

AUTOMATIC ANALYSIS AND CLASSIFICATION OF HEART SOUNDS

Submitted in the partial fulfilment of the requirements for the award of Bachelor of
Engineering in Electronics and Communication Engineering.

By

G. GOPI VISWANADH CHOWDARY (39130150)

G. HARSHITHA HARINDRA (39130153)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

SCHOOL OF ELECTRICAL & ELECTRONICS

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119.**

APRIL-2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

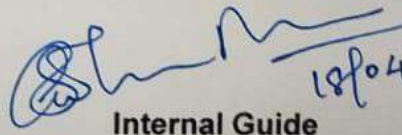
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

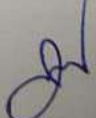
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

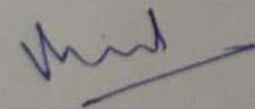
This is to certify that this Project Report is the Bonafide work of G. Harshitha Harindra (39130153) and G. Gopi Viswanadh Chowdary (39130150) who carried out the project entitled "**Automatic Analysis and Classification Of Heart Sounds**".

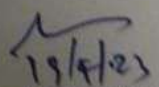

18/04/2023
Internal Guide

Dr. M. S. GODWIN PREMI, M.E., PhD


Head of the Department
Dr. T. RAVI M.E., Ph.D.,

Submitted for Viva voce Examination held on 19/4/23


Internal Examiner


19/4/23
External Examiner

DECLARATION

We, G. HARSHITHA HARINDRA (39130153) and G. GOPI VISWANADH CHOWDARY (39130150) hereby declare that the Project Report entitled "Automatic Analysis and Classification of Heart Sounds" done by us under the guidance of Dr. M.S. GODWIN PREMI, M.E., Ph.D., Professor of Department of Electronics and Communication Engineering is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

Chennai

18-04-2023

1. G. G. V. chowdary
2. G. H. Harish

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N.M. NANDHITHA, M.E., Ph.D.**, Dean, School of Electrical and Electronics Engineering and **Dr. T. RAVI, M.E., Ph.D.**, Head of the Department, Dept. of Electronics and Communication Engineering for providing me necessary support and details at the right time during the progressive reviews.

We would like to express my sincere and deep sense of gratitude to our Project guide **Dr. M. S. GODWIN PREMI M.E., Ph.D.**, Professor. Department of Electronics and Communications Engineering for her valuable guidance, suggestions, and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the department of Electronics and Communication Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

Digitalization of health care means that patient examinations from a distance are more realistic than ever. The automatic classification of a patient-recorded audio segment of its heartbeats as either healthy or pathological would be one of these examinations. This thesis investigates the various types of neural networks that can be used to accomplish this; convolutional brain organizations (CNN) and long present moment memory organizations (LSTM). The hypothesis of fake brain networks is made sense of. The feed forward CNN and recurrent LSTM-network's methods are described on this foundation. The pre-processing that needs to be done before these methods can be used is different for the two kinds of networks. Utilizing this hypothesis, the course of how to execute the organizations in MATLAB is made sense of. Diverse CNN: s is contrasted with one another, then, at that point, the best performing CNN is contrasted with the LSTM-organization. Cross validation is used to get the most accurate result possible when comparing the two distinct networks. The accuracy, shortest amount of training time, and smallest amount of training data are used to compare the networks. A discussion is included to explain the results, and a result is shown to demonstrate which kind of network performs best. In every way, the CNN performed better than the LSTM-network. The post includes a reflection on what could have been done differently to get a better result.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	v
	LIST OF FIGURES	viii
1	INTRODUCTION	1
	1.1 Introduction of Cardiovascular Disease	1
	1.2 Deep Learning	2
	1.3 Classification of Heart Sound	3
2	LITERATURE SURVEY	4
	2.1 Intelligent Systems for Heart Sound Signal Analysis	4
	2.2 heart disease and Stroke Statistics	4
	2.3 Telehealth in Cardiovascular and Stroke Care	5
	2.4 Portable Microfluidic and Smartphone-Based Devices For Monitoring	6
	2.5 Digital Monitoring and Care: Virtual Medicine	6
	2.6 Artificial Neural Network Approach for Paediatrics	7
	2.7 Heart Sounds Using an Artificial Neural Network	7
	2.8 Coronary Dysfunction Using Neural Networks	8
	2.9 Neural Network Assisted Cardiac Auscultation	8
	2.10 Non-invasive Detection of Coronary Artery Disease	8
	2.11 Problem Definition	9
	2.12 Objective of The Project	9
3	EXISTING SYSTEM	10
	3.1 Existing Methodology	10
	3.2 Block Diagram	11
	3.3 Disadvantages	11
4	PROPOSED SYSTEM	12
	4.1 Proposed Methodology	12
	4.2 Proposed Algorithms CNN Algorithm	13
	4.3 Block Diagram	18
	4.4 Flow Chart	18
	4.5 Advantages	19

4.6	Software Environment	19
4.7	Modules	21
4.7.1	Data Acquisition	21
4.7.2	Data Pre-processing	21
4.7.3	Segmentation	22
4.7.4	Feature Selection and Extraction	22
5	EXPERIMENTAL OR MATERIALS AND METHODS	23
5.1	CNN Algorithm	23
5.2	SVM Algorithm	24
5.2.1	Wavelet Packet Theory	25
5.2.2	Definition of Wavelet Packet	25
5.2.3	Wavelet Packet Basis Function	25
5.2.4	Wavelet Packet Energy Entropy	26
5.3	System Design	26
5.4	Module Used in The Heart Beat Sound Classification	28
5.4.1	Databases for Heart Sounds Analysis	28
5.4.2	Heart Sounds Segmentation	29
5.4.3	Feature Extraction	29
5.4.4	Classification of Heart Sounds	30
5.5	Signal Processing Using Python	30
5.6	EEG Signal Processing	32
6	RESULT, DISCUSSION AND PERFORMANCE ANALYSIS	36
7	SUMMARY AND CONCLUSION	37
8	REFERENCES	38
9	APPENDIX	39
9.1	Source Code	39
9.2	Code Output	48

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
3.2.1	Predictive Algorithm Block Diagram	11
4.2.1	Convolutional Neural Network	13
4.2.2	Filter Matrix	14
4.2.3	Strides	15
4.2.4	Padding	15
4.2.5	Max Pooling	16
4.2.6	Fully Connected Layer	17
4.3.1	Block Diagram	18
4.3.2	Flow Chart	18
5.1.1	CNN Flow Chart	24
6.1	Heart sound classification Flow Chart	36
9.2.1	Import Data	48
9.2.2	Data Imported	49
9.2.3	Pre-Process Data	49
9.2.4	Train Data	50
9.2.5	Heart Beat Graph	50

CHAPTER 1

INTRODUCTION

1.1 Introduction of Cardiovascular Disease:

Cardiovascular diseases (CVDs) are the leading cause of mortality worldwide resulting in over 17.7 million deaths each year. This number is predicted to increase to approximately 23 million per year by 2030. Apart from the personal consequences, the high prevalence and cost of cardiovascular diseases constitute a serious social and financial burden. As an illustration, 85 million Americans suffer from cardiovascular diseases resulting in an approximate healthcare cost of \$320 billion annually, with a projected increase to nearly \$1 trillion by 2030. While the estimated number of cardiac patients and health care costs are too high, an important thing to consider is that most cardiovascular diseases are preventable and curable. However, this requires early-stage diagnosis and proper disease management. Consequently, there is an urgent need to improve technologies to intensively monitor and analyse physiological parameters related to cardiac function in a timely and cost-effective manner. With recent evolution of mobile technologies, there is a growing, justified, interest on finding ways to continuously track the cardiovascular system for long periods of time, as a potentially more effective way to both diagnose and manage cardiac conditions. In literature, both invasive and non-invasive approaches for monitoring the cardiovascular system using different sensing schemes have been investigated. However, some of these approaches are not suitable for long-term continuous real time monitoring of cardiac signals in unsupervised environments, which would, however, be the optimum way of monitoring/managing some cardiac conditions. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. Recent advancements in computing together with the ever-shrinking size of electronic devices have enabled the design of wearable devices loaded with sensors that can perform the task of long-term continuous monitoring and have the potential of facilitating timely medical interventions for treatment and care. Wearables have the advantage of usability. Thus, wearables can allow self- health monitoring and save the time required for clinical appointments. This is why wearables have attracted a lot of attention from

scientists in this field. Though, potentially, available cardiac wearables can assist in real time monitoring, it is challenging to obtain a high degree of accuracy, especially under varying environmental conditions. Furthermore, in some cases, algorithms for signal interpretation have been validated with a limited database and hence their clinical reliability and diagnostic accuracy cannot be extrapolated for real clinical applications.

1.2 Deep Learning:

A heart sound is a kind of physiological signal, and its measurement is known as phonocardiography (PCG). It is produced by the heart systole and diastole and can reflect physiological information regarding body components such as the atria, ventricles, and large vessels, as well as their functional states. In general, fundamental heart sounds (FHSs) can be classified as the first heart sounds and the second heart sounds, referred to as S1 and S2, respectively. S1 usually occurs at the beginning of isovolumetric ventricular contraction, when the already closed mitral and tricuspid valves suddenly reach their elastic limit due to the rapid pressure increase within the ventricles. S2 occurs at the beginning of the diastole when the aortic and pulmonic valves close.

ECGs provide a graphic representation of the electrical activity of the heart muscle. This study evaluates ECG classification using pre-processing routines, feature extraction and Selection, and ECG beat classification using MSVM and CSVM. DCT and DST coefficients were used to extract features presented at the input Vector of the MSVM classifier whereas DFT coefficients are used for creating an input vector to the CSVM classifier. The MSVM and CSVM classifier is used to distinguish the four ECG arrhythmias types using the DCT, DST and DFT coefficients as feature vectors at the input vector of the classifier. Sequential minimal optimization (SMO) approach is used to train the CSVM and compute the corresponding complex hyper-plane parameters.

Cardiovascular illnesses (CVDs) are the main source of mortality overall bringing about over 17.7 million passings every year. By 2030, it is anticipated that this number will rise to approximately 23 million per year. The high prevalence and high cost of cardiovascular diseases represent a significant social and financial burden in addition to the personal consequences. For instance, 85 million Americans suffer from cardiovascular diseases, which account for approximately \$320 billion in annual healthcare costs and are projected to rise to nearly \$1 trillion by 2030.

Although the estimated number of cardiac patients and healthcare costs are excessive, it is essential to keep in mind that the majority of cardiovascular diseases can be prevented and treated. However, this necessitates accurate disease management and early diagnosis.

1.3 Classification of Heart Sound:

Numerous calculations have been accounted for computerized characterization of heart sounds with approaches that reach from Conventional thresholding techniques to late measurable machine learning and brain network-based ones. The primary point of programmed heart sounds examination is to accomplish an exact order of the obsessive occasions present in the cardiovascular cycle.

As a result, there is an urgent need to improve technologies that allow for prompt and cost-effective intensive monitoring and analysis of physiological parameters related to cardiac function. Finding ways to continuously track the cardiovascular system over extended periods of time as a potentially more effective method for both diagnosing and managing cardiac conditions has become increasingly popular in recent mobile technology developments.

CHAPTER 2

LITERATURE SURVEY

2.1 Intelligent Systems for Heart Sound Signal Analysis

M. Nabih-Ali, E. S. A. El-Dashan, and A. S. Yahia has introduced intelligent computer-aided diagnosis (CAD) systems can enhance the diagnostic capabilities of physicians and reduce the time required for accurate diagnosis. CAD systems could provide physicians with a suggestion about the diagnostic of heart diseases. The objective of this paper is to review the recent published pre-processing, feature extraction and classification techniques and their state of the art of phonocardiogram (PCG) signal analysis. Published literature reviewed in this paper shows the potential of machine learning techniques as a design tool in PCG CAD systems and reveals that the CAD systems for PCG signal analysis are still an open problem. Related studies are compared to their datasets, feature extraction techniques and the classifiers they used. Current achievements and limitations in developing CAD systems for PCG signal analysis using machine learning techniques are presented and discussed. In the light of this review, several future research directions for PCG signal analysis are provided.

2.2 Heart Disease and Stroke Statistics

D. Muzaffarabad is an American Heart Association (AHA), in conjunction with the Centres for Disease Control and Prevention, the National Institutes of Health, and other government agencies, brings together the most up-to-date statistics related to heart disease, stroke, and other cardiovascular and metabolic diseases and presents them in its heart disease and Stroke Statistical Update. The Statistical Update represents a critical resource for the lay public, policy makers, media professionals, clinicians, healthcare administrators, researchers, and others seeking the best available data on these conditions. Together, cardiovascular disease (CVD) and stroke produce immense health and economic burdens in the United States and globally. The Statistical Update brings together in a single document up-to-date information on the core health behaviours and health factors that define cardiovascular health; a range of major clinical disease conditions (including stroke, congenital heart disease, rhythm disorders, subclinical atherosclerosis, coronary heart disease, heart failure, valvular disease, and

peripheral arterial disease); and the associated outcomes (including quality of care, procedures, and economic costs). Since 2009, the annual versions of the Statistical Update have been cited >20 000 times in the literature. In 2014 alone, the various Statistical Updates were cited >5700 times. Each annual version of the Statistical Update undergoes major revisions to include the newest nationally representative data, add additional relevant published scientific findings, remove older information, add new sections or chapters, and increase the number of ways to access and use the assembled information. This year-long process, which begins as soon as the previous Statistical Update is published, is performed by the AHA Statistics Committee faculty volunteers and staff. For example, this year's edition includes a new chapter on cardiac arrest, new data on the monitoring and benefits of cardiovascular health in the population, additional information in many chapters on the global CVD and stroke burden, and further new focus on evidence-based approaches to changing behaviour, implementation strategies, and implications of the AHA's 2020 Impact Goals. Below are a few highlights from this year's Update.

