

DETECTION OF CARDIAC ARRHYTHMIA USING MACHINE LEARNING TECHNIQUES

Submitted in partial fulfilment of the requirements for the award of
Bachelor of Technology degree in Biomedical Engineering.

By

KARKUZHALI D (37240041)

VIGNESHWARI P (37240076)



**DEPARTMENT OF BIOMEDICAL ENGINEERING
SCHOOL OF BIO AND CHEMICAL ENGINEERING**

SATHYABAMA

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

**Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

MARCH -2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC | 12B Status by UGC | Approved by AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF BIOMEDICAL ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Karkuzhali D (37240041), Vigneshwari P (37240076)** who have done the project work as a team who carried out the project entitled “**Detection Of Cardiac Arrhythmia Using Machine Learning Technique**” under my supervision from **October 2020 to March 2021**

Mr. G. UMASHANKAR, M.Tech, (Ph.D.),

Internal Guide

Dr. T. SUDHAKAR, M.Sc., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

We, **Karkuzhali D (37240041)** and **Vigneshwari P (37240076)** hereby declare that the Project Report entitled “**Detection of Cardiac Arrhythmia Using Machine Learning Technique**” done by us under the guidance of **Mr. G. UmaShankar**, Department of Biomedical Engineering is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Biomedical Engineering.

DATE: 10.04.2021

PLACE:Chennai

1. 

2. 

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to the **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. J. Premkumar** M.Sc., Ph.D., **Dr. T. Sudhakar** M.Sc., Ph.D., **Head of the Department, Dept. of Biomedical Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Mr. G.UmaShankar**, M.Tech.,(Ph.D.), **Department of Bio Medical Engineering** for his valuable guidance, suggestions and constant encouragement paving the 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 Biomedical Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Cardiovascular disease is the number one deadly disease in the world. Arrhythmia is one of the types of cardiovascular disease which is hard to detect but by using the routine electrocardiogram (ECG) recording. Due to the variety and noise of ECG, it takes a long time for experts to identify it with their eyes alone. Learning from the previous research in order to help the experts, this research develop 11 layers Convolutional Neural Network 2D (CNN 2D) using MITBIH Arrhythmia Dataset. The dataset is firstly preprocessed. The challenge is how to conquer the imbalance and small amount of data but still get the optimal accuracy. This study could aid doctors in determining the type of arrhythmia a patient is experiencing. The result is cnn get the highest performance with 91% accuracy and faster 1s training duration than Adam which is approximately 18s per epoch.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO.
	ABSTRACT	i
	LIST OF FIGURES	iv
	LIST OF SYMBOLS AND ABBREVIATIONS	v
1	INTRODUCTION	1
	1.1.CARDIOVASCULAR DISEASES	1
	1.2.CARDIAC ARRHYTHMIA	1
	1.3. ELECTROCARDIOGRAM	1
	1.4 CONVOLUTIONAL NEURAL NETWORK	2
2	LITERATURE REVIEW	3
3	AIM AND SCOPE	8
	3.1. AIM	8
	3.2. SCOPE	8
	3.3. EXISTING SYSTEM	8
	3.4. PROPOSED SYSTEM	8
4	METHODOLOGY	9
	4.1 IMPLEMENTATION	9
	4.2 MODEULE DESCRIPTION	9

4.2.1 DATA COLLECTION	9
4.2.2. DATA PREPROCESSING	9
4.2.3. CLASSIFIER TRAINING	10
4.2.4. CNN ALGORITHM	10
4.3. MACHINE LEARNING	10
4.4. MACHINE LEARNING METHODS	11
4.4.1. SUPERVISED MACHINE LEARNING ALGORITHM S	11
4.4.2. UNSUPERVISED MACHINE LEARNING ALGORITHMS	11
4.4.3. SEMI SUPERVISED MACHINE LEARNING ALGORITHMS	11
4.4.4. REINFROCEMENT MACHINE LEARNING ALGORITHMS	12
4.5 ADVANTAGES OF MACHINE LEARNING	12
4.6. SYSTEM ANALYSIS	12
4.6.1 FEASIBILITY STUDY	12
4.7. REQUIREMENT SPECIFICATION	14
4.7.1. HARDWARE REQUIREMENTS	14
4.7.2. SOFTWARE REQUIREMENTS	14
4.8. TECHNOLOGIES USED	15

