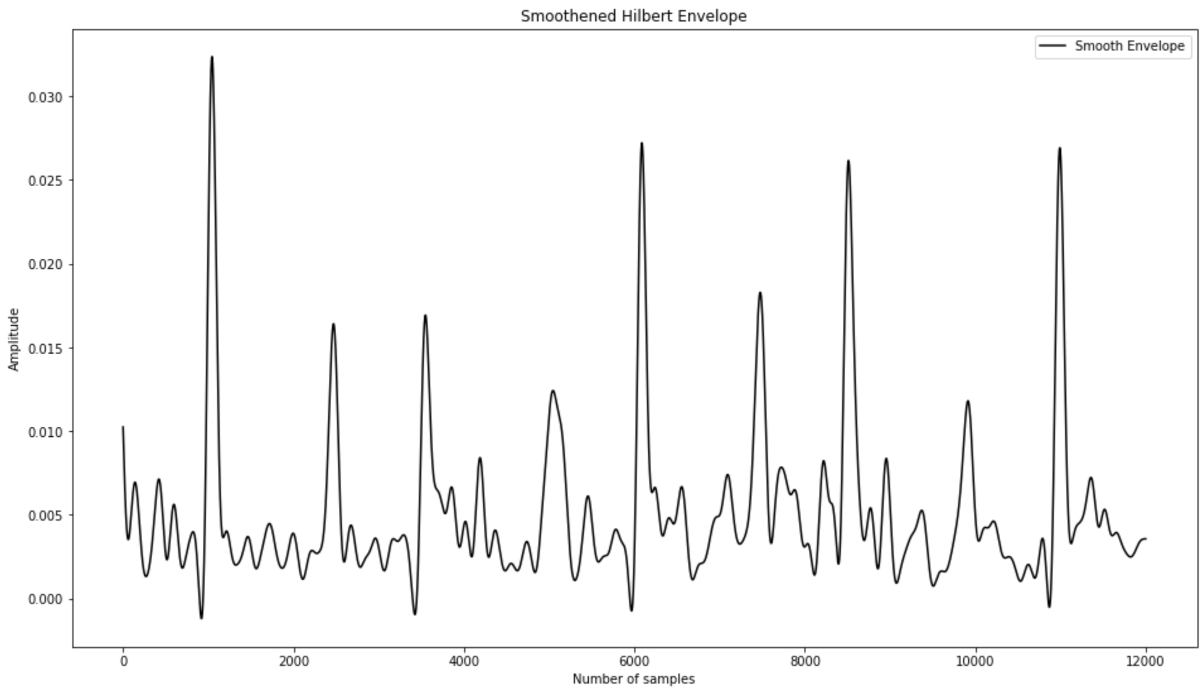

Figure 10: Smoothened Hilbert Envelope

4.3.4 Peak identification

This is done using the *peakutils* libraries from Python.The peaks identified in the above stages might belong to noise components or lung sounds at the given frequency. In order to identify only the heart peaks, only those that are above a threshold value are considered.
These peaks alone are retained as Heart Peaks and are removed subsequently.

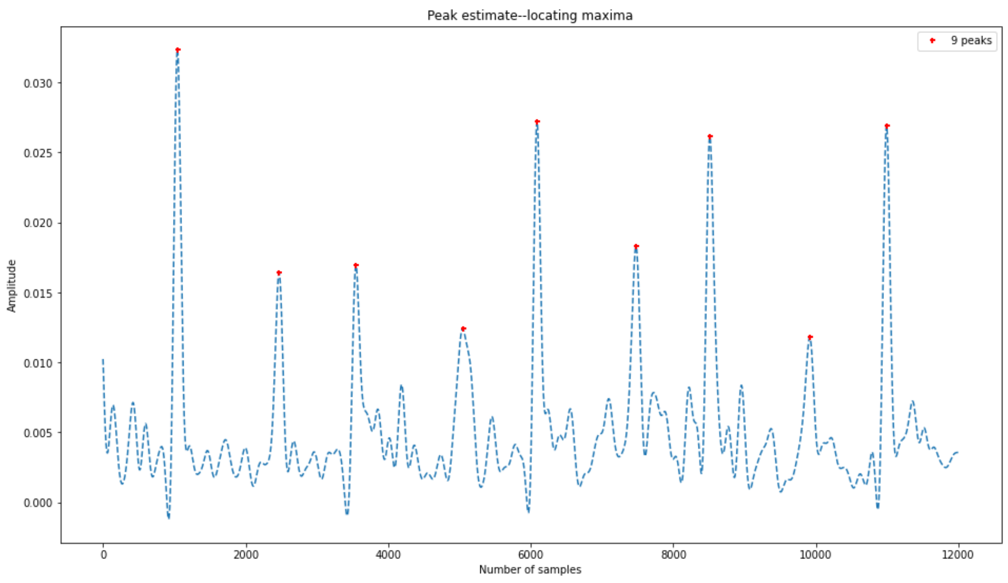

Figure 11: Peak Estimate - locating maxima

4.4 Boundary Estimation and Removal

After the Heart Peaks were identified in the previous step, we need to decide what portions of the sample around the peaks are to be removed. The boundaries of the heart signal need to be estimated. These boundaries indicate a range of samples around the Heart Peaks that shall be removed.

We used the concept that heart sounds dominate about 80ms on the right side and 40ms on the left side of the peak[4]

We arrived at the heart boundaries around the heart peak and decided to remove 160 samples on the left and 320 samples on the right of the heart peak.

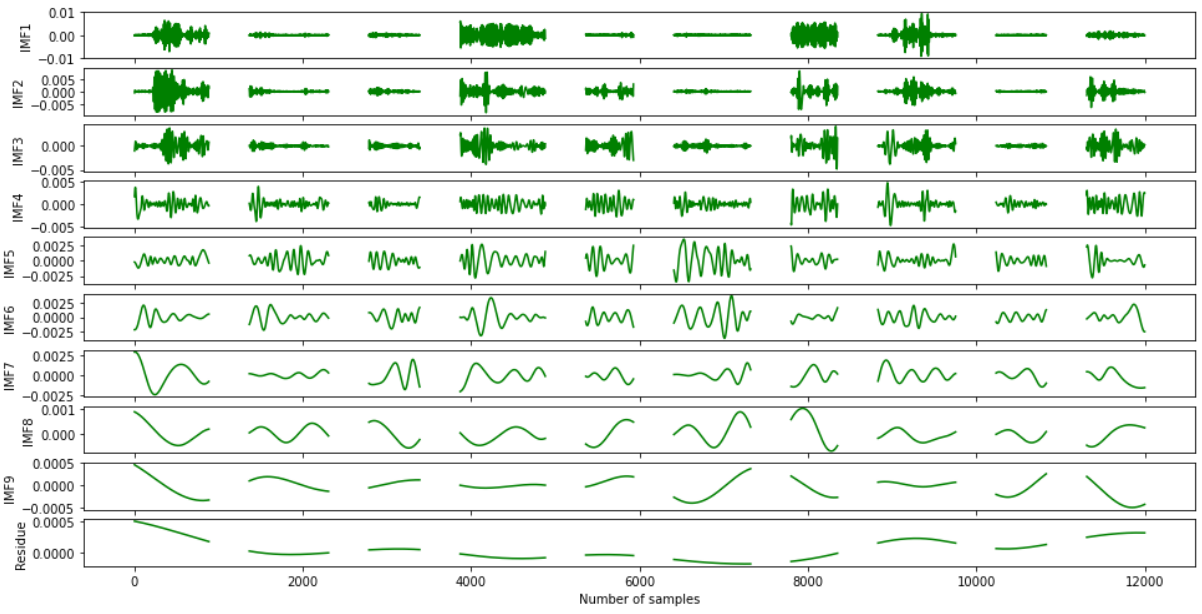

Figure 12: Boundary Estimation and Removal for Heart Signal

4.5 Predicting missing samples:

After removing the Heart Sounds, we are left with gaps in the Signal, as shown in the previous section. These gaps need to be filled according to the existing Lung Signal.