2.3 Telehealth In Cardiovascular And Stroke Care

L. H. Schramm's aim for this policy statement is to provide a comprehensive review of the scientific evidence evaluating the use of telemedicine in cardiovascular and stroke care and to provide consensus policy suggestions. We evaluate the effectiveness of telehealth in advancing healthcare quality, identify legal and regulatory barriers that impede telehealth adoption or delivery, propose steps to overcome these barriers, and identify areas for future research to ensure that telehealth continues to enhance the quality of cardiovascular and stroke care. The result of these efforts is designed to promote telehealth models that ensure better patient access to high-quality cardiovascular and stroke care while striving for optimal protection of patient safety and privacy.

2.4 Portable Microfluidic and Smartphone-Based Devices For Monitoring

Cardiovascular diseases (CVDs) are the main causes of morbidity and mortality in the world where about 4 in every 5 CVD deaths happen in low- and middle-income countries (LMICs). Most CVDs are preventable and curable, which is largely dependent on timely and effective interventions, including diagnosis, prognosis and therapeutic monitoring. However, these interventions are high-cost in high income countries and are usually lacking in LMICs. Thanks to the rapid development of microfluidics and nanotechnology, lots of portable analytical devices are developed for detection of CVDs at the point-of-care (POC). In the meantime, smartphone, as a versatile and powerful handheld tool, has been employed not only as a reader for microfluidic assays, but also as an analyser for physiological indexes. In this review, we present a comprehensive introduction of the current status and potential development direction on POC diagnostics for CVDs. First of all, we introduce some main facts about CVDs and their standard diagnostic procedures and methods. Second, we discuss about both commercially available POC devices and developed prototypes for detection of CVDs via immunoassays. Subsequently, we report the advances in smartphone-based readout for microfluidic assays. Finally, we present some examples using smartphone, individually or combined with other components or devices, for CVD monitoring. We envision an integrated smartphone-based system capable of functioning blood tests, disease examination, and imaging will come.

2.5 Digital Monitoring And Care: Virtual Medicine

J. S. Shinbane and L. A. Saxon introduced a remote digital health monitoring technologies can be synergistically organized to create a virtual medical system providing more continuous care centered on the patient rather than the bricks and mortar medical complex. Utilization of the digitalized patient health monitoring can facilitate diagnosis, treatment plans, physician–patient interaction, and accelerate the progress of medical research, education, and training. The field of cardiac electrophysiology has been an early adopter of this shift in care and serves as a paradigm applicable to all areas of medicine. The overall impact of this remote virtual care model on the quality of medical care and patient experience requires greater study, as well as vigilance as to the differences between technology and care in order to preserve the intangible and immeasurable factors that bring humanity to the art and science of medicine.

2.6 Artificial Neural Network Approach For Paediatrics

S. R. Batiker, C. DeGroff, and R. L. Mahajan given the paper presents a novel hybrid model for classifying time series of heart sound signal using time-growing neural network. The proposed hybrid model takes segmental behaviour of heart sound signal into account by combining two different deep learning methods, the Static and the Moving Time-Growing Neural Network, which we call STGNN and MTGNN, respectively. Flexibility of the model in learning both deterministic and stochastic segments of signal allow it to learn those complicated characteristics of heart sound signal caused by any obstruction on semilunar heart valve. The model is trained to distinguish between a patient group and a reference group. The patient group is comprised of the subjects with the semilunar heart valve abnormalities including aortic stenosis, pulmonary stenosis and bicuspid aortic valve, whereas the reference group which is composed of the individuals with the heart abnormalities other than those of the reference group or the healthy ones. The model is validated using two different databases: one comprised of 140 children with various heart conditions, and the other one constituted of 50 elderly patients with aortic stenosis. Both the datasets were collected from the referrals to the University hospitals. The overall accuracy and sensitivity of the model are estimated to be 84.2% and 82.8%, respectively. The results show that the model exhibits sufficient capability to distinguish between the patient and the reference group in such a demanding clinical application.

2.7 Heart Sounds Using An Artificial Neural Network

T. Olmec and Z. Dour, introduced a heart murmur recognition and classification play an important role in the auscultative diagnosis. The method based on hidden Markov model (HMM) was presented to recognize the heart murmur. The murmur was isolated on basis of the principle of wavelet analysis considering the time-frequency characteristics of the heart murmur. This method uses Mel frequency cepstral coefficient (MFCC) to extract representative features and develops hidden Markov model (HMM) for signal classification. The result shows that this method is able to recognize the murmur efficiently and superior to BP neural network (94.2% vs 82.8%). And the findings suggest that the method may have the potential to be used to assist doctors for a more objective diagnosis.

2.8 Coronary Dysfunction Using Neural Networks

R. Folland, E. L. Hines, P. Built, and D. Morgan made the paper applies for artificial neural networks (ANNs) to the analysis of heart sound abnormalities through auscultation. Audio auscultation samples of 16 different coronary abnormalities were collected. Data pre-processing included down-sampling of the auscultated data and use of the fast Fourier transform (FFT) and the Levinson-Durbin autoregression algorithms for feature extraction and efficient data encoding. These data were used in the training of a multi-layer perceptron (MLP) and radial basis function (RBF) neural network to develop a classification mechanism capable of distinguishing between different heart sound abnormalities. The MLP and RBF networks attained classification accuracies of 84% and 88%, respectively. The application of ANNs to the analysis of respiratory auscultation and consequently the development of a combined cardio-respiratory analysis system using auscultated data could lead to faster and more efficient treatment.

2.9 Neural Network Assisted Cardiac Auscultation

Catheters in his daily practice cardiologists often have to make a diagnosis based on measured or estimated data. Sometimes this is relatively simple, for example when a patient presents with: (a) typical chest pain, (b) characteristic ECG abnormalities and finally (c) a typical rise in cardiac enzymes, the diagnosis of acute myocardial infarction can be made with a high positive predictive accuracy, regardless of whether other parameters have been measured or not. Also in the interpretation of the electrocardiogram a diagnosis can sometimes be made easily, with almost 100% certainly. In a wide-QRS tachycardia with evident AV dissociation a ventricular origin is almost certain.

2.10, Non-invasive Detection Of Coronary Artery Disease

Y. M. Akay, M. Akay, W. Wolfowitz, and J. Kostis has found that the partial coronary stenosis generates sounds due to turbulent blood flow. The diastolic heart sounds gives the indications of this stenosis. The electronic stethoscopes can be used to record the heart sounds, which can then be analysed using computers for detecting coronary occlusions. In this study, Heart sound segmentation is done using Duration dependant Hidden Markov Model which extracts diastolic duration from heart sound. Noise signals are reduced using sub segmentation of diastolic signal which is then modelled by AR method. Akaike's Information Criteria showed the AR pole orders 5-15 to describe diastolic period. The diastolic signal is analysed over three

overlapping frequency bands. The MUSIC method is also used to study the frequency spectrum of heart signal. The Principal Component Analysis is used to find the correlation between the features of AR model and peak frequencies obtained using MUSIC method.

2.11 Problem Definition

Nearly 18.6 million people died of cardiovascular disease worldwide in 2019, the most recent year for which figures are available. This represents a 17.1 percent rise over the previous ten years. In 2019, there have been more than 523.2 million cases of cardiovascular disease, a 26.6 percent rise from 2010. The problem of particular interest to deep learning involves classification of audio data, usually described as a lob and a dub sounds, to distinguish between classes of interest. Data can be gathered in real-world situations and frequently which usually contain background noise of every conceivable type caused by the surrounding environment and detector operation. Differences in heart sounds that lead to different heart symptoms can be very small and difficult to distinguish. Successfully classifying this form of audio data, as a challenging problem, requires the use of incredibly sophisticated classifiers.

2.12 Objective Of The Project

With the aim of improving the accuracy of heart sounds classification, an in-depth systematic review and an analysis of existing deep learning methods were performed in the present study, with an emphasis on the convolutional neural network (CNN) methods developed over the last five years. This paper also discusses the challenges and expected future trends in the application of deep learning to heart sounds classification with the objective of providing an essential reference for further study.

CHAPTER 3

EXISTING SYSTEM

3.1 Existing Methodology

The before all existing system works on sets of both Deep learning and data mining. The existing system modules generates comprehensive report by implementing the predictive algorithm. The main aims of the existing system to compare and check the before patient whose having disease outputs and new patient disease and determine future possibilities of the heart disease to a particular patient By Implementing the above mentioned model we will get the goal of developing a system with increased rate of accuracy of estimating the new patient getting heart attack percentage. The model which is proposed for Heart Attack Prediction System is invented for using Deep learning algorithms and approach. But by using all the existing systems the accuracy is very less.

Coronary Heart Disease (CHD), also known as Cardiovascular Disease (CVD) is a result of lack of blood supply to the heart organ. CHD is attributed to many different types of arrhythmia, which are generally defined as irregular, slow, or rapid heartbeats. Acute Myocardial Infarction (MI) can result in death, but fatalities usually depend on the severity of an arrhythmia.

The rising mortality rates due to these heart diseases have demanded early diagnosis of cardiac conditions before they progress to acute MI and leading to death. The most common tool to diagnose different types of arrhythmia is Electrocardiogram (ECG).

Thorough analysis of ECG has gained much attention among researchers to accurately and effectively diagnose arrhythmia and critical cardiac conditions. Traditional diagnosis involves bedside ECG recording and the physician's presence to analyse and diagnose a condition.

However, such methods are time consuming and in most severe cardiac cases, time is crucial in diagnosing such a condition.

3.2 Block Diagram:

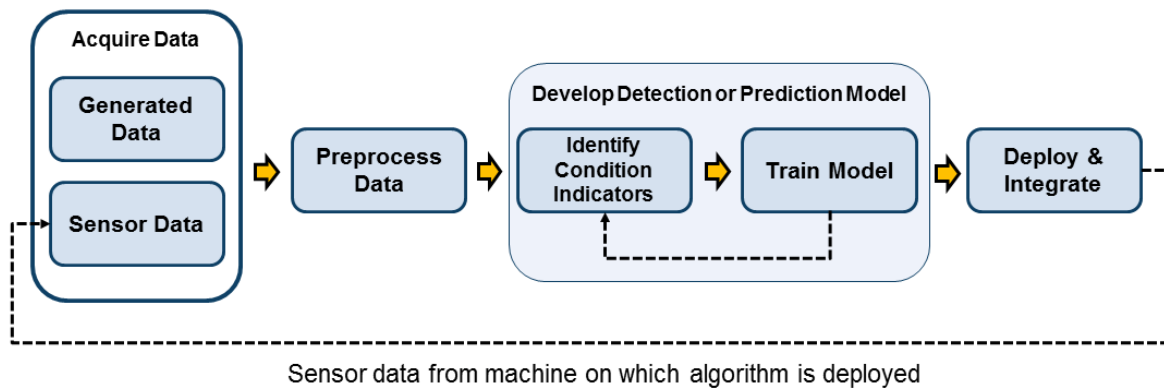


Fig 3.2.1 Predictive Algorithm Block Diagram

3.3 Disadvantages

1. Prediction of cardiovascular disease results is not accurate.
2. Data mining techniques does not help to provide effective decision making.
3. Cannot handle enormous datasets for patient records.

CHAPTER 4

PROPOSED SYSTEM

4.1 Proposed Methodology

1. In proposed system provides an in-depth systematic review and critical analysis of all the existing approaches for automatic identification and classification of the heart sounds.
2. At first the data is collected and pre-processing is performed on the data which is collected.
3. Then the segmentation is performed on the data which is pre-processed.
4. Then it will identify which feature is important and extract then will apply algorithm in order to classifies the heart of sound.
5. The proposed work provides a useful methodology for multi-lead ECG analysis and classification of arrhythmia conditions based on CSVM classification following DFT signal pre-filtering. The study focused on the use of four types of beats obtained from the INCARTDB database although other databases with well-annotated files could have also been used.
6. Single-beat classification was performed. SMO was used to compute the hyper-plane parameters and threshold values for CSVM during the training phase. Results illustrate that the proposed beat classifier is very reliable, and that it may be adopted for automatic detection of arrhythmia conditions and classification. Using CSVM, a 4-class problem can be classified rapidly by decomposing it into two distinct SVM tasks.
7. The CSVM algorithm demonstrated excellent performance, mapping the extracted ECG features to a higher dimensional space through the use of two complex hyper-planes. Using the CSVM algorithm indicated a significant improvement in multi-class classification over MSVM. The overall accuracy using ECG beats extracted from six precordial and limb leads was 98.25%.
8. In terms of using CSVM, the major limitation is that only four types of ECG beats can be classified, this provides motivation to further develop Clifford algebra based ECG analysis classifiers.