	4.8.1. PYTHON LANGUAGE	15
	4.8.2.PYTHON PROGRAMMING CHARACTERISTICS	15
	4.8.3.APPLICATION OF PYTHON PROGRAMMING	15
	4.8.4. OPENCV PACKAGE	16
	4.9. ANACONDA	17
	4.9.1. ANACONDA NAVIGATOR	17
	4.10. ANACONDA DISTRIBUTION	18
	4.10.1.APPLICATION PROVIDED IN ANACONDA DISTRIBUTION	19
	4.11. NEW FEAYTURES OF ANACONDA	21
	5.3	
	4.12. TESTING	22
	4.12.1. TEST CASE	22
	4.12.2. TEST TECHNIQUES	22
5	RESULTS AND DISCUSSION	25
	5.1. RESULTS	25
	5.2. DISCUSSION	29
6	SUMMARY AND CONCLUSION	31
	6.1. SUMMARY	31
	6.2. CONCLUSION	31
	REFERENCES	32
	APPENDIX	36

LIST OF FIGURES

FIGURE NO	NAME OF THE FIGURE	PAGE NO.
4.1	SYSTEM ARCHITECTURE	9
4.2	ANACONDA NAVIGATOR	18
4.3	ANACONDA DISTRIBUTION	19
4.4	ANACONDA 5.3	21
5.1	PHYTHON PROGRAMMING	25
5.2	MIT-BIH DATASET TRAINING	26
5.3	REMOVING NULL SET	26
5.4	FILTERED VALUES	27
5.5	HISTOGRAM ECG SIGNAL	27
5.6	CONFUSION MATRIX WITH NORMALISATION	28
5.7	TRAINING	28
5.8	TAINING ACCURACY AND LOSS	29

LIST OF SYMBOLS AND ABBREVIATIONS

CNN	Convolution neural network
AI	Artificial intelligence
GUI	Graphical user interface
CMS	Content management system
ECG	Electrocardiogram
CVD	Cardiovascular diseases

CHAPTER 1

INTRODUCTION

1.1 CARDIOVASCULAR DISEASES

Cardiovascular disorders (CVDs) are the most common cause of death in humans, killing over 17 million people per year. According to the World Heart Federation, three-quarters of all CVD deaths occur in the middle and lower income classes. A classification model to identify CVDs at an early stage could potentially decrease the mortality rate by including a classification model to identify CVDs at an early stage

1.2 CARDIAC ARRHYTHMIA

Cardiac arrhythmia, in which heartbeats deviate from their normal sequence, is one of the most frequent causes of CVDs. The rate of a regular heartbeat varies depending on age, body size, activity, and emotions. Palpitations are a disorder in which the pulse sounds too rapid or too slow. An arrhythmia is a condition in which the heart beats irregularly. It does not always signify that the heart is pounding too quickly or too slowly. It could indicate that the heart is pounding very fast—tachycardia (more than 100 beats per minute (bpm)) or too slowly—bradycardia (less than 60 bpm)—skipping a beat, or even cardiac failure in severe situations. Atrial fibrillation, atrial flutter, and ventricular fibrillation are some more typical forms of irregular heart rhythms. These irregularities can be divided into subclasses that describe various forms of cardiac arrhythmia. A precise classification of these categories could aid in the diagnosis and treatment of patients with heart disease. Arrhythmia may refer to either a sluggish or rapid heartbeat, as well as rhythms that aren't associated with a regular heartbeat. In clinical practice, an automatic identification of certain trends is critical. Certain identified symptoms of heart arrhythmia necessitate specialist professional experience for diagnosis.