4.5.1 Using Transform domain (baseline method)

In order to predict the missing samples, we are performing FFT and IFFT operations on the signal. This is done by initially creating sub segments of the gap. We used 128 point FFT and IFFT operations on the data signal as the length of each gap is fixed at 480. The initial sub segment of the gap is predicted by performing FFT on 128 parts before the initial point. Similarly, the final sub-segment of the gap is predicted by applying FFT on 128 parts on the signal immediately after the final sub segment. This method is applied on either side of the gap as we inwards in the gap. The sub segments lying closer to the starting point will have information corresponding to the signals before starting.

The sub segment lying closer to the ending point of the gap will have information corresponding to the signal immediately after the ending point of the gap. This FFT predicted sample is converted to the time domain by performing an IFFT operation on them. This final converted signal is used for further applications of the project.

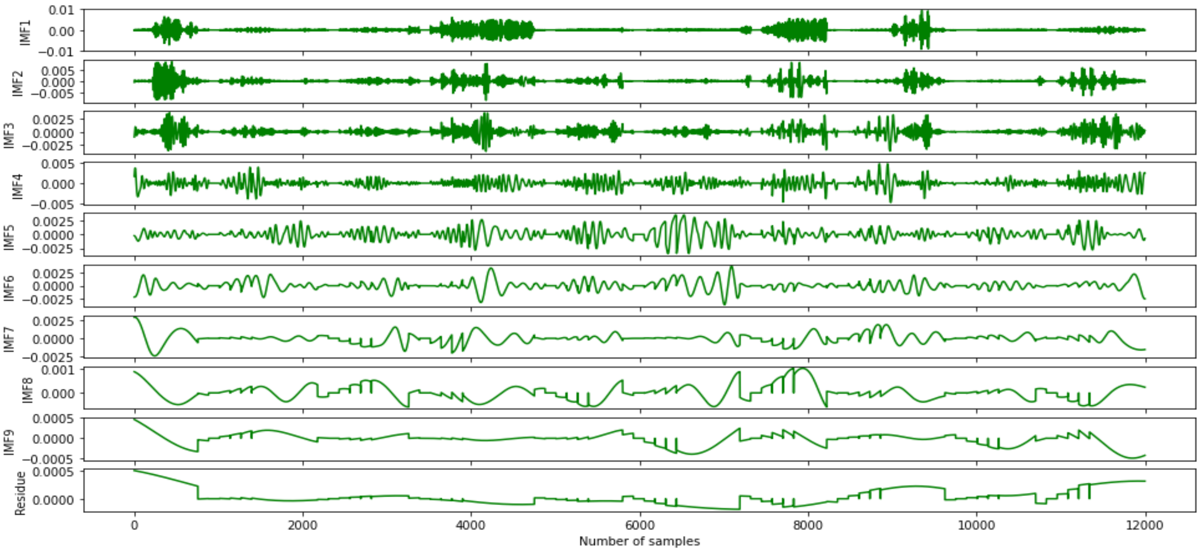

Figure 13: Predicting missing samples using transform technique

4.5.2 Using Interpolation

We used Interpolation techniques to bridge the gaps generated after removing the heart sound components. In its essence, interpolation imputes a number of values in a set of smaller values, considering the amplitude distribution of those smaller values. This is done by mathematical techniques that are comparable to linear regression.

In the project, we used the *Scipy.interp* Library and used 1d Interpolation as well as Univariate Spline Interpolation. This was done around the gaps in the sample data. 240 samples on either side of the gap were taken to arrive at new interpolated values. These were then replaced in the gaps to give us the following output:

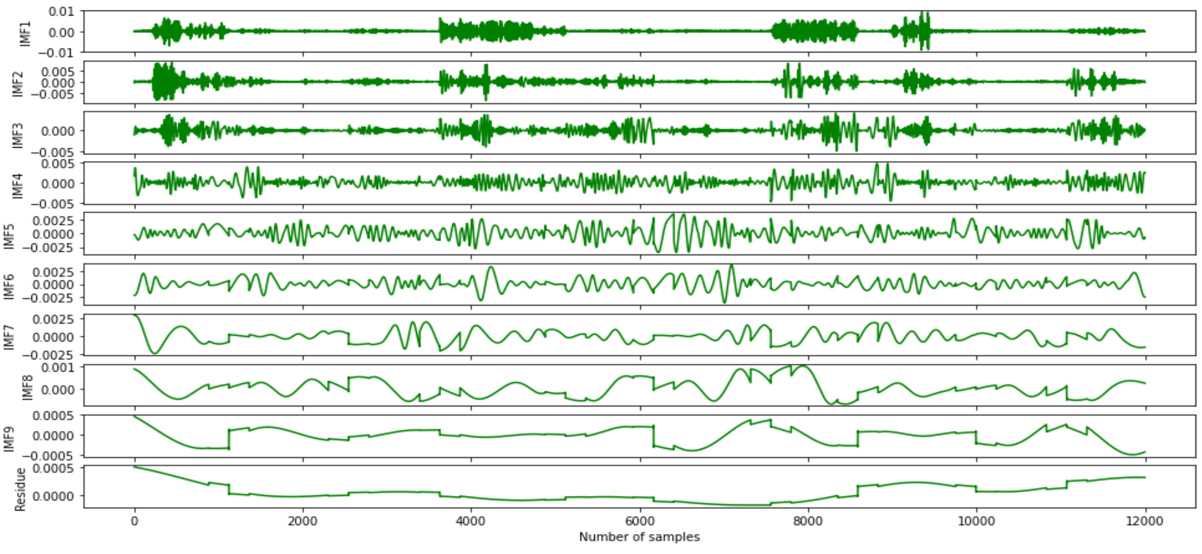

Figure 14: Interpolated values replaced in the missing samples

4.6 Reconstruction of Signal

After predicting missing samples, the individual IMFs are put together using the *PyEMD* Library to regenerate the final Lung Sound. At this step, we have separated the Heart Sounds entirely from the mixture of Heart and Lung Sounds given in the input signal. The resulting Lung Sound is ready for feature extraction and classification. The reconstructed signal from both the techniques, i.e., interpolation and transforms, are as below:

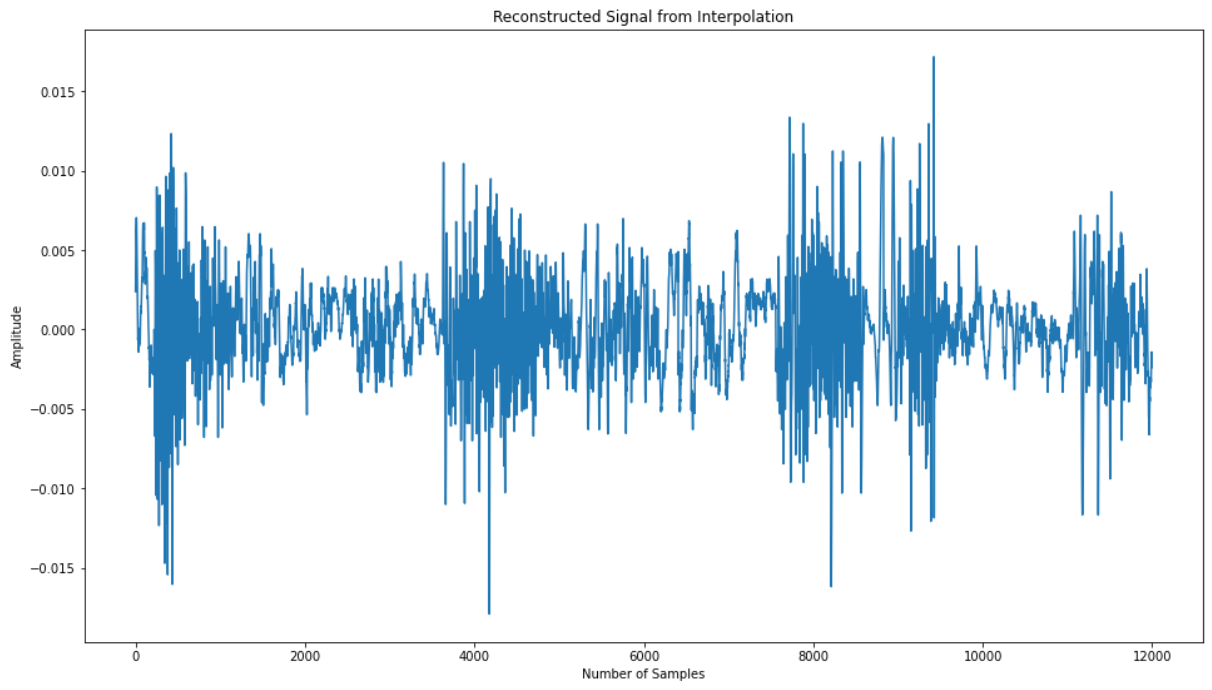

Figure 15: Reconstructed Signal from Interpolation

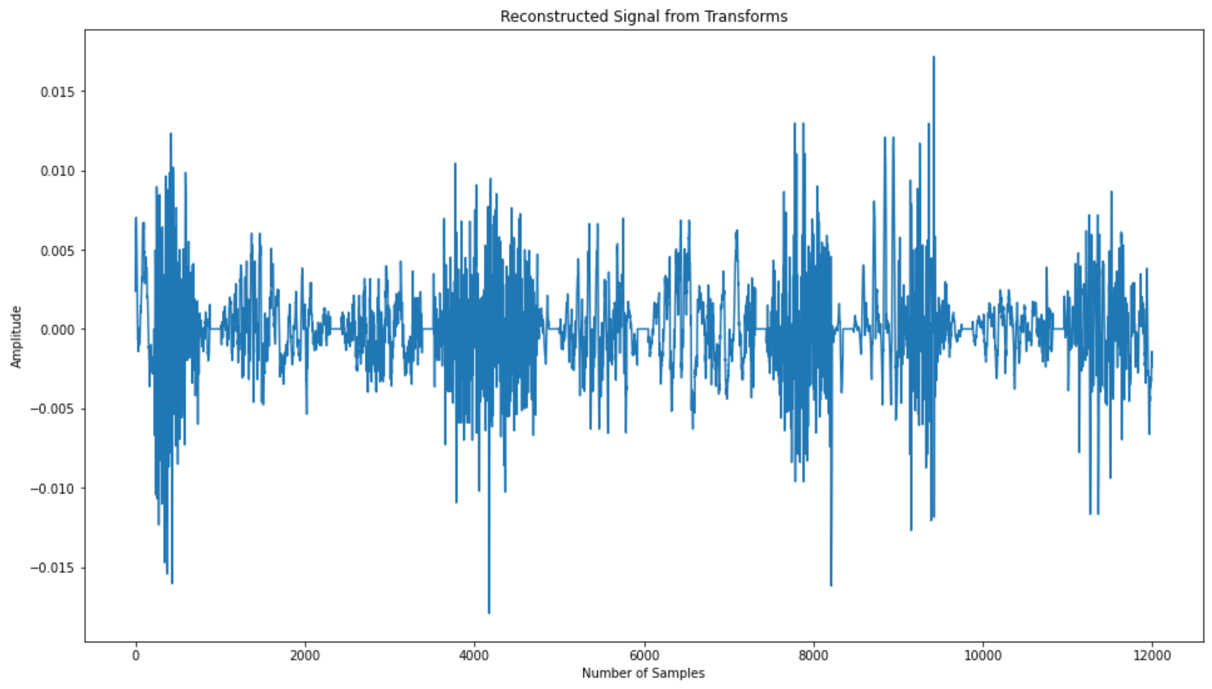

Figure 16: Reconstructed Signal from Transformation

4.7 Performance Evaluation:

Following metrics were used to evaluate the performance of the signal processing techniques used above:

1. Root Mean Square: RMSE implies the occurrence of undesired information in the denoised signal, and its value must be low for enhanced signal compared to noisy signal and is defined as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(x_p[n] - \hat{x}[n]\right)^2}$$

2. Signal to Noise Ratio(SNR): This is another widely used metric to check how efficiently noise is removed from the given signal.

   *SNR=10*log((signal power)/(noise power))*

   Lower RMSE and high SNR tell us that the method is performing well.

   With the above method, we were able to obtain an RMSE value of 0.0029736 with Interpolation and 0.0027852 using transform techniques(baseline method).

   While the transform domain(baseline method) gives an SNR as low as 1.4dB, the interpolation technique clearly stands better with an SNR of 17.8dB.

CHAPTER 5
FEATURE EXTRACTION

Feature extraction is a part of the dimensionality reduction process, so the initial set of the raw data is divided and reduced to manageable groups. So when you want to process, it will be much easier. In short, Feature extraction helps to get the best feature from those big data sets by selecting and combining variables into features, thus, effectively reducing the amount of data to process by the model.

These features are effortless to process, but still, they can describe the actual data set with high accuracy and originality. There are several ways of extracting features from the dataset. We during our project surveyed a few methods and settled on using Mel Frequency Cepstral Coefficient (MFCC) for feature extraction. MFCC is the best known and most efficient way of extracting features from a sound dataset.
To explain MFCC, we need to understand the usage of Spectrograms and Mel Frequencies as well, which are presented in the following subsections.

5.1 Spectrogram

Spectrograms, in general, are a visualization of a continuous signal with varying strength over a range of frequencies. Spectrograms of audio signals, for instance, have varying loudness which is represented on the Y-axis of the frequency with shaded colors. The X-axis has time.

The colors in a spectrogram represent the energy level of the signal. Spectrograms are 3-dimensional graphs, where the 2 basic dimensions are Frequency and Time. The 3rd Dimension is described in color (strength). Usually, the darker colors represent lower energy, and brighter colors represent higher energy concentration.