4.2 Proposed Algorithm CNN Algorithm

Convolutional Neural Network is one of the main categories to do image classification and image recognition in neural networks. Scene labelling, objects detections, and face recognition, etc., are some of the areas where convolutional neural networks are widely used. CNN takes an image as input, which is classified and process under a certain category such as dog, cat, lion, tiger, etc. The computer sees an image as an array of pixels and depends on the resolution of the image. Based on image resolution, it will see as $h * w * d$, where h = height w = width and d = dimension. For example, An RGB image is $6 * 6 * 3$ array of the matrix, and the grayscale image is $4 * 4 * 1$ array of the matrix. In CNN, each input image will pass through a sequence of convolution layers along with pooling, fully connected layers, filters (Also known as kernels). After that, we will apply the Soft-max function to classify an object with probabilistic values 0 and 1.

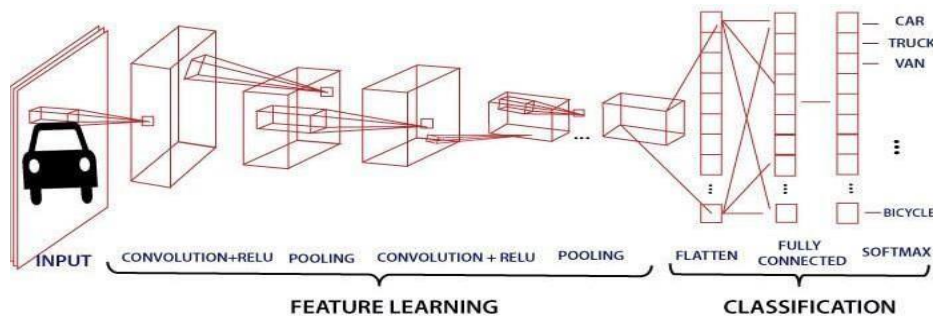


Fig.4.2.1 Convolutional Neural Network

Convolution Layer

Convolution layer is the first layer to extract features from an input image. By learning image features using a small square of input data, the convolutional layer preserves the relationship between pixels. It is a mathematical operation which takes two inputs such as image matrix and a kernel or filter.

1. The dimension of the image matrix is hewed.
2. The dimension of the filter is $f_h \times f_w \times d$.
3. The dimension of the output is $(h-f_h+1) \times (w-f_w+1) \times 1$.

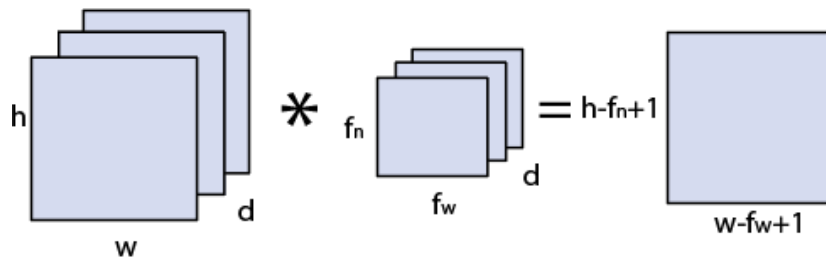


Image matrix multiplies kernel or filter matrix

Fig.4.2.2 Filter matrix

Let's start with consideration a 5*5 image whose pixel values are 0, 1, and filter matrix 3*3 as:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

5 × 5 – Image Matrix 3 × 3 – Filter Matrix

The convolution of 5*5 image matrix multiplies with 3*3 filter matrix is called "Features Map" and show as an output.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 3 & 4 \\ 2 & 4 & 3 \\ 2 & 3 & 4 \end{bmatrix}$$

Convolved Feature

Convolution of an image with different filters can perform an operation such as blur, sharpen, and edge detection by applying filters.

Strides

Stride is the number of pixels which are shift over the input matrix. When the stride is equalled to 1, then we move the filters to 1 pixel at a time and similarly, if the stride is equalled to 2, then we move the filters to 2 pixels at a time. The following figure shows that the convolution would work with a stride of 2.

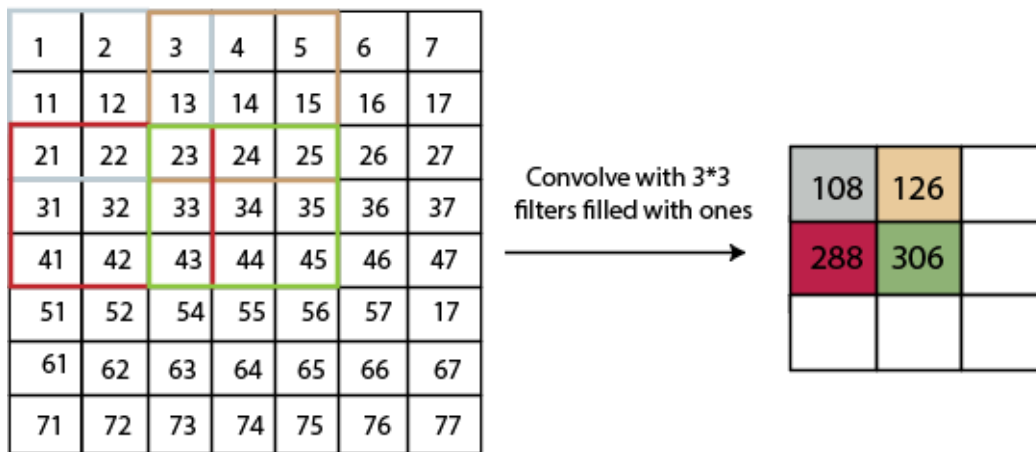


Fig.4.2.3 Strides

Padding

Padding plays a crucial role in building the convolutional neural network. If the image will get shrink and if we will take a neural network with 100's of layers on it, it will give us a small image after filtered in the end.

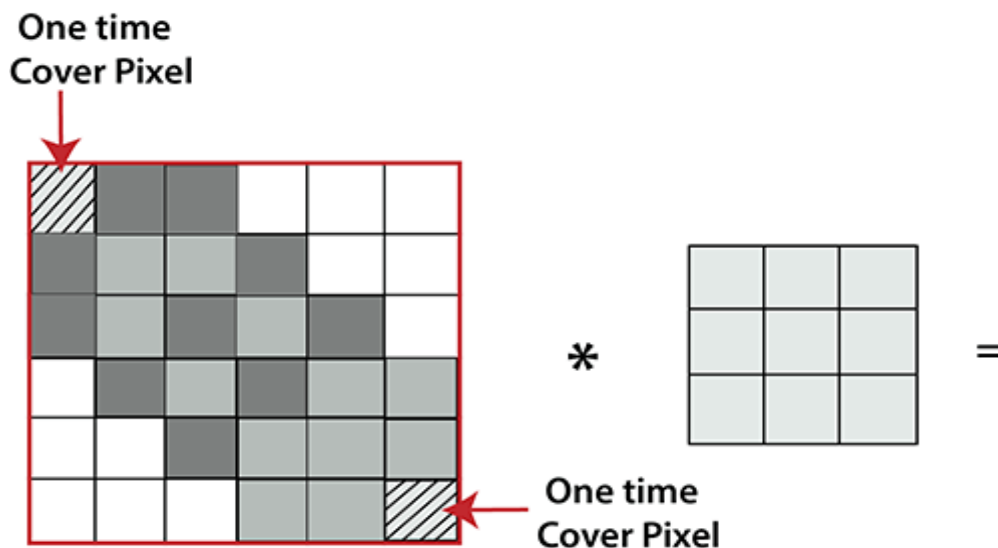


Fig.4.2.4 Padding

It is clear from the above picture that the pixel in the corner will only get covers one time, but the middle pixel will get covered more than once. It means that we have more information on that middle pixel, so there are two downsides:

1. Shrinking outputs.
2. Losing information on the corner of the image.

To overcome this, we have introduced padding to an image. "Padding is an additional layer which can add to the border of an image."

Pooling Layer

Pooling layer plays an important role in pre-processing of an image. Pooling layer reduces the number of parameters when the images are too large. Pooling is "downscaling" of the image obtained from the previous layers. It can be compared to shrinking an image to reduce its pixel density. Spatial pooling is also called down sampling or subsampling, which reduces the dimensionality of each map but retains the important information. There are the following types of spatial pooling.

Max Pooling

Max pooling is a sample-based discretization process. Its main objective is to downscale an input representation, reducing its dimensionality and allowing for the assumption to be made about features contained in the sub-region binned.

Max pooling is done by applying a max filter to non-overlapping sub-regions of the initial representation.

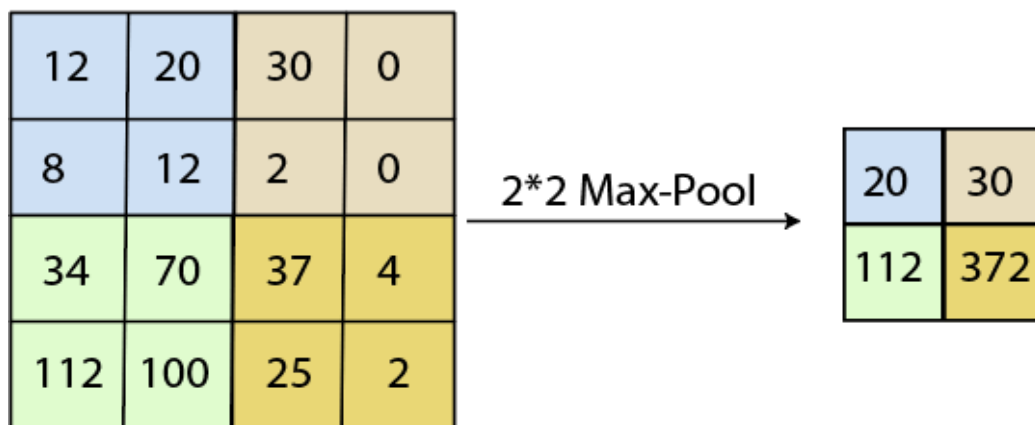


Fig.4.2.5 Max Pooling

Average Polling

Down-scaling will perform through average pooling by dividing the input into rectangular pooling regions and computing the average values of each region.

Syntax

layer = averagePooling2dLayer (pool Size)

layer = averagePooling2dLayer(poolSize,Name,Value)

Sum Pooling

The sub-region for sum pooling or mean pooling are set exactly the same as for max-pooling but instead of using the max function we use sum or mean.

Fully Connected Layer

The fully connected layer is a layer in which the input from the other layers will be flattened into a vector and sent. It will transform the output into the desired number of classes by the network.

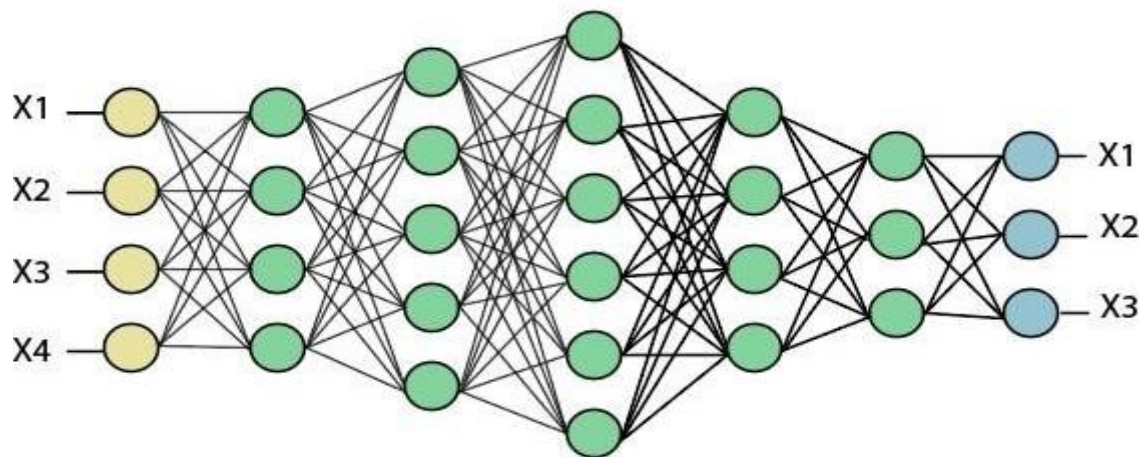


Fig.4.2.6 Fully Connected Layer

In the above diagram, the feature map matrix will be converted into the vector such as $x_1, x_2, x_3, \dots$ in with the help of fully connected layers. We will combine features to create a model and apply the activation function such as soft max or sigmoid to classify the outputs as a car, dog, truck, etc.