1.2 ELECTROCARDIOGRAM

Electrocardiogram (ECG) recordings are often used to diagnose and predict cardiac arrhythmias in order to diagnose heart failure. In order to diagnose heart arrhythmia, health professionals can need to examine ECG recordings for a prolonged period of time. The ECG is a one-dimensional (1-D) signal that represents a time series and can be analyzed using machine learning techniques to identify anomalies automatically. Deep learning strategies have recently been established that offer substantial performance in the study of radiological images.

1.4 CONVOLUTIONAL NEURAL NETWORK

Convolutional neural networks (CNNs) (Robby Rohmantri, et al., 2020) have recently been shown to operate with multi-dimensional (1-D, 2-D, and in some cases 3-D) inputs, but they were originally designed to solve problems with two-dimensional inputs. 1-D CNNs are suggested for time series data, but they are less robust than 2-D CNNs. As a result, displaying time series data in a two-dimensional format could be advantageous for some machine learning tasks. As a result, a 2-D transformation must be added to ECG signals in order to make them ideal for deep learning approaches that use 2-D images as data. A 1-D signal can be converted into a 2-D spectrogram using the short-time Fourier transform (STFT), which encapsulates the time and frequency information in a single matrix. As a result, displaying time series data in a two-dimensional format could be advantageous for some machine learning tasks. As a result, a 2-D transformation must be added to ECG signals in order to make them ideal for deep learning approaches that use 2-D images as data. A 1-D signal can be converted into a 2-D spectrogram using the short-time Fourier transform (STFT), which encapsulates the time and frequency information in a single matrix. The 2-D spectrogram coefficients matrix may be useful for extracting robust features for cardiac ECG signal representation. This representation could enable CNN architectures (designed to run on 2-D inputs) to be used in the production of CVD-related automated systems.

CHAPTER 2

LITERATURE REVIEW

Aditi Gavhane et al. (2018) created a Heart Disease Prediction Model. We need to bring a mechanism in place to be able to predict the signs of a heart attack at an early stage and thereby avert it, using Machine Learning, given the rampant rise in heart stroke rates at juvenile levels. Since it is impractical for the average person to perform expensive checks like the ECG on a regular basis, there must be a method in place that is both convenient and effective in estimating the risk of heart disease. As a result, we suggest developing an app that can predict the risk of heart attack based on simple symptoms such as age, sex, pulse rate, and so on. The suggested method employs the deep learning algorithm neural networks, which has proved to be the most accurate and effective.

Aakash et al. (2018) Using Evolutionary Rule Learning, we can forecast heart disease. In a continuous basis, the healthcare company generates and collects a large volume of data. Data innovation allows for the extraction of data by the atomization of systems to obtain fascinating information. The Weighted Association Rule is a data mining method that aims to remove manual tasks while still retrieving data directly from electronic documents. This will continue to reduce the cost of care while also assisting with the saving of lives. In this article, we will look at a rule that can be used to estimate a patient's probability of coronary artery disease. The overwhelming majority of the rules aid in the best prediction of coronary disease, according to test reports.

Latha and Vetrivelan (2019) Using a partly measurable markov decision method, a model for predicting heart disease has been suggested (POMDP). In an emergency, fog computing alerts the patient via the doctor. During emergency cases, an ambulance is sent to the site of the patient. IFogSim, a fog computing system, provides the details to

the doctor. Fog computing in healthcare is a relatively recent field that is gaining popularity among researchers. Many studies concentrate on cardiovascular disease, or heart disease. Increased blood viscosity is a major risk factor for cardiovascular disease. Blood's extremely viscous composition prevents it from flowing freely, causing a blood flow resistance. High blood pressure, obesity, diabetes, and elevated blood viscosity are also risk factors for heart disease. The patient's wellbeing is tracked using POMDP's states, assumptions, values, and likelihood transitions. The regulation estimate is computed using states and timeslots in the POMDP model for heart disease prediction. Through several iterations, rewards are tabulated using policy approximations.