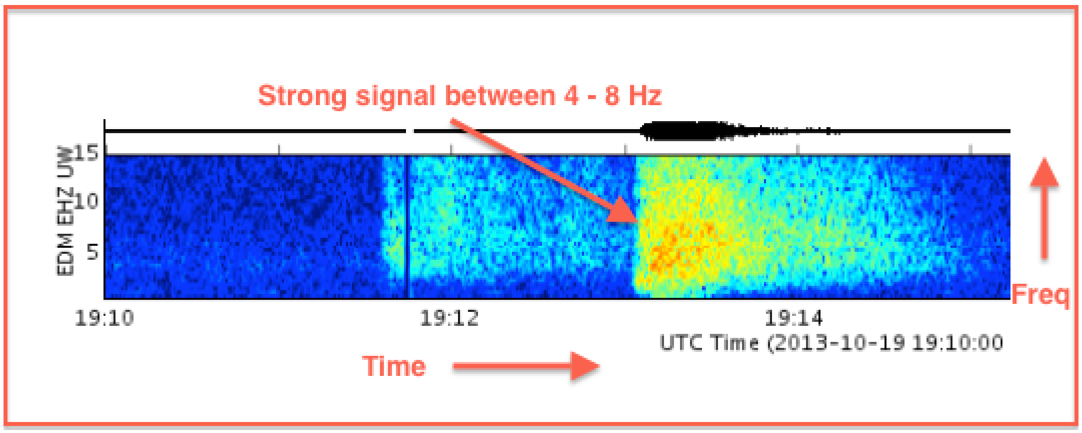

Figure 17: Example of a Spectrogram of an Audio Signal taken from[PNSN.org]

5.2 Mel Spectrogram

Mel Frequency is a particular frequency that is used to make up the Mel Scale. This scale defines closely what it is like, for the human ear to listen to sounds. Regular frequency in Hertz is converted to Mel Frequency to grasp a better understanding of how the human ear perceives it.

The plotting of these Mel frequencies in a spectrogram gives us the Mel Spectrogram. Mel Spectrograms help us analyze lower frequency signals in an effective manner, all the while maintaining the necessary data required to interpret the signal. The Logarithmic scale helps identify the compressions in the sound properly and enhances the features of the data.

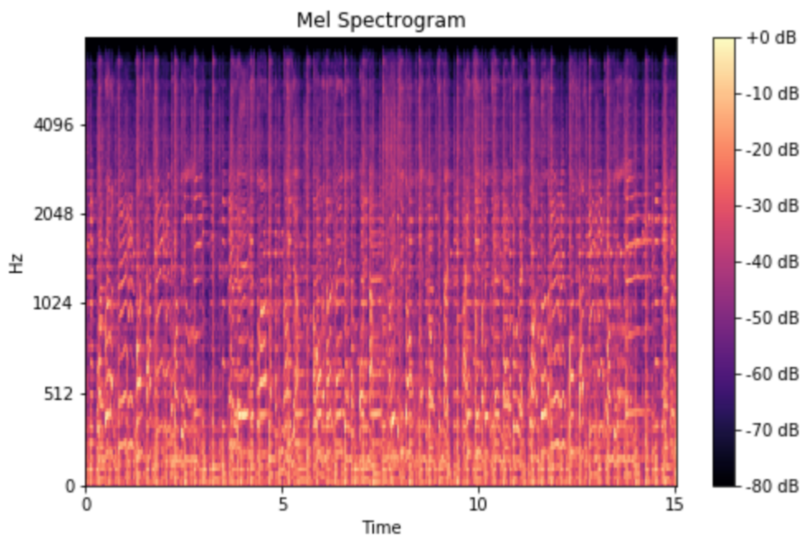

Figure 18: Example of Mel Spectrogram taken from here

The following formula is popularly used to convert frequencies into Mel scale[6]

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

5.3 Mel Frequency Cepstrum Coefficients (MFCC)

The Mel Frequency Cepstrum coefficients make up the Mel Frequency Cepstrum (MFC). This MFC is just another representation of the sound signal of the Mel Scale, but here cosines of the frequencies are also employed.

This further step simplifies the feature extraction from the data to a large extent. Mel frequency coefficients can be used to quantify sound data and use for classification by our algorithms.

Mel filter banks are used to generate the MFCCs. Using the triangular filter-bank helps capture the energy at each critical band, gives a rough approximation of the spectrum shape, and smooths the harmonic structure. In theory, you could manipulate raw DFT bins. Still, you are not reducing the dimensionality of your features - this is the whole point of doing filter-bank analysis to capture the spectral envelope.

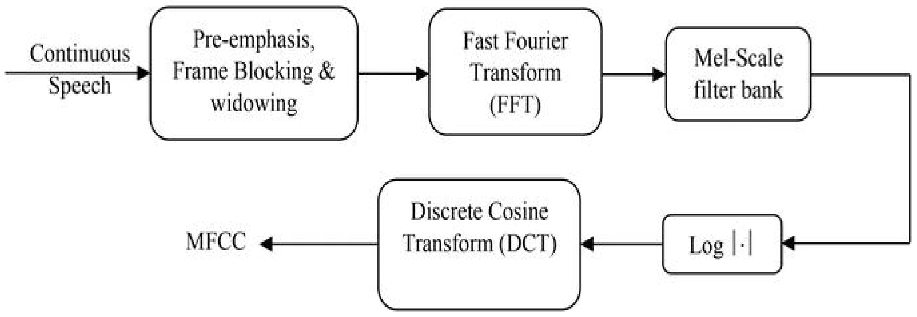
Continuous Speech
Pre-emphasis, Frame Blocking & widowing
Fast Fourier Transform (FFT)
Mel-Scale filter bank
Log |·|
Discrete Cosine Transform (DCT)
MFCC

Figure 19: Process of arriving at MFCCs

Librosa is the python library that is being used in the feature extraction process [a]. The audio files in consideration are 20 seconds in duration. The preprocessing on the a-file included applying a 6th order Butterworth filter followed by the data augmentation techniques on the files that are not labeled as “COPD”.

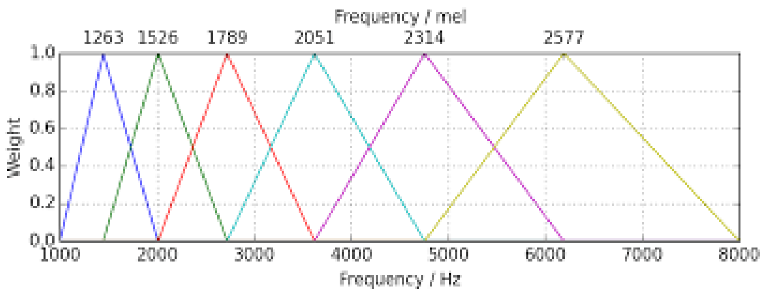

Figure 20: Mel Bands

CHAPTER 6
CLASSIFICATION USING DEEP LEARNING

6.1 Neural Network

A neural network may be a network OR circuit of neurons or a man-made neural network composed of artificial neurons or nodes in a modern sense. A neural network is either a biological neural network made from real biological neurons or an artificial neural network for solving Artificial Intelligence (AI) problems. The connections of the biological neuron are modeled as weights. A positive weight reflects an excitatory connection, while negative values mean inhibitory connections. All inputs are modified by weight and summed. This activity is mentioned as a linear combination. Finally, an activation function controls the amplitude of the output. For instance, a suitable range of output is typically between 0 and 1, or it might be −1 & 1.