Hardware Requirements:

- System - Windows 7/10
- Speed - 2.4GHZ
- Hard disk - 40GB
- Monitor - 15VGA colour
- RAM - 4GB

Software Requirements:

- Operating System - Open CV
- Coding language - PYTHON
- IDE - PYCHARM

4.3 Block Diagram:

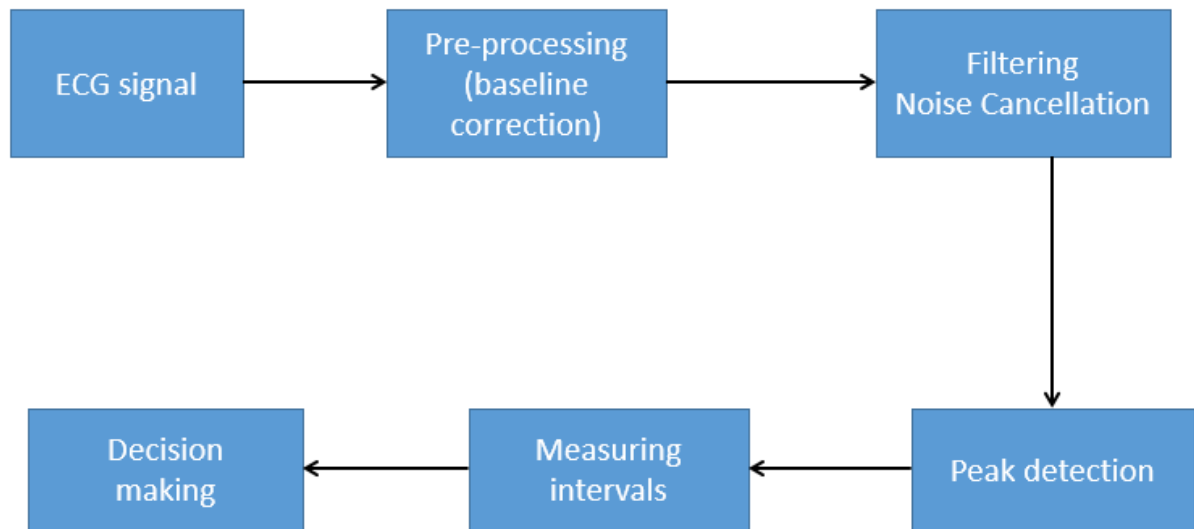


Fig.4.3.1 Block diagram

4.4 Flow

/Chart:

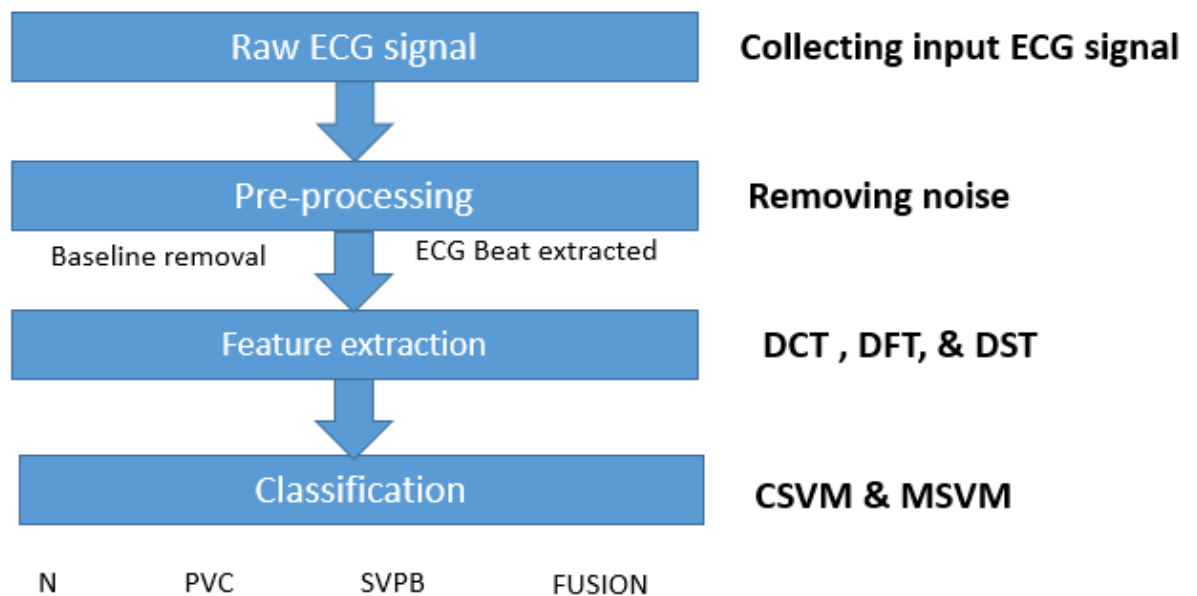


Fig.4.3.2 Flow Chart

4.5 Advantages

1. Reducing Risk.
2. Optimising Marketing Campaigns.
3. Improving Decision Making.
4. Improving Efficiency in Operations.
5. Banking and Financial Services.
6. Retail.
7. Pharmaceuticals.

4.6 Software Environment

Open CV

Open CV (Open-source computer vision) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then It see (which was later acquired by Intel). The library is cross-platform and free for use under the open-source BSD license, Open CV supports the deep learning frameworks Tensor Flow, Torch/ Porch and Caffé.

Applications

OpenCV Application areas include

1. 2D and 3D feature toolkits.
2. Ego motion estimation.
3. Facial recognition system.
4. Gesture recognition.
5. Human–computer interaction (HCI).
6. Mobile robotics.
7. Motion understanding.
8. Object identification.
9. Segmentation and recognition.
10. Stereopsis stereo vision: depth perception from 2 cameras.
11. Structure from motion (SFM).
12. Motion tracking.
13. Augmented reality.

To support some of the above areas, OpenCV includes a statistical machine learning library that contains:

1. Boosting.

2. Decision tree learning.
3. Gradient boosting trees.
4. Expectation-maximization algorithm.
5. k-nearest neighbour algorithm.
6. Naive Bayes classifier.
7. Artificial neural networks.
8. Random forest.
9. Support vector machine (SVM).
10. Deep neural networks (DNN).

Programming Language

OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation. Wrappers in other languages such as C#, Perl, Ch, Haskell, and Ruby have been developed to encourage adoption by a wider audience. Since version 3.4, OpenCV.js is a JavaScript binding for selected subset of OpenCV functions for the web platform. All the new developments and Algorithms in OpenCV are now developed in the C++ interface.

Hardware Acceleration

If the library finds Intel's Integrated Performance Primitives on the system, it will use these proprietary optimized routines to accelerate itself. A CUDA-based GPU interface has been in progress since September 2010. An OpenCL-based GPU interface has been in progress since October 2012, documentation for version 2.4.13.3 can be found at docs.opencv.org.

Oss Support

OpenCV runs on the following desktop operating systems: Windows, Linux, macOS, FreeBSD, NetBSD, OpenBSD. OpenCV runs on the following mobile operating systems: Android, iOS, Maemi, BlackBerry 10. The user can get official releases from Source Forge or take the latest sources from GitHub. OpenCV uses Cake.

4.7 MODULES

4.7.1 Data Acquisition

The term "image retrieval" refers to the process of obtaining an image for processing from a source, typically a hardware-based source. Since there is no way to process without an image, this is the first step in the workflow. The image that is obtained has not been altered in any way.

In image processing, image acquisition is an action of retrieving image from an external source for further processing. It's always the foundation step in the workflow since no process is available before obtaining an image. Learn more in: Development of Class Attendance System Using Face Recognition for Faculty of Mechanical and Manufacturing Engineering, University Tun Hussein Onn Malaysia. The process of capturing an unprocessed image from an object or scene by an optical device into a manageable form for processing and analysis purposes. Learn more in: An Intelligent Machine-Driven Perspective to Archaeological Pottery Reassembly.

4.7.2 Data Pre-processing

Pre-processing is done to improve the image's quality so that we can better analyse it. We can eliminate unwanted distortions and enhance some features that are essential to the application we are working on by pre-processing.

Data preparation and filtering steps can take considerable amount of processing time. Examples of data pre-processing include cleaning, instance selection, normalization, one hot encoding, transformation, feature extraction and selection, etc. The product of data pre-processing is the final training set.

Process for data pre-processing:

1. Acquire the dataset.
2. Import all the crucial libraries.
3. Import the dataset.
4. Identifying and handling the missing values.
5. Encoding the categorical data.
6. Splitting the dataset.
7. Feature scaling.

4.7.3 Segmentation

The process of dividing a digital image into subgroups known as image segments is known as image segmentation. This reduces the complexity of the image and makes it possible to perform additional processing or analysis on each image segment. Segmentation in deep learning.

Image segmentation is the task of clustering parts of an image together that belong to the same object class. This process is also called pixel-level classification. In other words, it involves partitioning images (or video frames) into multiple segments or objects.

4.7.4 Feature Selection and Extraction

Using only relevant data and eliminating data noise, feature selection reduces your model's input variable. It is the process of selecting relevant features for your machine learning model automatically based on the kind of problem you are trying to solve.

Feature Selection and Feature Extraction

The main difference: -

Feature Extraction transforms an arbitrary data, such as text or images, into numerical features that is understood by machine learning algorithms. Feature Selection on the other hand is a machine learning technique applied on these (numerical) features.

Types Of Feature Extraction

1. De noising Auto encoder.
2. Variation Auto encoder.
3. Convolutional Auto encoder.
4. Sparse Auto encoder.

CHAPTER-5

EXPERIMENTAL OR MATERIALS AND METHODS

5.1 CNN Algorithm:

Algorithm Used:

1. CNN (convolutional neural network) is a form of deep learning neural network. In a nutshell, consider CNN to be a machine learning algorithm that can take an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and distinguish one from the other.
2. CNN operates by extracting features from videos
3. Gray scale method uses in image pre-processing for frames conversion. After frame conversion each frame compared with train dataset and detects the suspicious activity.

Image Processing:

1. The Convolutional Neural Network (CNN) is a subtype of Neural Networks that is mainly used for applications in image and speech recognition.
2. Its built-in convolutional layer reduces the high dimensionality of images without losing its information. That is why CNNs are especially suited for this use case.
3. It is used to have applications in image and video recognition, recommender systems, image classification, image segmentation, medical image analysis, natural language processing, brain-computer interfaces, and financial time series.

A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. There are other types of neural networks in deep learning, but for identifying and recognizing objects, CNNs are the network architecture of choice.

CNN Flow Chart:

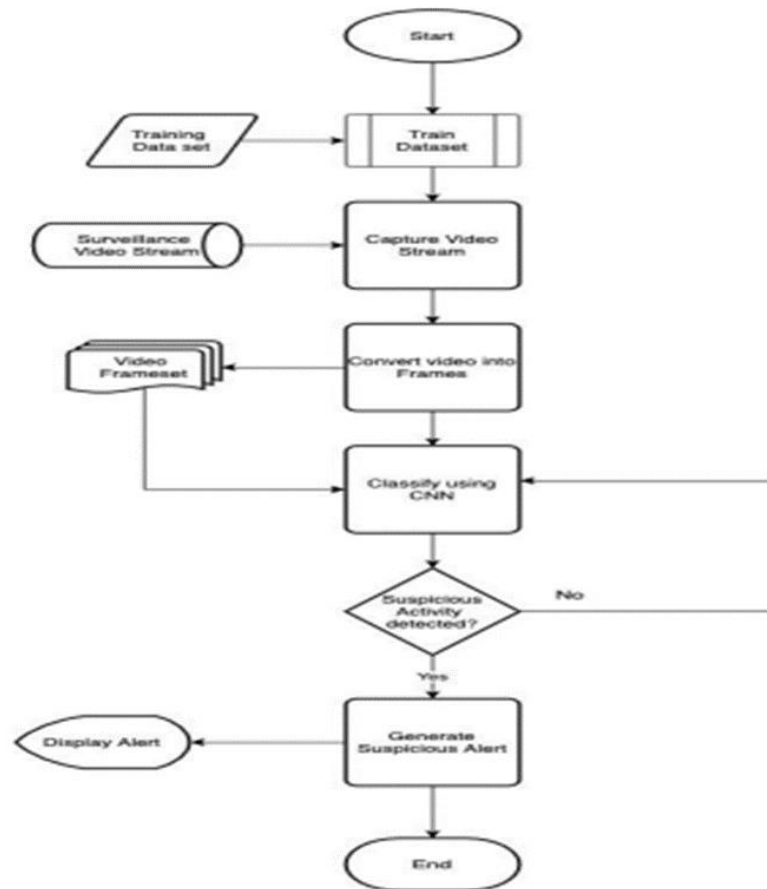


Fig 5.1.1 CNN Flow Chart

5.2 SVM Algorithm:

The state of the heart and the operation of the cardiovascular system can be gauged from the information contained in heart sounds. As a result, this reflects the heart's pathological changes. The capacity to decipher pathological information is a crucial factor in determining the value of heart sound signals. The auxiliary basis for the diagnosis and treatment of heart diseases can be provided by extracting the corresponding features from heart sound signals. As a result, heart sound signals can be processed in a variety of ways. The acquisition of the heart sound signal, its removal of noise, its sampling, its feature extraction, and its classification are the processing steps of the heart sound signal. The heart sound signal sampling, noise removal, and acquisition phases of the proposed algorithm have all been completed first. Highlight extraction and sign arrangement are the centres of the functioning in

this paper. As per the proposed calculation, the steps to implement this algorithm:
(a) Proposed algorithm using SVM;

5.2.1 Wavelet Packet Theory:

The high frequency, or detail, part of the signal is not further decomposed by the wavelet transform because it only further decomposes the low frequency portion. The wavelet packet transform can be used to more precisely break down the high frequency portion, in contrast to the wavelet transform. This decay is neither repetitive nor discarded, so it can perform better time-recurrence restriction examination of the sign. In order to decipher some aspects of the heart sound signal, this paper employs wavelet packet decomposition.