Malar Selvam et al. (2019) the type of illness that progresses to heart failure is shown by predict heart attack. Heart pain is not a sign in all people with coronary artery disease. Other factors that contribute to heart disease include R-Blood Pressure, S-Cholesterol, F-Blood Sugar, R-ECG, Thalach, Ex-Ang, Number of main Vessels blocked, and Thallium Scan The early detection and diagnosis of heart disease will save lives. In the prediction of Heart Disease based on parameters / variables, data mining strategies such as Naive Bayes, Decision Tree, K-nearest neighbor, and others are used. The aim of this study is to learn more about the different factors and their importance in detecting heart disease. It also looks at the various prediction methods and models that are currently in use. The work ends with a discussion of the main issues that are yet to be addressed.

Varnagar et al., (2019) Data Mining is being used to develop a method for early detection and prevention of ischemic heart disease. Finding the precise cause of a disease will aid in the cure of a patient at an early stage of risk. Ischemic heart disease is one of the leading causes of mortality in today's society. We used data mining classification methods on the stroke dataset in this study and suggested a guidance method for determining an individual's IHD risk level. If they have a high chance of IHD, adequate care and remedy will help them live longer. On the Ischemic Stroke Dataset, we used five data mining algorithms: Logistic Regression, Decision Tree, K nearest

Neighbor, Nave Bayes, and SVM, with the Support Vector Machine algorithm achieving the maximum precision of 97.91 percent.

Khalil and Maalmi (2019) Heart disease can be detected using real-time machine learning along with a big data approach. The death rate can be reduced by early diagnosis and regular screening of heart failure. Data from various channels, such as wearable sensor devices used in Internet of Things health tracking, broadcasting systems, and others, has been continuously producing an immense amount of data. The fusion of streaming big data analytics and machine learning is a game-changing technology that has the potential to revolutionize the healthcare industry, especially in terms of early detection of heart disease. This technology has the potential to be more efficient while still being less costly. To address this problem, this paper proposes a real-time heart disease prediction system based on Apache Spark, which is a powerful large-scale distributed computing platform that can successfully be used for streaming data events against machine learning through in-memory computations. Streaming processing and data collection and simulation are the two key components of the device. The first applies a classification algorithm to data events using Spark MLlib and Spark streaming to forecast heart disease. The large volume of processed data is stored in Apache Cassandra by the seconds.

BonomaliKhuntia and Radhanath Patra (2019) using data mining to investigate the rapid spread of heart disease. In today's world, heart disease is on the rise in both the older and younger generations of males and females. To address the epidemic of heart-related diseases, the need for appropriate interventions, as well as the establishment and enforcement of successful health-monitoring policies, should be stressed. In this article, we used the UCI machine learning library Cleveland heart disease index, which has 303 instances and 76 attributes, for early diagnosis and prevention using both traditional and advanced techniques. For the proposed approach, we used the information benefit principle to pick the best attribute and weka and python to process the selected features. This paper addresses a research void in heart disease prediction using Python's Anaconda Navigator, Spyder, and Weka network, which we have

focused on extensively. Feature discovery, numpy, panda's collection, decision tree classifier, KNN classifier, entropy, and other techniques and processes were used to train the model of heart datasets. Confusion matrix, benefit-index the results reveal that the decision tree classifier is the most efficient and sufficient classifier for predicting the UCI Cleveland heart dataset.