These artificial networks could also be used for predictive modeling, adaptive control, and applications where they will be trained via a dataset. Self-learning resulting from experience can occur within networks, which may derive conclusions from a posh and seemingly unrelated set of data. .”In the artificial intelligence field, artificial neural networks are applied successfully to speech recognition, image analysis, and adaptive control to construct software agents (in computer and video games) or autonomous robots.

A neural network (NN), with respect to artificial neurons called artificial neural network (ANN) or simulated neural network (SNN), is an interconnected group of natural or artificial neurons that uses a mathematical or computational model for information processing based on a connectionist approach to computation. In most cases, an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network.

In many practical cases neural networks are non-linear statistical data modeling or decision-making tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. An artificial neural network involves a network of simple processing elements (artificial neurons) that can exhibit complex global behavior, determined by the connections between the processing and element parameters.

6.2 Convolutional Neural Network(CNN)

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks primarily used to analyze visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation equivariant responses known as feature maps.

The name “convolutional neural network” indicates that the network employs a mathematical operation called convolution. Convolutional networks are specialized neural networks that use convolution instead of general matrix multiplication in at least one of their layers.[10]

CNN’s are regularized versions of multilayer perceptrons. Multilayer perceptrons generally mean fully connected networks, i.e., each neuron in one layer is connected to all neurons in the next layer. The "full connectivity" of these networks makes them prone to overfitting data. Typical ways of regularization or preventing overfitting include: penalizing parameters during training (such as weight decay) or trimming connectivity (skipped connections, dropout, etc.) CNNs take a different approach towards regularization. They take advantage of the hierarchical pattern in data and assemble patterns of increasing complexity using smaller and simpler patterns embossed in their filters. So, based on connectivity and complexity, CNNs are on the lower extremity.

6.3 CNN Architecture

A convolutional neural network consists of an input layer, hidden layers, and an output layer. Any middle layers are called hidden in any feed-forward neural network because the activation function and final convolution mask their inputs and outputs. In a convolutional neural network, the hidden layers include layers that perform convolutions. Typically this consists of a layer that performs a dot product of the,,,, convolution kernel with the layer's input matrix. This product is usually the Frobenius inner product, and its activation function is commonly ReLU. As the convolution kernel slides along the input matrix for the layer, the convolution operation generates a feature map, which contributes to the input of the next layer. This is followed by other layers such as pooling layers, fully connected layers, and normalization layers.

Convolution layers

In a CNN, the input is a tensor with a shape: (number of inputs) x (input height) x (input width) x (input channels). After passing through a convolutional layer, the image becomes abstracted to a feature map, also called an activation map, with shape: (number of inputs) x (feature map height) x (feature map width) x (feature map channels).

Padding

When we do the convolution of the image with the filters/kernels, the output dimension gets reduced, and because of this, we lose some data. So, to overcome this problem, we introduce some supplemental zero boundaries to the convolution and then recalculate it to cover all of the input values.

Pooling

A small portion is referred to as pooling in general. Pooling can be classified into two categories. The first is average pooling, which utilizes the average value, and the second is max pooling, where we take the maximum value from the group of values.

Function of Activation

It's a node that appears at the end of or in the midst of a neural network. They assist in determining whether or not a neuron will fire. We have a variety of activation functions, but we have primarily focused on Rectified Linear Unit (ReLU) for the objectives of our project.

6.4 CNN Model

CNN Model We have used four convolution layers with activation function as “ReLU” with 64, 64, 96 and 96 filters respectively and there are 6 hidden layers with 256, 128, 64, 32, 16, and 8 neurons, respectively with dropouts of 60%, 30%, 15%, 7.5%, and 3.25% respectively. Finally, there is one output layer with three neurons as we are classifying three different types of diseases: chronic,, non-chronic, and healthy. The dropout layers used in between the hidden layers were to ensure that there is a balance between generalization and memorization. Dropout layers in the sequential model of Keras ensure that overfitting of the model is prevented. The proposed model is shown in Figure 4. The input to the CNN file is the MFCC coefficient of dimension (40x 862x1).

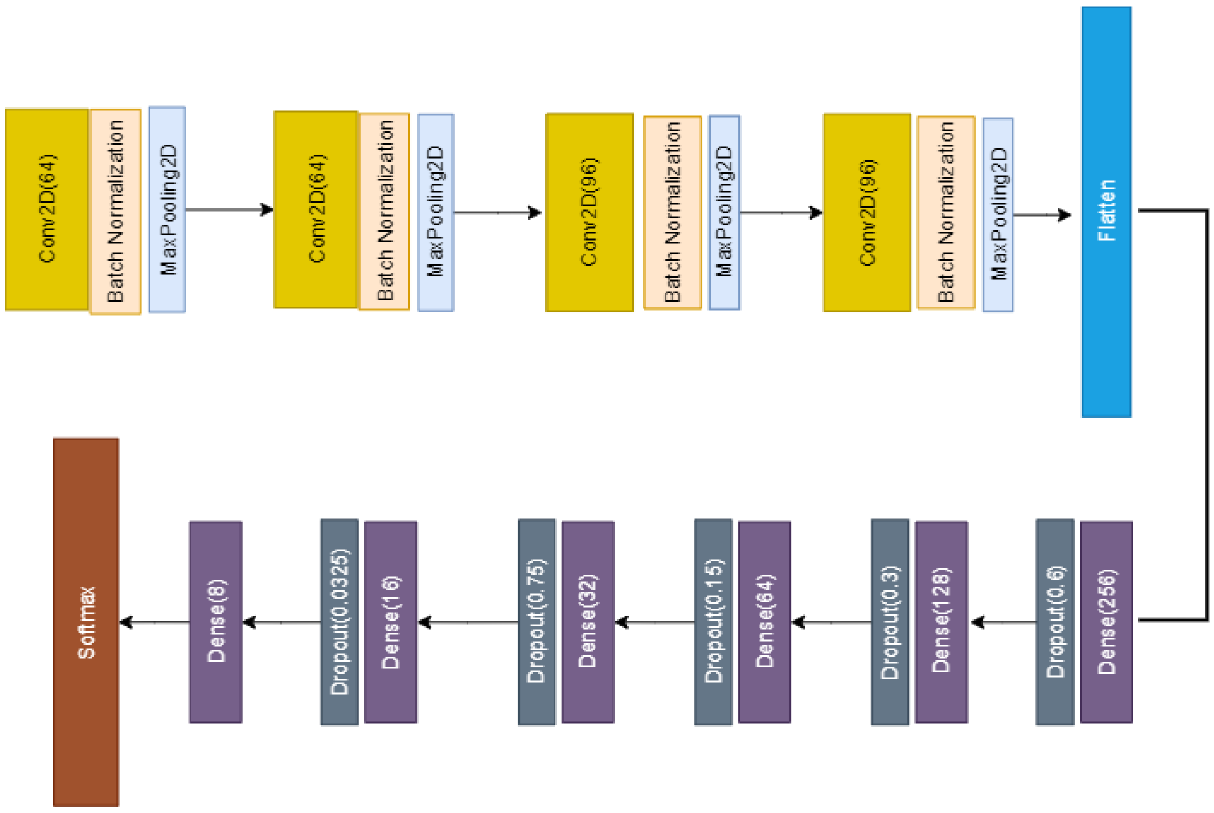

Figure 21: Proposed CNN Architecture

6.5 Proposed Models

The following set of models were worked upon to finally arrive at a desirable model that fit all the requirements and had highest accuracy.