5.2.2 Definition of Wavelet Packet:

Essentially, the orthogonal wavelet packet is a family of functions. They can be used to create $L_2(\mathbb{R})$'s standard orthogonal bases. Various groups of $L_2(\mathbb{R})$ standard orthogonal bases can be chosen here. Typically, one of them is the orthogonal wavelet base. The wavelet packet function family includes the wavelet function. Therefore, the wavelet packet is the wavelet function's generalization.

The orthogonal scaling function (ϕ) is represented by the orthogonal low-pass real coefficient filter $h(k)$. The high-pass channel is $\{g(k)\}$. Comparing to the symmetrical wavelet capability $\psi(t)$, where $g(k) = (-1)^{k+1}h(1-k)$. Both the scale equation and the wavelet equation are satisfied by them:

5.2.3. Wavelet Packet Basis Function:

The Daubechies wavelet, or Daubechies wavelet, is utilized in this paper. N is the order in db., and $N = 2 \dots 10$. The db1 is a Haar wavelet when N is 1. Orthogonality and biorthogonality are two properties of the db. wavelet. It has strong support. The db. is not symmetric and does not have an explicit expression unless $N = 1$. A common and widely used orthogonal wavelet is the db. wavelet.

5.2.4. Wavelet Packet Energy Entropy:

Entropy can be used to measure the uncertainty of information contained in one-dimensional signal or two-dimensional image. The energy entropy of the wavelet

packet obtained here has the characteristics of both feature representation and feature dimension reduction. Its theory is as follows:

Assuming that the length of the sample signal to be analysed is N , the signal is decomposed by wavelet packet and the decomposition level is j . Then the decomposition coefficients are reconstructed and the sequence of the reconstructed coefficients is expressed.

5.3 System Design

Signal Processing:

Signal processing is an electrical designing subfield that spotlights on examining, changing, and combining signals, like sound, pictures, possible fields, seismic signs, altimetry handling, and logical measurements. Signal handling strategies are utilized to enhance transmissions, computerized capacity effectiveness, revising contorted signals, emotional video quality and to likewise recognize or pinpoint parts of interest in a deliberate signal.

Analog- Signal:

Analog signal processing is for signals that have not been digitized, as in most 20th-century radio, telephone, and television systems. This involves linear electronic circuits as well as nonlinear ones. The former are, for instance, and phase-locked loops.

Continuous-Time Signal Processing:

It is used for signals that change as the continuous domain changes (without taking into account individual interrupted points).

Time domain, frequency domain, and complex frequency domain signal processing techniques are all options. This innovation basically examines the displaying of a straight time-invariant nonstop framework, indispensable of the framework's zero-state reaction, setting up framework capability and the constant time separating of deterministic signs

Discrete-Time Signal Processing:

It is used for sampled signals that are quantized in time but not in magnitude because they are only defined at discrete points in time.

Electronic components like sample and hold circuits, analogy time-division multiplexers, analogy delay lines, and analogy feedback shift registers are the foundation of the technology known as analogy discrete-time signal processing. The advanced processing of gigahertz signals still makes use of this technology, which was a precursor to digital signal processing (see below).

A theoretical field that establishes a mathematical foundation for digital signal processing without taking into account quantization error is also referred to as discrete-time signal processing.

Digital Signal Processing:

The processing of digitized discrete-time sampled signals is known as digital signal processing. General-purpose computers or digital circuits like ASICs, field-programmable gate arrays, and specialized digital signal processors (DSP chips) are used for processing. Ordinary arithmetical activities incorporate fixed-point and drifting point, genuine esteemed and complex-esteemed, augmentation and expansion. Lookup tables and circular buffers are two other common operations that the hardware can support. The fast Fourier transform (FFT), the finite impulse response (FIR) filter, the infinite impulse response (IIR) filter, and adaptive filters like the Wiener and Kalman filters are all examples of algorithms.

Nonlinear:

Nonlinear sign handling includes the examination and handling of signs created from nonlinear frameworks and can be in the time, recurrence, or patio-worldly domains. Nonlinear frameworks can deliver exceptionally complex ways of behaving including bifurcations, turmoil, music, and subharmonics which can't be created or broke down utilizing direct techniques.

Polynomial systems can be conceptually understood as straightforward extensions of linear systems to the non-linear case, making polynomial signal processing a type of non-linear signal processing.

Statically Signal Processing:

Statistical Signal processing is an approach that treats signals as stochastic processes and uses the statistical properties of those signals to carry out tasks related to signal processing. Statistical techniques are frequently utilized in applications related to signal processing. To reduce the noise in an image, one could, for instance, build techniques based on a model of the probability distribution of noise that occurs when photographing an image.

5.4 Module Used in The Heart Beat Sound Classification:

1. Databases For Heart Sounds Analysis.
2. Heart Sounds Segmentation.
3. Feature Extraction.
4. Classification Of Heart Sounds.

5.4.1 Databases for Heart Sounds Analysis:

Non-accessibility of normalized, great quality, completely approved, and archived datasets prevent the turn of events of calculations for heart sounds investigation. Physio Net is currently the most comprehensive database of recordings of heart sounds. The PASCAL database the Open Michigan Heart Sound & Murmur Library (OMHSML), the Cardiac Auscultation of Heart Murmurs database (E General Medical), the heart sounds library by Think labs, the heart sounds Podcast Series by Robert J. Hall Heart Sounds Laboratory, Texas Heart Institute, Texas, the Bioscience normal and abnormal heart sounds database (BHSD), and the Cardiac Auscultator Recording Database (CARD) were additional databases that were Moreover, a book by Bricklayer includes a Disc with a set number of heart sounds further more, mumbles. Essentially, heart sounds signals from an general media show by Travel et al was likewise utilized as a data set in a portion of the explored papers. Researchers have also gathered their own data in addition to the databases that are readily available. The greater part of these accounts were gotten during clinical preliminaries in clinics by auscultation utilizing a computerized stethoscope/receiver.

5.4.2 Heart Sounds Segmentation:

The localization of sound peaks, including the fundamental heart sounds (S1 and S2), is the goal of heart sounds segmentation. The pinnacles of S1 and S2 are expected for deciding the systolic and diastolic stages and to help in the ensuing assessment of heart cycles. This works with distinguishing proof and extraction of acoustic signs of interest in each cardiovascular cycle. Methods for reported segmentation fall broadly into the following categories: envelope-based techniques (47, 57, 58, 68, 79, 84, and 89), electrocardiogram, and/or carotid artery.

5.4.3 Feature Extraction:

Portrayals of the cardiovascular signs in various spaces uncover different physiological and neurotic attributes and permit effective component extraction. To catch simultaneous varieties and underlying parts in both time and recurrence, time-recurrence portrayal of the transient signs has been accounted for as a favoured mean throughout the time-space and recurrence space portrayals. Various transforms for the analysis of heart sounds were used to obtain qualitative and quantitative measurements of the signals; S-transform, Fourier transform, Short-Time Fourier Transform (STFT), Wigner-Ville Distribution (WVD), Choi-Williams Distribution (CWD), wavelet transform, and Short-Time Modified Hilbert Transform (STMHT) are just a few examples of time-frequency representations.

Even though the Short-Time Fourier transform (STFT) was found to be popular, it is difficult to get a good resolution for feature extraction using STFT because there is a fixed window for the analysis. By substituting a time or frequency scaling operation for the frequency shifting operation of the STFT, wavelet analysis emerged as an alternative. Due to its suitability for representing signals where the length of the temporal window can be engineered for multi-resolution analysis with a wide frequency range across the length, the wavelet transform received a lot of attention in the literature.

Discrete Wavelet Change (DWT), Persistent Wavelet Change (CWT), and Mel-Scaled Wavelet Change have all been utilized for heart sounds examination. Pattern

recognition and classification of heartbeat signals rely heavily on feature selection and extraction. In order to reduce the system's complexity and computational burden, reviewed articles extracted and optimized features based on the diagnosed cardiovascular disease. In order to properly classify non-stationary PCG signals, features with high-order statistics, non-linear fractal complexity, entropy information, and chaos theory assisted in capturing relevant information. Wavelet coefficients, perceptual features like Mel-Frequency Cepstral Coefficients (MFCCs), spectrum, Variance Fractal Dimension (VFD), and fractal features like the largest Lyapunov Exponent or Hurst Exponent were among the other characteristics.

Time-space highlights are not difficult to extricate and evaluate. Timing characteristics like the locations of S1 and S2, systolic and diastolic intervals, and amplitude information (like the mean absolute value of S1 and S2 and other peaks in the cardiac cycle) are the primary components of these features. The frequency domain analysis can show some physiological and pathological information that is missed by the time-domain analysis. Band-pass filter banks and zero-crossing analysis were the most common frequency-domain based feature extraction techniques. Other discriminant highlights incorporated the mean force of unmistakable heart sounds portions (S1, S2, systole and diastolic) in various recurrence groups and MFCCs. Methods for and types of features extracted are described in detail.

5.4.4 Classification of Heart Sounds:

A heart sound analysis algorithm's final step is to feed the extracted features to an appropriate classifier for interpretation. Revealed approaches for grouping incorporate Help Vector Machine(SVM), Stowed away Markov Models (Gee), k- Closest Neighbours (k-NN), Brain Organizations, rule-based classifier or choice trees, Bayes Net classifier, AI based approaches, Gaussian-Bayes model, Gullible Baye, Gaussian Combination Model (GMM), arbitrary woods, and discriminant investigation.

5.5 Signal Processing Using Python:

An always expanding number of logical examinations are creating bigger, more mind boggling, and multi-modular datasets. As a result, data analysis jobs become

more difficult. To assist with handling these new difficulties, more trains currently need to integrate progressed perception procedures into their standard information handling and investigation strategies. While numerous frameworks have been created to permit researchers to investigate, break down, and envision their information, large numbers of these arrangements are space explicit, restricting their degree as broad handling apparatuses. Building on top of an interpreted language is one way to increase their adaptability.

The Python programming language gives an improvement climate reasonable to both computational and representation errands. One of Python's main advantages is that it can be extended with packages to include advanced features like manipulating arrays and matrices, image processing, digital signal processing , and visualization all well-known data exploration and visualization tools. We make use of Vis-Trails in our work; nonetheless, almost any python-empowered application would be equipped for delivering comparative outcomes.

Example for neuro science

An Example from Neuroscience The field of neuroscience frequently analyses participant data using computationally complex algorithms and multimodal data. Working memory is the subject of a study in which electroencephalography (EEG) and magnetic resonance imaging (MRI) are used together.

The EEG sensors are shown to have a natural spatial organization in the MRI, which depicts the brain's structure in three dimensions. EEG information is gathered from 64 sensors put on the scalp. The voltages that are produced at the scalp by brain activity are measured by these sensors.

The neural assemblies that control the short-term retention of information and the incorporation of this data into the process of making executive decisions are commonly referred to as working memory. Working memory-related brain activity has been shown to be spectrally and spatially organized [10]. Changes in the energy densities, phase relationships, and frequency shifts in the alpha band of frequencies (7–13 Hz) in the dorsal-lateral pre-frontal 1 can specifically be used to measure working memory performance. Top) A plot of the raw data trace from a single sensor. Bottom) The sensor's raw data as a S-transformed representation. This view makes it possible to examine the frequency evolution of the time series because it

displays all representable frequencies at each timestep, cortex. Advanced signal processing methods are utilized to transform the time series gathered from EEG sensors in specific locations into Fourier-based representations in order to analyse the temporal and spectral organization of working memory. It is difficult to see spatial representations of various EEG-based quantities because the sensor locations are known.

The utilization of X-ray information with the known EEG sensor areas features spatial connections between the sensors and the mind movement they measure. By assigning scalar values to the cortical surface based on data collected at sensor locations, MRI data is frequently used to determine participant-specific finite element meshes for use in solving the inverse problem, which is more robust than interpolation schemes. Tackling the backwards issue in this manner is helpful for undertakings, for example, source limitation for epileptogenic districts of the mind. Fortunately, radial basis function interpolation has proven to be a good approximation for determining the spatial relationships between EEG sensors and active brain regions [4]. In order to map scalars from discrete point locations to a surface representation, it is necessary to combine data from various methods of acquisition into a coherent representation. However, the MRI data must first be registered with the EEG sensor locations before this kind of data fusion can be carried out.

In addition, the MRI must be segmented in order to obtain the surface of the brain on which the scalar values derived from the EEG will be mapped.

A visualization of the spectral power at the alpha band of frequencies (7–13Hz) across the entire cortical surface represents the study's conclusion. In addition, each sensor's data must have a time-frequency representation that can be selected to provide details when questioned. The EEG data must be decomposed using signal processing methods, the MRI must be segmented using image processing methods, the EEG data must be approximated using interpolation methods, and the data must be presented using one, two, and three-dimensional rendering methods.