6.5.1 Model 1

4 convolution layers with 16, 32,64 and 128 filters respectively and added dropouts of 20% on each layer. One dense layer with 6 output nodes.

Number of Epochs =250

```
Training Accuracy:  0.9263301491737366

Testing Accuracy:  0.875
```

Table 1: Summary of Classification Performance of Model 1

	precision	recall	f1-score	support
Bronchiectasis	0.00	0.00	0.00	3
Bronchiolitis	0.00	0.00	0.00	3
COPD	0.90	1.00	0.95	159
Healthy	0.00	0.00	0.00	7
Pneumonia	0.40	0.29	0.33	7
URTI	0.00	0.00	0.00	5
accuracy			0.88	184
macro avg	0.22	0.21	0.21	184
weighted avg	0.80	0.88	0.83	184

```
Confusion Matrix:

[[  0   0   3   0   0   0]
 [  0   0   3   0   0   0]
 [  0   0 159   0   0   0]
 [  0   0   4   0   1   2]
 [  0   0   5   0   2   0]
 [  0   0   2   1   2   0]]
```

6.5.2 Model 2

4 convolution layers with 16, 32,64 and 128 filters respectively and added dropouts of 20% on each layer. One dense layer with 6 output nodes.

Number of Epochs =125

```
Training Accuracy:  0.9249659180641174

Testing Accuracy:  0.8804348111152649
```

Table 2: Summary of Classification Performance of Model 2

	precision	recall	f1-score	support
Bronchiectasis	0.67	0.67	0.67	3
Bronchiolitis	0.33	0.33	0.33	3
COPD	0.96	0.96	0.96	159
Healthy	0.00	0.00	0.00	7
Pneumonia	0.50	0.71	0.59	7
URTI	0.17	0.20	0.18	5
accuracy			0.88	184
macro avg	0.44	0.48	0.45	184
weighted avg	0.87	0.88	0.87	184

```
Confusion Matrix:

[[  2   0   1   0   0   0]
 [  0   1   1   1   0   0]
 [  1   0 153   1   1   3]
 [  0   1   3   0   1   2]
 [  0   0   2   0   5   0]
 [  0   1   0   0   3   1]]
```

6.5.3 Model 3

3 convolution layers with 16, 32, and 64 filters respectively, and added dropouts of 20% on each layer. One dense layer with 6 output nodes.

Number of Epochs =250

```
Training Accuracy: 0.9085947871208191

Testing Accuracy: 0.885869562625885
```

Table 3: Summary of Classification Performance of Model 3

	precision	recall	f1-score	support
Bronchiectasis	1.0	0.67	0.80	3
Bronchiolitis	0.00	0.00	0.00	3
COPD	0.91	0.99	0.95	159
Healthy	0.00	0.00	0.00	7
Pneumonia	0.20	0.14	0.17	7
URTI	0.67	0.40	0.50	5
accuracy			0.89	184
macro avg	0.46	0.37	0.40	184
weighted avg	0.83	0.89	0.85	184

```
Confusion Matrix:

[[  2   0   1   0   0   0]
 [  0   0   3   0   0   0]
 [  0   0 158   0   1   0]
 [  0   0   3   0   3   1]
 [  0   0   6   0   1   0]
 [  0   0   3   0   0   2]]
```

6.5.4 Model 4

3 convolution layers with 16, 32 and 64 filters respectively and added dropouts of 20% on each layer. One Dense layer with 6 output nodes.

Number of Epochs =150

```
Training Accuracy:  0.9004092812538147

Testing Accuracy:  0.885869562625885
```

Table 4: Summary of Classification Performance of Model 4

	precision	recall	f1-score	support
Bronchiectasis	0.67	0.67	0.67	3
Bronchiolitis	0.50	0.33	0.40	3
COPD	0.93	0.97	0.95	159
Healthy	0.50	0.29	0.36	7
Pneumonia	0.33	0.43	0.38	7
URTI	0.00	0.00	0.00	5
accuracy			0.89	184
macro avg	0.49	0.45	0.46	184
weighted avg	0.86	0.89	0.87	184

```
Confusion Matrix:

[[  2   0   1   0   0   0]
 [  0   1   2   0   0   0]
 [  1   0 155   0   3   0]
 [  0   1   2   2   2   0]
 [  0   0   4   0   3   0]
 [  0   0   2   2   1   0]]
```

6.5.5 Model 5

4 convolution layers with 16, 32, 64 and 128 filters respectively and added dropouts of 20% on each layer. One dense hidden layer with 32 neurons and one output layer with 6 neurons.

Number of Epochs = 250

```
Training Accuracy:  0.9345157146453857

Testing Accuracy:  0.875
```

Table 5: Summary of Classification Performance of Model 5

	precision	recall	f1-score	support
Bronchiectasis	0.00	0.00	0.00	3
Bronchiolitis	0.00	0.00	0.00	3
COPD	0.93	0.97	0.95	159
Healthy	0.00	0.00	0.00	7
Pneumonia	0.42	0.71	0.53	7
URTI	0.00	0.00	0.00	5
accuracy			0.88	184
macro avg	0.27	0.31	0.28	184
weighted avg	0.82	0.88	0.85	184

```
Confusion Matrix:

[[  0   0   1   0   1   1]
 [  0   0   3   0   0   0]
 [  0   1 155   0   3   0]
 [  0   0   4   0   1   2]
 [  0   0   2   0   5   0]
 [  0   0   2   0   2   1]]
```

6.5.6 Model 6

4 convolution layers with 16, 32, 64 and 128 filters respectively and added dropouts of 20% on each layer. One dense hidden layer with 32 neurons and one output layer with 6 neurons.

Number of Epochs = 125

```
Training Accuracy:  0.9045020341873169
```

```
Testing Accuracy:  0.8586956262588501
```

Table 6: Summary of Classification Performance of Model 6

	precision	recall	f1-score	support
Bronchiectasis	0.00	0.00	0.00	3
Bronchiolitis	0.00	0.00	0.00	3
COPD	0.89	0.98	0.93	159
Healthy	0.00	0.00	0.00	7
Pneumonia	0.40	0.29	0.33	7
URTI	0.00	0.00	0.00	5
accuracy			0.86	184
macro avg	0.22	0.21	0.21	184
weighted avg	0.79	0.86	0.82	184

```
Confusion Matrix:

[[  0   0   3   0   0   0]
 [  0   0   2   0   0   1]
 [  0   0 156   0   1   2]
 [  0   0   5   0   1   1]
 [  0   0   5   0   2   0]
 [  0   0   4   0   1   0]]
```

6.5.7 Model 7

4 convolution layers with 16, 32, 64 and 128 filters respectively and added dropouts of 20% on each layer. One dense hidden layer with 64 neurons and one output layer with 6 neurons.

Number of Epochs = 250

```
Training Accuracy:  0.9154161214828491

Testing Accuracy:  0.8695651888847351
```

Table 7: Summary of Classification Performance of Model 7

	precision	recall	f1-score	support
Bronchiectasis	0.00	0.00	0.00	3
Bronchiolitis	0.00	0.00	0.00	3
COPD	0.88	1.00	0.94	159
Healthy	0.00	0.00	0.00	7
Pneumonia	0.00	0.00	0.00	7
URTI	0.33	0.20	0.25	5
accuracy			0.87	184
macro avg	0.20	0.20	0.20	184
weighted avg	0.77	0.87	0.82	184

```
Confusion Matrix:

[[  0   0   3   0   0   0]
 [  0   0   3   0   0   0]
 [  0   0 159   0   0   0]
 [  0   0   5   0   0   2]
 [  0   0   7   0   0   0]
 [  0   0   3   1   0   1]]
```

6.5.8 Model 8

4 convolution layers with 16, 32, 64 and 128 filters respectively and added dropouts of 20% on each layer. One dense hidden layer with 64 neurons and one output layer with 6 neurons.

Number of Epochs = 125

```
Training Accuracy:  0.8976807594299316

Testing Accuracy:  0.875
```

Table 8: Summary of Classification Performance of Model 8

	precision	recall	f1-score	support
Bronchiectasis	1.00	0.67	0.80	3
Bronchiolitis	0.33	0.33	0.33	3
COPD	0.90	0.98	0.94	159
Healthy	0.00	0.00	0.00	7
Pneumonia	1.00	0.14	0.25	7
URTI	0.25	0.20	0.22	5
accuracy			0.88	184
macro avg	0.58	0.39	0.42	184
weighted avg	0.85	0.88	0.85	184

Confusion Matrix

```
[[  2   0   1   0   0   0]
 [  0   1   1   0   0   1]
 [  0   0 156   1   0   2]
 [  0   1   6   0   0   0]
 [  0   0   6   0   1   0]
 [  0   1   3   0   0   1]]
```

6.5.9 Model 9

4 convolution layers with 16, 32, 64 and 128 filters respectively and added dropouts of 20% on each layer. One dense hidden layer with 128 neurons and one output layer with 6 neurons.

Number of Epochs = 250

```
Training Accuracy:  0.9386084675788879

Testing Accuracy:  0.8913043737411499
```

Table 9: Summary of Classification Performance of Model 9

	precision	recall	f1-score	support
Bronchiectasis	0.50	0.33	0.40	3
Bronchiolitis	0.00	0.00	0.00	3
COPD	0.92	0.99	0.96	159
Healthy	1.00	0.14	0.25	7
Pneumonia	0.50	0.29	0.36	7
URTI	0.40	0.40	0.40	5
accuracy			0.89	184
macro avg	0.55	0.36	0.40	184
weighted avg	0.87	0.89	0.87	184

```
Confusion Matrix:

[[  1   0   1   0   0   1]
 [  0   0   3   0   0   0]
 [  1   0 158   0   0   0]
 [  0   0   3   1   1   2]
 [  0   0   5   0   2   0]
 [  0   1   1   0   1   2]]
```

6.5.10 Model 10

4 convolution layers with 16, 32, 64 and 128 filters respectively and added dropouts of 20% on each layer. One dense hidden layer with 128 neurons and one output layer with 6 neurons.

Number of Epochs = 1

```
Training Accuracy:  0.8813096880912781

Testing Accuracy:  0.8478260636329651
```

Table 10: Summary of Classification Performance of Model 10

	precision	recall	f1-score	support
Bronchiectasis	0.00	0.00	0.00	3
Bronchiolitis	0.17	0.67	0.27	3
COPD	0.90	0.97	0.93	159
Healthy	0.00	0.00	0.00	7
Pneumonia	0.00	0.00	0.00	7
URTI	0.00	0.00	0.00	5
accuracy			0.85	184
macro avg	0.18	0.27	0.20	184
weighted avg	0.78	0.85	0.81	184

Confusion Matrix

```
[[  0   3   0   0   0   0]
 [  0   2   1   0   0   0]
 [  0   5 154   0   0   0]
 [  0   1   6   0   0   0]
 [  0   0   7   0   0   0]
 [  0   1   4   0   0   0]]
```

6.5.11 Model 11

4 convolution layers with 16, 32, 64 and 128 filters respectively and added dropouts of 20% on each layer. One dense hidden layer with 256 neurons and one output layer with 6 neurons.

Number of Epochs = 250

```
Training Accuracy:  0.9781718850135803

Testing Accuracy:  0.907608687877655
```

Table 11: Summary of Classification Performance of Model 11

	precision	recall	f1-score	support
Bronchiectasis	0.50	0.33	0.40	3
Bronchiolitis	0.00	0.00	0.00	3
COPD	0.97	0.99	0.97	159
Healthy	0.60	0.43	0.50	7
Pneumonia	0.56	0.71	0.63	7
URTI	0.20	0.20	0.20	5
accuracy			0.91	184
macro avg	0.47	0.44	0.45	184
weighted avg	0.90	0.91	0.90	184

```
Confusion Matrix:
[[  1   0   1   0   1   0]
 [  1   0   0   0   0   2]
 [  0   0 157   0   2   0]
 [  0   0   2   3   0   2]
 [  0   1   1   0   5   0]
 [  0   0   1   2   1   1]]
```

6.5.12 Model 12

4 convolution layers with 16, 32, 64, and 128 filters, respectively, and added dropouts of 20% on each layer. One dense hidden layer with 256 neurons and one output layer with 6 neurons.

Number of Epochs = 125

```
Training Accuracy:  0.9222373962402344

Testing Accuracy:  0.885869562625885
```

Table 12: Summary of Classification Performance of Model 12

	precision	recall	f1-score	support
Bronchiectasis	0.67	0.67	0.67	3
Bronchiolitis	0.27	1.00	0.43	3
COPD	0.94	0.97	0.96	159
Healthy	1.00	0.14	0.25	7
Pneumonia	0.50	0.29	0.57	7
URTI	0.00	0.00	0.25	5
accuracy			0.89	184
macro avg	0.56	0.51	0.44	184
weighted avg	0.88	0.89	0.87	184

```
Confusion Matrix:

[[  2   1   0   0   0   0]
 [  0   3   0   0   0   0]
 [  1   2 155   0   1   0]
 [  0   3   3   1   0   0]
 [  0   0   5   0   2   0]
 [  0   2   2   0   1   0]]
```

6.5.13 Model 13

4 convolution layers with 32, 64, 128, and 256 filters, respectively, and added dropouts of 20% on each layer. One dense hidden layer with 256 neurons and one output layer with 6 neurons.

Number of Epochs = 250

```
Training Accuracy:  0.9754433631896973


Testing Accuracy:  0.89673912525177
```

Table 13: Summary of Classification Performance of Model 13

	precision	recall	f1-score	support
Bronchiectasis	1.00	0.33	0.50	3
Bronchiolitis	0.00	0.00	0.00	3
COPD	0.92	0.99	0.96	159
Healthy	0.50	0.14	0.22	7
Pneumonia	0.57	0.57	0.57	7
URTI	0.33	0.20	0.25	5
accuracy			0.90	184
macro avg	0.55	0.37	0.42	184
weighted avg	0.86	0.90	0.87	184

```
Confusion Matrix:

[[  1   0   1   0   1   0]
 [  0   0   2   1   0   0]
 [  0   0 158   0   1   0]
 [  0   0   4   1   0   2]
 [  0   0   3   0   4   0]
 [  0   0   3   0   1   1]]
```

6.5.14 Model 14

4 convolution layers with 32, 64, 128, and 256 filters, respectively, and added dropouts of 20% on each layer. One dense hidden layer with 256 neurons and one output layer with 6 neurons.