5.6 Erg Signal Processing:

Fourier-based transforms can be utilized to identify the spectral properties of brain activity in order to process EEG data for further analysis and interpretation. The

study of working memory relies heavily on the ability to ascertain how spectral properties alter over time. These methods are necessary because specific spectral properties of EEG signals measured at the scalp can be used to evaluate working memory performance. Alpha-band power increases and shifts to slightly higher frequencies when working memory is tasked. The SciPy Python package includes wavelet implementations, various filters, and fast Fourier transforms (FFTs) for achieving this objective. SciPy's use of the FFTW libraries makes FFT computation quick [2]. However, when looking at how spectral content has changed over time, standard FFTs are insufficient.

There are a number of different transformations that can be used to look at the frequency-based representations of a time series from a temporal perspective. Short-Time Fourier Transforms (STFTs) employ time windows to investigate spectral evolution in one time frequency decomposition. The time frequency space of STFTs is uniformly packed. On the other hand, wavelet-based approaches utilize a versatile goal plan to pack the time-recurrence space, yet miss the mark on direct planning from scale to recurrence. Then again, the Stockwell Change utilizes a versatile goal plot like Wavelet Changes, yet keeps a direct planning to the recurrence space. During the course of an experiment, the results of a Stockwell transform that represented the energy density between 1 and 250 Hz are depicted.

Stockwell Transforms have been selected as the method of choice for our analysis. The processing pipeline needs to be adaptable to a variety of Fourier-based decompositions because this dataset can be analysed with other time-frequency representations. When investigating not only time series data but also the various processed and analysed data products as well, this kind of adaptability is crucial. Utilizing 3. Enlisting sensor areas with X-ray based information uncovers the spatial connections between the EEG sensors and the designs present in the X-ray volume.

Volume segmentation and registration

Once EEG data is processed and analysed, structural information must be extracted from the MRI volume collected. Segmentation of such volumes is a difficult task, best suited for specialty libraries and algorithms. We have employed the Insight Toolkit (ITK) to extract the brain area from the MRI data. ITK is a collection of image processing algorithms implemented in C++ that provide volumetric operations from

smoothing to segmentation. As with many other libraries, ITK is distributed with a collection of Python bindings. In the case of ITK, this is done automatically by Kitware's Cable Swig, which provides a mechanism to wrap highly templated C++ libraries for use with Python. Without Cable Swig, the templated nature of ITK would generate an unnecessarily large number of bindings, complicating its use in Python. We are using ITK here because we need to segment the MRI volume. To simplify this procedure, we decided to use the automatic cortical extraction method of Pasta et al. After segmentation Data processing and visualization performed in Python results in interactive visualizations of multi-modal data. Here we see alpha-band activity (7–13 Hz) in the frontal regions of the brain (left). Specific time-frequency representations are determined by the user-driven selection of EEG sensors (right). This is consistent with known patterns described in literature. an iso-surface of the cortex was extracted to augment the MRI visualization by embedding the surface in a traditional volume rendering. Shows the result of this automatic segmentation. Spatial relationships between different structures are revealed by combining the clipped MRI volume with the extracted cortical surface. Visualizations of this nature make it easier to analyse the data. In particular, correlations between spatial relationships of brain structures and brain activity measured by EEG are highlighted by this type of multimodal visualization. One-way spatial relationships are visualized is using topographic maps (too-maps). This method uses a surrogate representation of the scalp and places values on it based on the EEG data collected. It shows two timesteps in the experiment, but does not take into account the three dimensional nature of the MRI data. By registering the MRI volume, cortical surface, and sensor locations provided by the EEG manufacturer, a cohesive representation of the entire dataset is formed respecting the individual structures measured by the MRI. It shows the dataset after the 3D registration of the MRI volume and sensor locations is performed.

Visualization

We utilize the VTK library for our perception undertakings. Python programs can take advantage of this C++ library's advanced visualization capabilities. We are able to combine point rendering, surface rendering, and volume rendering for the registered sensor locations, cortical surface, and MRI volume, respectively.

To properly map scalar values onto the cortex, however, additional processing is required.

We utilize Spiral Premise Capability (RBF) addition that is accessible in SciPy to assist us with this undertaking. Geographical guides produced from EEG information projected onto a proxy head give a few spatial signs of mind action. The top maps are coloured according to a participant's average alpha power during a working memory task (right) and a resting state (left). The proxy head model addresses just information gathered and introduced onto the scalp trying to plan values to the cortical surface.

Given the values at each sensor location, we can thus approximate scalar values on the cortical surface. Although this method does not solve the inverse problem, it gives a good estimate of how brain activity is organized in space. Figure 4 shows the consequences of applying RBF addition for the normal power in the alpha band of frequencies (7-13 Hz) estimated at every sensor area. VTK likewise gives usefulness to choose and on-the-fly updates to upgrade communication. It is through this system that we can choose individual sensors and show their special time-recurrence portrayals. Utilize SciPy's Pilaf plotting functionality rather than VTK to display the 2D time-frequency planes. Pilaf is an excellent rendering library for time-frequency data because it focuses on methods for plotting in one and two dimensions. The right side of Figure 4 depicts the time-frequency planes of the selected sensors using Pilaf, while the left side displays 3D data rendered with VTK.

CHAPTER-6

RESULT, DISCUSSION AND PERFORMANCE ANALYSIS

A heart sound analysis algorithm's final step is to feed the extracted features to an appropriate classifier for interpretation. Revealed approaches for grouping incorporate Help Vector Machine(SVM).

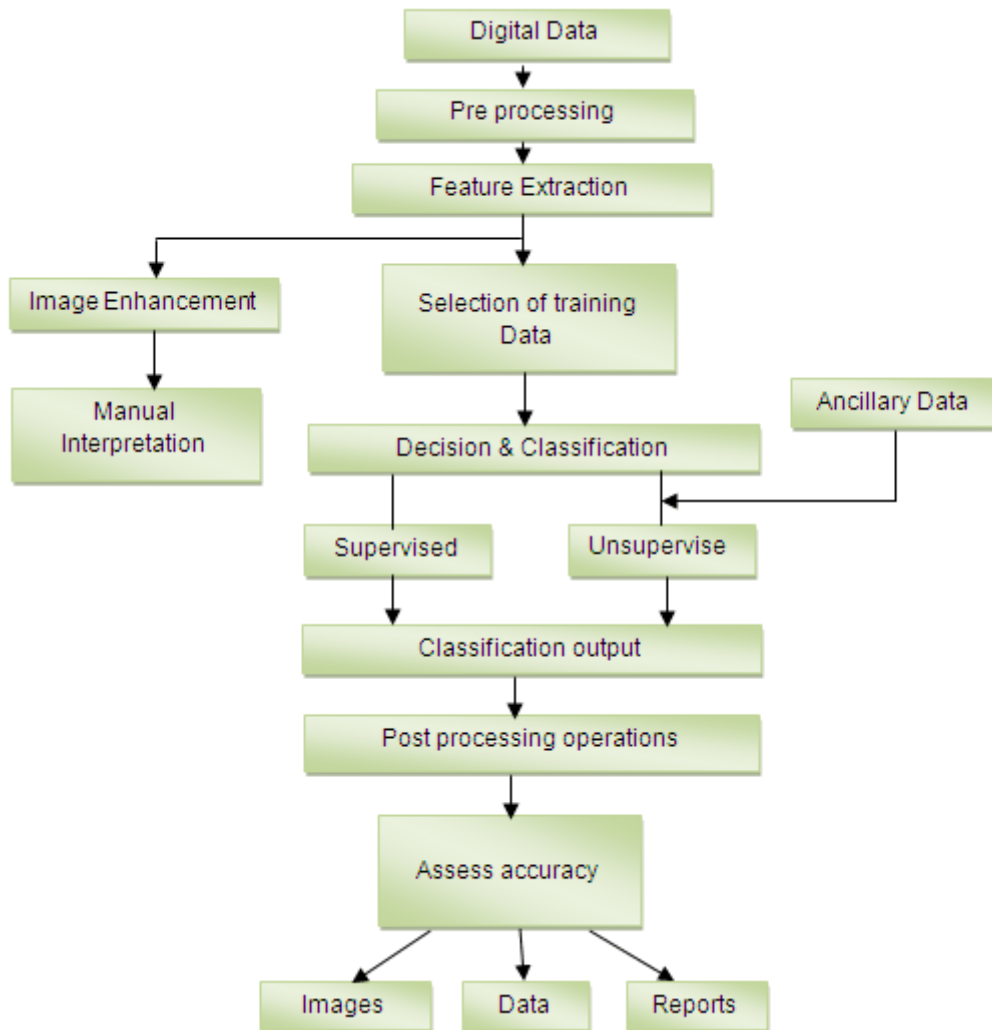


Fig 6.1 Heart sound classification Flow Chart

In terms of using CSVM, the major limitation is that only four types of ECG beats can be classified, this provides motivation to further develop Clifford algebra-based ECG analysis classifiers.

CHAPTER-7

SUMMARY AND CONCLUSION

The key objective of this systematic review was the identification of methodological approaches for computerized heart sounds analysis and classification. This included the review of databases used for testing of the different algorithms, methods for segmentation, feature extraction and classification of heart sounds. A cost-effective system with precise automatic analysis of heart sounds may assist in early diagnosis and to improve the outcomes of cardiovascular diseases. However, extraction and analysis of these signals is a challenging task because of their complex non-stationary nature as well as the noise and interference corruption due to the limitations associated with the acquisition systems. Algorithms for automated analysis of the acoustic cardiac signals have been reported but with limited capabilities. There is a large variation in data in terms of accuracy of some of the studied algorithms. Evaluation with universally standardized databases still needs to be carried out for a proper comparison, and if the algorithms are intended to be used with wearable systems the design and validation needs to take into account the practical challenges associated to the specific wearable. The identification of methodological approaches for the analysis and classification of computerized heart sounds was the primary goal of this systematic review. This included looking over the databases that were used to test the various algorithms, segmentation techniques, feature extraction, and classification of heart sounds. A low-cost system that can precisely analyse heart sounds automatically could help diagnose cardiovascular diseases earlier and improve outcomes. The complex non-stationary nature of these signals, as well as the noise and interference corruption caused by the limitations of the acquisition systems, make extracting and analysing them difficult. Although they have limited capabilities, algorithms for automated analysis of acoustic cardiac signals have been reported. The accuracy of some of the studied algorithms varies greatly depending on the data. If the algorithms are intended to be used with wearable systems, the design and validation must take into account the specific wearable's practical challenges. Evaluation with universally standardized databases is still required for a proper comparison.

CHAPTER - 8

REFERENCES

1. L. H. Schramm et al., "Recommendations for the Implementation of Telehealth in Cardiovascular and Stroke Care A Policy Statement From the American Heart Association," *Am. Hear. Assoc. Policy Statement*, pp. 24–45, 2017.
2. J. Hu et al., "Portable microfluidic and smartphone-based devices for monitoring of cardiovascular diseases at the point of care," *Biotechnology. Adv.*, vol. 34, no. 3, pp. 305–320, 2016.
3. J. S. Shinbane and L. A. Saxon, "Digital monitoring and care: Virtual medicine," *Trends Cardiovasc. Med.*, vol. 26, no. 8, pp. 722–730, 2016.
4. Z. Jiang and S. Choi, "A cardiac sound characteristic waveform method for in-home heart disorder monitoring with electric stethoscope," *Expert Syst. Appl.*, vol. 31, no. 2, pp. 286–298, 2006.
5. S. Mangione and L. Z. Nieman, "Cardiac auscultatory skills of internal medicine and family practice trainees: A comparison of diagnostic proficiency," *JAMA*, vol. 278, no. 9, pp. 717–722, 1997.
6. L. G. Durand and P. Picaro, "Digital Signal Processing of the Phonocardiogram: Reviews of the Most Recent Advancements," *Crit. Rev. Biomed. Eng.*, vol. 23, no. 3/4, pp. 163–219, 1995.
7. A.K. Bhoi, K. S. Sherpa, and B. Khandelwal, "Multidimensional Analytical Study of Heart Sounds : A Review," *Int. J. Bio automation*, vol. 19, no. 3, pp. 351–376, 2015.
8. G. D. Clifford et al., "Recent advances in heart sound analysis," *Physiol. Meas.*, vol. 38, no. 8, pp. 10–25, 2017.
9. R. M. Ranga yan and R. J. Lehner, "Phonocardiogram Signal Analysis: A Review," *CRC Crit. Rev. Biomed. Eng.*, vol. 15, no. 3, pp. 211–236, 1998.
10. M. Nabih-Ali, E. S. A. El-Dashan, and A. S. Yahia, "A review of intelligent systems for heart sound signal analysis," *J. Med. Eng. Technol.*, vol. 41, no. 7, pp. 553–563, 2017.
11. D. Moher, A. Liberati, J. Tetzlaff, D. G. Altman, and P. Group, "Preferred reporting items for systematic reviews and metanalyses: the PRISMA statement," *Plops Med.*, vol. 6, no. 7, p. e1000097, 2009.