Number of Epochs = 125

```
Training Accuracy:  0.9699863791465759

Testing Accuracy:  0.885869562625885
```

Table 14: Summary of Classification Performance of Model 14

	precision	recall	f1-score	support
Bronchiectasis	1.00	0.33	0.50	3
Bronchiolitis	0.250	0.33	0.29	3
COPD	0.96	0.96	0.96	159
Healthy	0.25	0.14	0.18	7
Pneumonia	0.60	0.86	0.71	7
URTI	0.20	0.20	0.20	5
accuracy			0.89	184
macro avg	0.54	0.54	0.47	184
weighted avg	0.88	0.89	0.88	184

```
Confusion Matrix:

[[  1   0   1   1   0   0]
 [  0   1   0   1   0   1]
 [  0   0 153   1   4   1]
 [  0   1   3   0   0   2]
 [  0   0   1   0   6   0]
 [  0   2   2   0   0   1]]
```

6.5.15 Final Model 15

After training 14 different models, model imbalance issue was considered, and signal processing using 6th order Butterworth filter was done. Data augmentation techniques were used to counter model imbalance, and the classification was changed to report the chronicity of the disease rather than the actual disease. The model has 4 convolution layers with 64,64, 96, and 96 filters, respectively. 6 hidden layers with 256, 128, 64, 32, 16, and 8 neurons respectively with dropouts of 60%, 30%, 15%, 7.5% and 3.25% respectively. One output layer with 3 neurons.

Number of Epochs= 120

```
Training Accuracy:  0.9709020256996155

Testing Accuracy:  0.8914728760719299
```

Table 15: Summary of Classification Performance of Model 15

	precision	recall	f1-score	support
Chronic	0.91	0.97	0.94	159
Healthy	0.81	0.75	0.78	28
Non-chronic	0.89	0.76	0.82	71
accuracy			0.89	258
macro avg	0.87	0.83	0.85	258
weighted avg	0.89	0.89	0.89	258

```
Confusion Matrix:

 [[155   0   4]
  [  4  21   3]
  [ 12   5  54]]
```

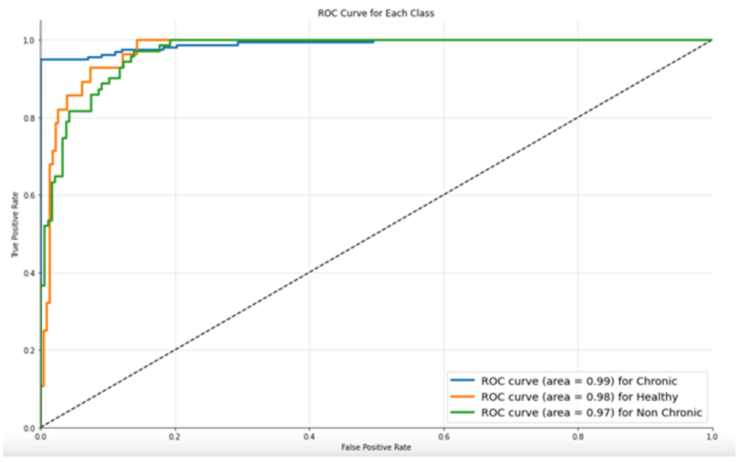

Figure 22: ROC Curve

6.6 Performance Evaluation:

The model initially showed a pre-training accuracy of 61.6279%. The time taken to train the model over 120 epochs was thirty-six minutes and fourteen seconds. Post training, the model showed a training accuracy of 97.09% and a testing accuracy of 89.15%.The model showed the least training loss of 0.0622 at the 104th epoch. The results for ternary classification were better than most models proposed in the previous related works.The ROC curves were plotted for the three output classes, and the areas for chronic, healthy, and non-chronic diseases were 0.99, 0.98, and 0.97, respectively.

Table 16: Comparison of Classifiers

Methodology	Accuracy
MFCC, Hidden Markov model	39.56%
Low level feature, decision tree	49.62%
STFT + Wavelet, SVM classifier	59.88%
Deep Feature with CNN model, and SVM classifier	65.50%
Transfer learning with CNN Model, and SoftMax classifier	63.09%
MFCC features based CNN classifier(Proposed Model)	89.15%

CHAPTER 7
CONCLUSION AND FUTURE SCOPE

In this work, we have proposed the implementation of a Smart digital stethoscope for pulmonary disease diagnosis. A hybrid approach employing both preprocessing techniques and CNN(Convolution Neural Network) model is used to classify noisy lung sounds into different categories. By decomposing the signal components in terms of IMF components, the EMD method allowed for a more comprehensive examination of the signal. After obtaining the Hilbert envelope and processing it further, HS-included segments in the LS record are determined and are then eliminated from each IMF component. Following that, an univariate spline interpolation is used to estimate the missing values of the LS signal corresponding to the produced gap. This method provides an improved signal to noise ratio of 18.3dB as opposed to the baseline method that gives only 3dB after removing the heart sounds from the noisy sample.Further, an approach using the short-time Fourier transform and the cosine transform to obtain MFCCs is extensively discussed. The publicly available ICBHI 2017 data set has been used to classify the diseases according to their chronicity. The proposed model has given a reliable accuracy of 89.15% for ternary chronic classification. The existing model for LS denoising can be further improved with better prediction techniques developed for time-series prediction for non-stationary signals. One of the most critical characteristics that individuals seek in this pandemic-stricken world is getting accurate findings quickly. As Covid-19 continues to claim lives daily, faster and more efficient approaches for detecting the disease in its early stages are critical. Given that the model is trained in that specific direction, the proposed model can also have more extensive and significant usage in terms of consistency and accuracy. We feel that implementing the model is an important step forward in the development of portable technology that can save countless lives.

BIBLIOGRAPHY

1. Performance Improvement of Ensemble Empirical Mode Decomposition for Roller Bearings Damage Detection (hindawi.com)
2. https://www.clear.rice.edu/elec301/Projects02/empiricalMode/app.html
3. https://www.researchgate.net/publication/221534245_Empirical_Mode_Decompositio n_-_an_introduction
4. https://www.weisang.com/en/documentation/envelopeanalysis_en/#
5. https://link.springer.com/chapter/10.1007%2F978-1-59259-835-9_15
6. https://pnsn.org/spectrograms/what-is-a-spectrogram
7. https://en.wikipedia.org/wiki/Mel_scale
8. https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html
9. Neural network - Wikipedia
10. Convolutional neural network - Wikipedia
11. Enhancement of lung sounds based on empirical mode decomposition and Fourier transform algorithm Ashok Mondal 1, Poulami Banerjee 2, Ajay Somkuwar Comput Methods Programs Biomed 2017 Feb;139:119-136.
12. A. Mondal, P. Bhattacharya, G. Saha, An automated tool for localization of heart sound components S1, S2, S3 and S4 in pulmonary sounds using Hilbert transform and Heron's formula, Springerplus 2 (1) (2013) 1–14.
13. Lung sounds classification using convolutional neural networks *by* Dalal Bardoua, Kun Zhanga and Sayed Mohammad Ahmad - Elsevier.Artificial Intelligence in medicine 88(2018) 58-69
14. A Lightweight CNN Model for Detecting Respiratory Diseases from Lung Auscultation Sounds using EMD-CWT-based Hybrid Scalogram" Samiul Based Shuvo, Shams Nafisa Ali, Soham Irtiza Swapil, Taufiq Hasan (Member IEEE) and Mohammed Imamul Hassan Bhuiyan (Member IEEE)