CHAPTER - 9

APPENDIX

9.1 Source Code:

```
from codecs import BOM32_BE
from types import alignment
from unit test import result
from xml.dom. parboiler import parse String
import NumPy as np
import pandas as pd
import pydicom as discom
import of
import matplotlib. pilot as plot
import cv2
import math
import TensorFlow. _ape. v2.compat.v1 as ft
ft. disable_v2_behavior()
import pandas as pd
import tolerant

from tflearn.layers.conv import conv_3d, max_pool_3d
from tflearn.layers.core import input_data, dropout, fully_connected
from tflearn.layers.estimator import regression
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
from tkinter import *
from tkinter import messagebox,tkk
import tkinter as tk
from PIL import Image,ImageTk class LCD_CNN:

def _init_(self,root):
```

```

self.root=root
#window size
self.root.geometry("1006x500+0+0")
self.root.resizable(False, False)
self.root.title("Heart beat classification")
img4=Image.open(r"Images\image.jpg")
img4=img4.resize((1006,500),Image.ANTIALIAS)
self.photoimg4=ImageTk.PhotoImage(img4)
bg_img=Label(self.root,image=self.photoimg4)
bg_img.place(x=0,y=50,width=1006,height=500)
title_lbl=Label(text="Heart beat classification",font=("Bradley Hand
ITC",30,"bold"),bg="black",fg="white",)
title_lbl.place(x=0,y=0,width=1006,height=50)
self.b1=Button(text="Import

Data",cursor="hand2",command=self.import_data,font=("Times New
Roman",15,"bold"),bg="white",fg="black")
self.b1.place(x=80,y=130,width=180,height=30)
self.b2=Button(text="Pre-Process

Data",cursor="hand2",command=self.preprocess_data,font=("Times New
Roman",15,"bold"),bg="white",fg="black")
self.b2.place(x=80,y=180,width=180,height=30)
self.b2["state"] = "disabled"
self.b2.config(cursor="arrow")
self.b3=Button(text="Train

Data",cursor="hand2",command=self.train_data,font=("Times New
Roman",15,"bold"),bg="white",fg="black")
self.b3.place(x=80,y=230,width=180,height=30)
self.b3["state"] = "disabled"
self.b3.config(cursor="arrow")

def import_data(self):

```

```

self.dataDirectory = 'sample_audio/'
self.lungPatients = os.listdir(self.dataDirectory)
self.labels = pd.read_csv('stage1_labels.csv', index_col=0)
self.size = 10
self.NoSlices = 5
messagebox.showinfo("Import Data" , "Data Imported Successfully!")
self.b1["state"] = "disabled"
self.b1.config(cursor="arrow")
self.b2["state"] = "normal"
self.b2.config(cursor="hand2")

def preprocess_data(self):
    def chunks(l, n):
        count = 0
        for i in range(0, len(l), n):
            if (count < self.NoSlices):
                yield l[i:i + n]
                count = count + 1

    def mean(l):
        return sum(l) / len(l)

def dataProcessing(patient, labels_df, size=10, noslices=5,
visualize=False):
    label = labels_df._get_value(patient, 'cancer')
    path = self.dataDirectory + patient
    slices = [dicom.read_file(path + '/' + s) for s in os.listdir(path)]
    slices.sort(key=lambda x: int(x.ImagePositionPatient[2]))
    new_slices = []
    slices = [cv2.resize(np.array(each_slice.pixel_array), (size, size)) for
each_slice in slices]
    chunk_sizes = math.floor(len(slices) / noslices)
    for slice_chunk in chunks(slices, chunk_sizes):
        slice_chunk = list(map(mean, zip(*slice_chunk)))

```

```

        new_slices.append(slice_chunk)
    if label == 1: #Cancer Patient
        label = np.array([0, 1])
    elif label == 0: #Non Cancerous Patient
        label = np.array([1, 0])
    return np.array(new_slices), label
imageData = []

#Check if Data Labels is available in CSV or not for num, patient in
enumerate(self.lungPatients):
    if num % 50 == 0:
        print('Saved -', num)
    try:
        img_data, label = dataProcessing(patient, self.labels,
size=self.size, noslices=self.NoSlices)
        imageData.append([img_data, label, patient])
    except KeyError as e:
        print('Data is unlabeled')
    np.save('imageDataNew-{}-{}-{}.npy'.format(self.size, self.size,
self.NoSlices), imageData)
    messagebox.showinfo("Pre-Process Data" , "Data Pre-Processing
Done Successfully!")

self.b2["state"] = "disabled"
self.b2.config(cursor="arrow")
self.b3["state"] = "normal"
self.b3.config(cursor="hand2")
def train_data(self):

imageData = np.load('imageDataNew-10-10-5.npy', allow_pickle=True)
trainingData = imageData[0:45]
validationData = imageData[45:50]

```

```

training_data=Label(text="Total Training Data: " +
str(len(trainingData)),font=("Times New Roman",13,"bold"),bg="black",
fg="white",)

```

```

training_data.place(x=750,y=150,width=200,height=18)

```

```

validation_data=Label(text="Total Validation Data: " +
str(len(validationData)),font=("Times New
Roman",13,"bold"),bg="black",fg="white",)

```

```

validation_data.place(x=750,y=190,width=200,height=18)

```

```

x = tf.placeholder('float')

```

```

y = tf.placeholder('float')

```

```

size = 10

```

```

keep_rate = 0.8

```

```

NoSlices = 5

```

```

def convolution3d(x, W):

```

```

    return tf.nn.conv3d(x, W, strides=[1, 1, 1, 1, 1], padding='SAME')

```

```

def maxpooling3d(x):

```

```

    return tf.nn.max_pool3d(x, ksize=[1, 2, 2, 2, 1], strides=[1, 2, 2, 2,
1], padding='SAME')

```

```

def cnn(x):

```

```

    x = tf.reshape(x, shape=[-1, size, size, NoSlices, 1])

```

```

    convolution1 = tf.nn.relu(

```

```

        convolution3d(x, tf.Variable(tf.random_normal([3, 3, 3, 1, 32]))) +
tf.Variable(tf.random_normal([32])))

```

```

    convolution1 = maxpooling3d(convolution1)

```

```

    convolution2 = tf.nn.relu(

```

```

        convolution3d(convolution1, tf.Variable(tf.random_normal([3, 3, 3,
32, 64]))) + tf.Variable(

```

```

        tf.random_normal([64])))

```

```

    convolution2 = maxpooling3d(convolution2)

```

```

    convolution3 = tf.nn.relu(

```

```

        convolution3d(convolution2, tf.Variable(tf.random_normal([3, 3, 3,
64, 128]))) + tf.Variable(

```

```

        tf.random_normal([128])))

```

```

convolution3 = maxpooling3d(convolution3)
convolution4 = tf.nn.relu(
    convolution3d(convolution3, tf.Variable(tf.random_normal([3, 3, 3,
128, 256]))) + tf.Variable(tf.random_normal([256])))
convolution4 = maxpooling3d(convolution4)
convolution5 = tf.nn.relu(
    convolution3d(convolution4, tf.Variable(tf.random_normal([3, 3, 3,
256, 512]))) + tf.Variable(tf.random_normal([512])))
convolution5 = maxpooling3d(convolution4)
fullyconnected = tf.reshape(convolution5, [-1, 256])
fullyconnected = tf.nn.relu(
    tf.matmul(fullyconnected, tf.Variable(tf.random_normal([256,
256]))) + tf.Variable(tf.random_normal([256])))
fullyconnected = tf.nn.dropout(fullyconnected, keep_rate)
output = tf.matmul(fullyconnected,
tf.Variable(tf.random_normal([256, 2]))) + tf.Variable(tf.random_normal([2]))
return output
def network(x):
    prediction = cnn(x)
    cost =
tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(logits=prediction,
labels=y))
    optimizer = tf.train.AdamOptimizer(learning_rate=1e-
3).minimize(cost)
    epochs = 100
    with tf.Session() as session:
        session.run(tf.global_variables_initializer())
        for epoch in range(epochs):
            epoch_loss = 0
            for data in trainingData:
                try:
                    X = data[0]
                    Y = data[1]
_, c = session.run([optimizer, cost], feed_dict={x: X, y: Y})

```

```

        epoch_loss += c
    except Exception as e:
        pass
    correct = tf.equal(tf.argmax(prediction, 1), tf.argmax(y, 1))
    # if tf.argmax(prediction, 1) == 0:
    accuracy = tf.reduce_mean(tf.cast(correct, 'float'))
    print('Epoch', epoch + 1, 'completed out of', epochs, 'loss:',
epoch_loss)
    # print('Correct:',correct.eval({x:[i[0] for i in validationData],
y:[i[1] for i in validationData]}))
    print('Accuracy:', accuracy.eval({x: [i[0] for i in validationData],
y: [i[1] for i in validationData]}))
    #print('Final Accuracy:', accuracy.eval({x: [i[0] for i in
validationData], y: [i[1] for i in validationData]}))
    x1 = accuracy.eval({x: [i[0] for i in validationData], y: [i[1] for i in
validationData]})
    final_accuracy=Label(text="Final Accuracy: " +
str(x1),font=("Times New Roman",13,"bold"),bg="black", fg="white",)
    final_accuracy.place(x=750,y=230,width=200,height=18)
    patients = []
    actual = []
    predicted = []
    finalprediction = tf.argmax(prediction, 1)
    actualprediction = tf.argmax(y, 1)
    for i in range(len(validationData)):
        patients.append(validationData[i][2])
        for i in finalprediction.eval({x: [i[0] for i in validationData], y: [i[1]
for i in validationData]}):
            if(i==1):
                predicted.append("Brady cardia")
            else:
                predicted.append("Tachy cardia")
        for i in actualprediction.eval({x: [i[0] for i in validationData], y: [i[1]
for i in validationData]}):

```

```

        if(i==1):
            actual.append("Brady cardia")
        else:
            actual.append("Tachy cardia")

for i in range(len(patients)):
    print("-----")
    print("Heart patient: ",patients[i])
    print("Actual: ", actual[i])
    print("Detected: ", predicted[i])
    print("-----")
    # messagebox.showinfo("Result" , "Patient: " + '
'.join(map(str,patients)) + "\nActual: " + str(actual) + "\nPredicted: " +
str(predicted) + "Accuracy: " + str(x1))

y_actual = pd.Series(
    (actualprediction.eval({x: [i[0] for i in validationData], y: [i[1] for i
in validationData]})),
    name='Actual')
y_predicted = pd.Series(
    (finalprediction.eval({x: [i[0] for i in validationData], y: [i[1] for i
in validationData]})),
    name='Detected')
df_confusion = pd.crosstab(y_actual,
y_predicted).reindex(columns=[0,1],index=[0,1], fill_value=0)
print('Confusion Matrix:\n')
print(df_confusion)
prediction_label=Label(text=">>>>  DETECTION
<<<<",font=("Times New Roman",14,"bold"),bg="#778899", fg="black",)
prediction_label.place(x=0,y=458,width=1006,height=20)
result1 = []
for i in range(len(validationData)):
    result1.append(patients[i])
    if(y_actual[i] == 1):

```

```

        result1.append("brady cardia")
    else:
        result1.append("Tachy cardia")

    if(y_predicted[i] == 1):
        result1.append("Brady Cardia")
    else:
        result1.append("Tachy cardia")
# print(result1)
total_rows = int(len(patients))
total_columns = int(len(result1)/len(patients))
heading = ["Dataset: ", "Actual: ", "Detected: "]

self.root.geometry("1006x"+str(500+(len(patients)*20)-20)+"+0+0")
self.root.resizable(False, False)
for i in range(total_rows):
    for j in range(total_columns):
        self.e = Entry(root, width=42, fg='black', font=('Times New
Roman',12,'bold'))

        self.e.grid(row=i, column=j)
        self.e.place(x=(j*335),y=(478+i*20))
        self.e.insert(END, heading[j] + result1[j + i*3])
        self.e["state"] = "disabled"
        self.e.config(cursor="arrow")
        self.b3["state"] = "disabled"
        self.b3.config(cursor="arrow")
        messagebox.showinfo("Train Data" , "Model Trained
Successfully!")

    ## Function to plot confusion matrix

    def plot_confusion_matrix(df_confusion, title='Confusion matrix',
cmap=plt.cm.gray_r):\
plt.matshow(df_confusion, cmap=cmap) # imshow

```

```

plt.colorbar()
tick_marks = np.arange(len(df_confusion.columns))
plt.title(title)
plt.xticks(tick_marks, df_confusion.columns, rotation=45)
plt.yticks(tick_marks, df_confusion.index)
plt.ylabel(df_confusion.index.name)
plt.xlabel(df_confusion.columns.name)
plt.show()

plot_confusion_matrix(df_confusion)
network(x)

if __name__ == "__main__":
    root=Tk()
    obj=LCD_CNN(root)
    root.mainloop()

```

9.2. CODE OUTPUT:

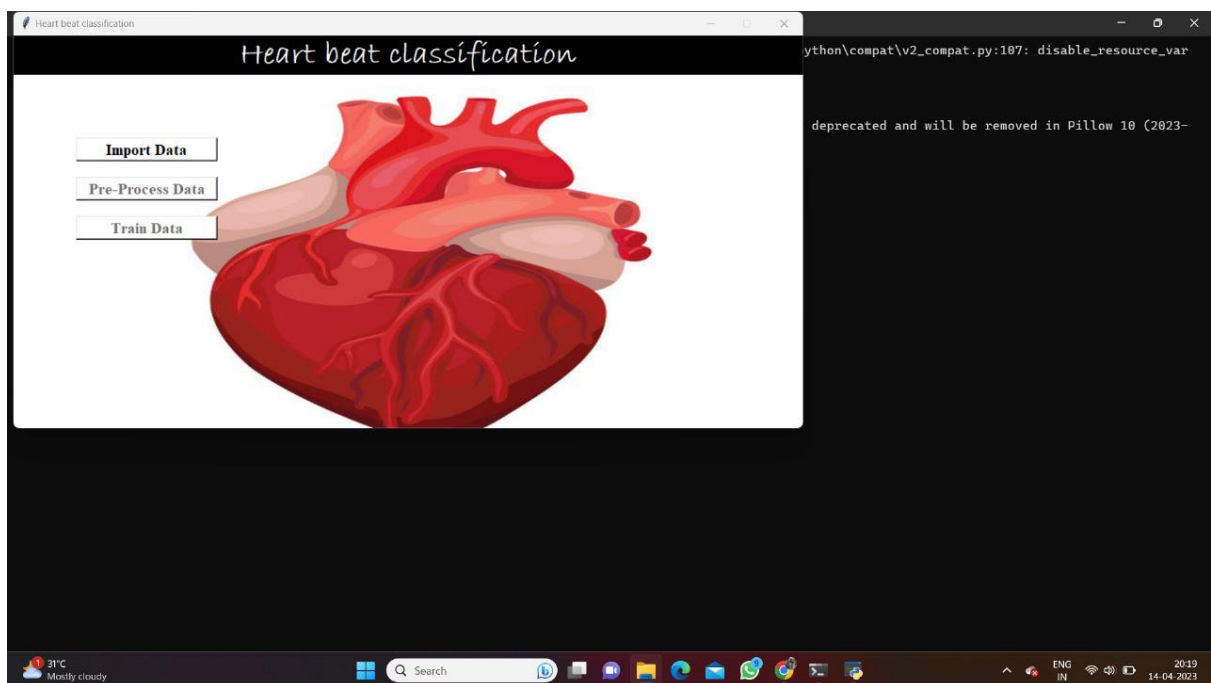


Fig 9.2.1. Import Data

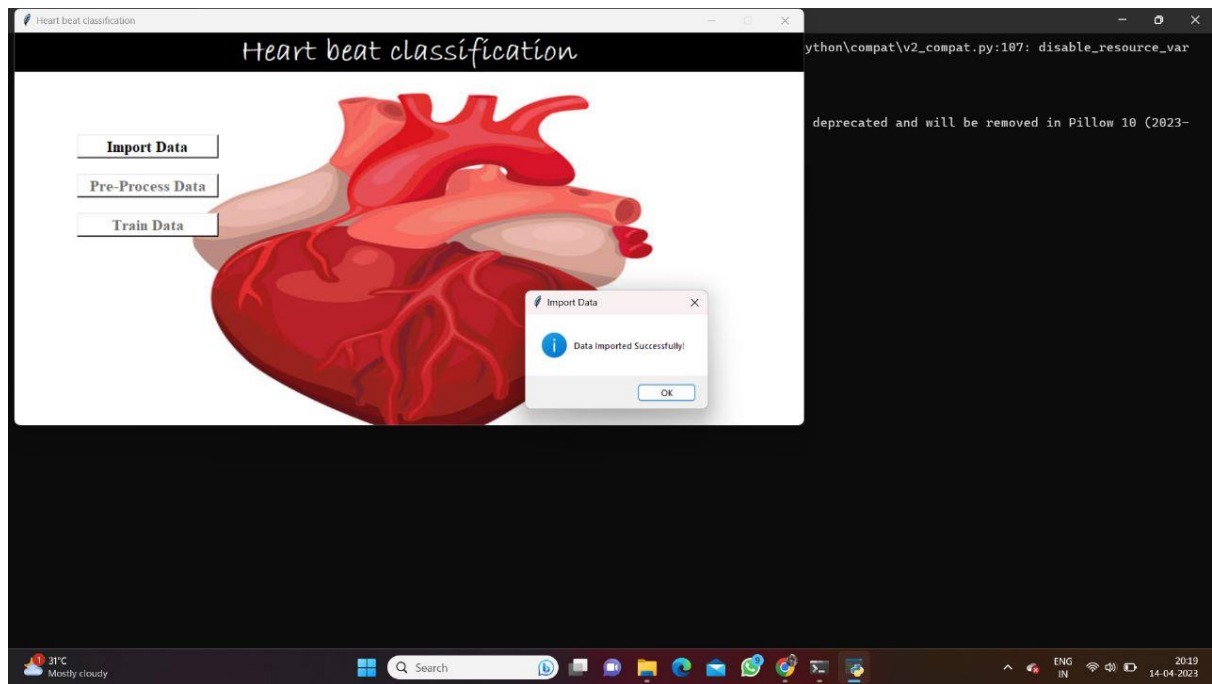


Fig 9.2.2. Data Imported

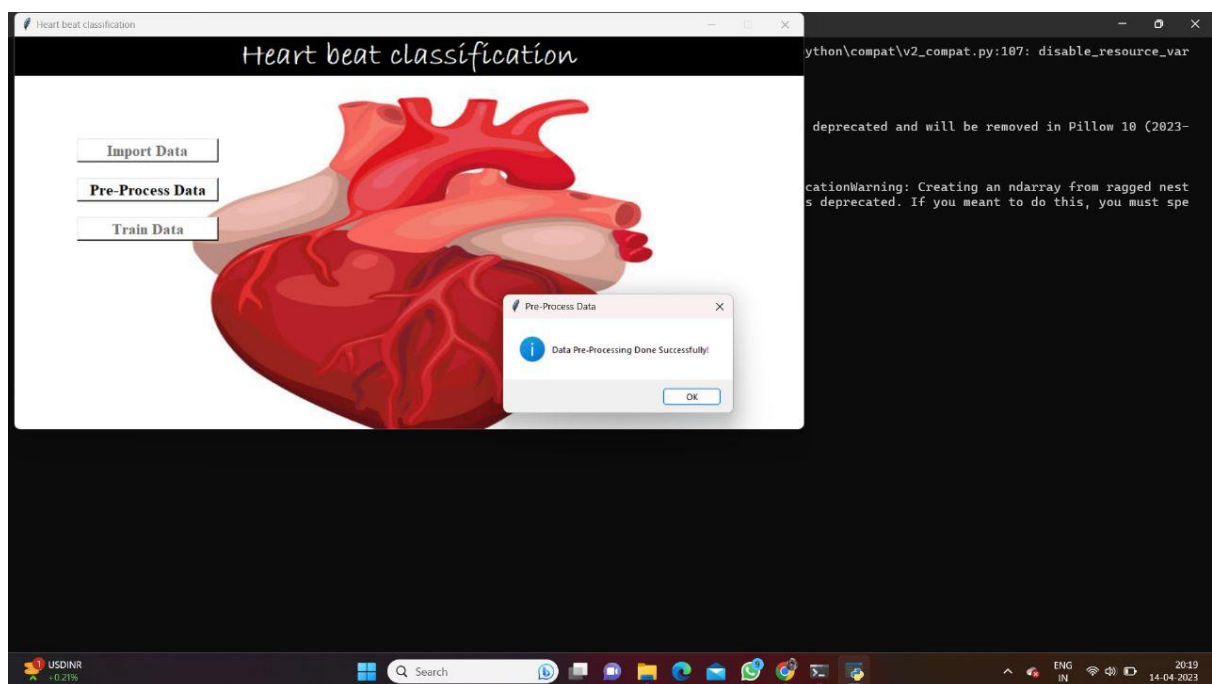


Fig 9.2.3. Pre-Process Data

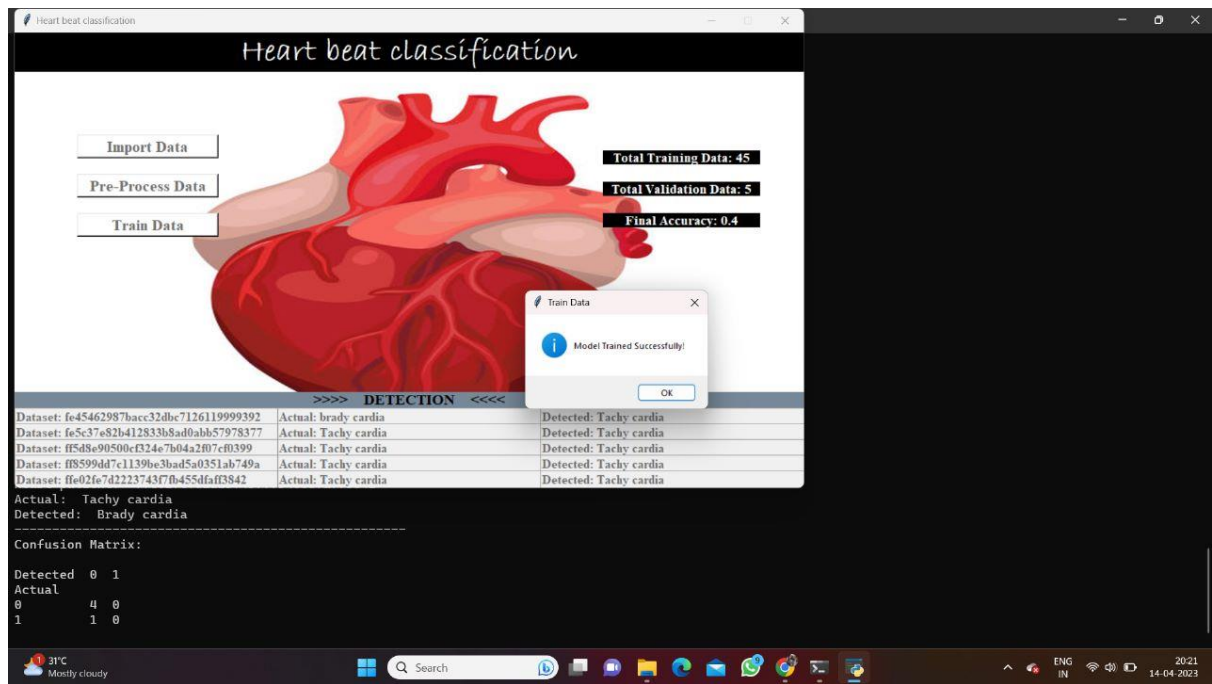


Fig 9.2.4. Train Data

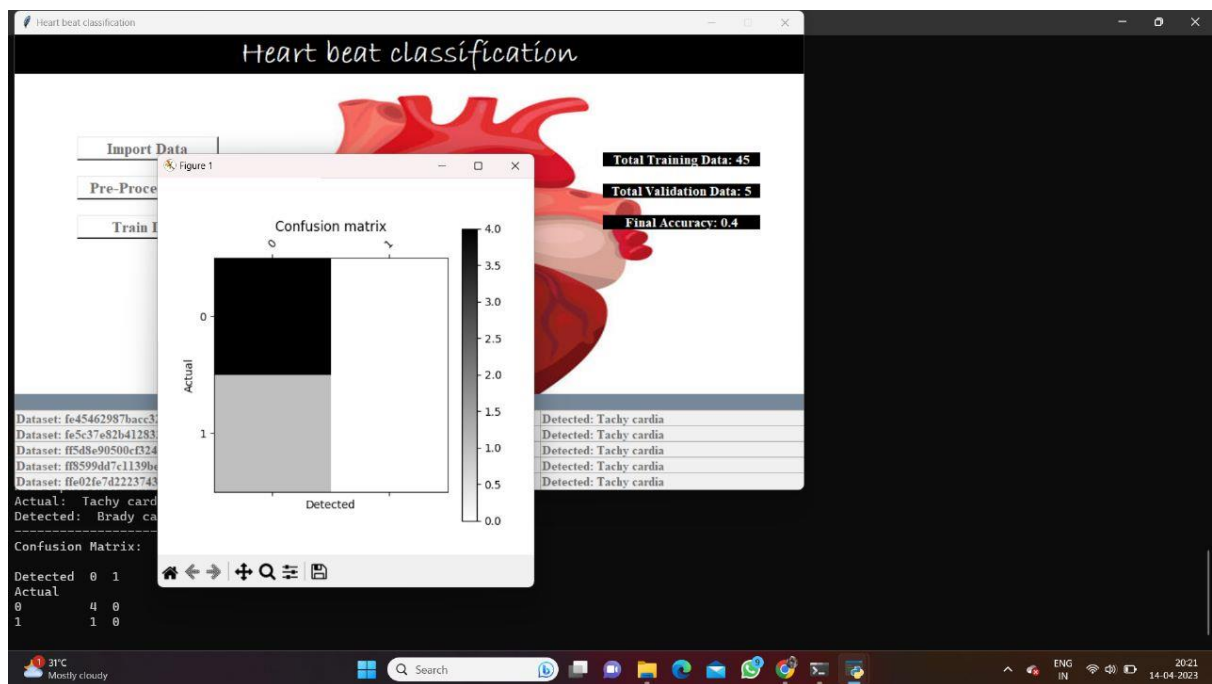


Fig 9.2.5. Heart Beat Graph