Priyal et al. (2018) CART-C can be used to predict heart disease. It is the colossal reason for our existence. As a result, it is important to assess the problem and treat it as soon as possible before it becomes fatal. It is difficult to find capable experts nowadays, and even a pro may not be aware of all the minor claims to fame, necessitating the use of a computerized system that can evaluate the issue a computerized system can aid in determining the severity of the problem and how quickly it can be addressed. This would also reduce a specialist's efforts to gather all of the individual's current subtle elements in order to search at the problem both inside and out. As a result, by reducing the time and effort required of a doctor and providing precise results, this illness can be treated quickly and a man's life can be saved. A robotized system for health research can improve restorative treatment while still lowering costs. During this investigation, we developed a system that can effectively recognize the tenets in order to predict the risk level of patients based on a given parameter affecting their health. The rules are often structured in light of the client's need. The framework's implementation is evaluated in terms of grouping precision, and the results show that the method has incredible ability in predicting the coronary hazard level even more precisely.

Nusrat Zahan et al. (2019) Linear Discriminant Analysis (LDA) is a simple and effective approach for pattern detection, and it is widely used for the initial discovery of diseases using EHR info. Nonetheless, two main factors often affect the efficiency of LDA for EHR data classification: LDA metrics are under-appreciated. (e.g., covariance matrix), and “linear inseparability” of the EHR data for classification. In this article, we suggest a new classifier for heart disease diagnosis called SDA (Sparse Discriminant Analysis). If the boundaries between classes are nonlinear or if subgroups are possible within each class, the time complexity of this algorithm will be minimized by optimal scoring analysis

of LDA, and it will be exhaustive to perform sparse discrimination by the combination of Gaussians. In general, our proposed approach is more suitable for the detection of heart attack patients with greater accuracy than previous therapies.

Dutkiewicz et al., (2018) the development of a heart disease detection model based on non-laboratory risk factors is critical in preventive treatment, particularly for high-risk patients. The model enables doctors and epidemiologists to accurately identify a person's heart condition. We want to use age, gender, rest blood pressure, maximal heart rate, and rest electrocardiography to create a non-invasive risk prediction model for automatic heart disease detection. We looked at four public databases that included 1071 people who were sent for a special X-ray of their heart arteries (i.e., to see if they are narrowed or blocked). A physical assessment and three non-invasive examinations were also performed on the participants. We use a generalized linear model with regularization paths and coordinate descent to predict the severity of heart disease. We observed a prediction accuracy of 72 percent also without laboratory-based data (e.g., serum cholesterol, fasting blood sugar), relative to 76 percent for other detailed models. This finding shows that only a few non-invasive variables based on recent advancements in data science will completely override conventional heart disease risk management methods.

CHAPTER 3

AIM AND SCOPE

3.1 AIM

This paper suggests a convolutional neural network to diagnose cardiac arrhythmia in this research. Arrhythmia simulation is useful for the accurate classification of cardiac arrhythmia. To provide a method that aids doctors in determining the kind of arrhythmia a patient has.

3.2 SCOPE

The AR simulation is useful for the classification of cardiac arrhythmias, and it has a high degree of accuracy. Further validation of the experimental procedure would result in a technique that is suitable for clinical use.

3.3 EXISTING SYSTEM

For detecting the early stages of heart attacks, an appropriate rule reduction algorithm is applied to the data sets. The algorithm's output rules are used to efficiently classify heart disease. Sensitivity, accuracy, precision, accurate and false positive rates, and other performance assessment measures are used to assess the performance.

3.4 PROPOSED SYSTEM

In this article, we suggest a two-dimensional (2D) (milad salim. et al., 2018) deep convolutional neural network approach for ECG arrhythmia classification (CNN). The classifier reached the maximum precision and lowest loss when the learning rate was 0.001 and the batch size parameter was 2500. The biggest benefit of CNN over its counterparts is that it identifies crucial characteristics without the need for human intervention. It discovers unique characteristics for each class on its own. In addition, CNN is computationally efficient.

CHAPTER 4

METHODOLOGY

4.1 IMPLEMENTATION

The aim of the proposed research is to use a convolutional neural network to diagnose and identify cardiac arrhythmia (Yonghui Dai., et al., 2020).

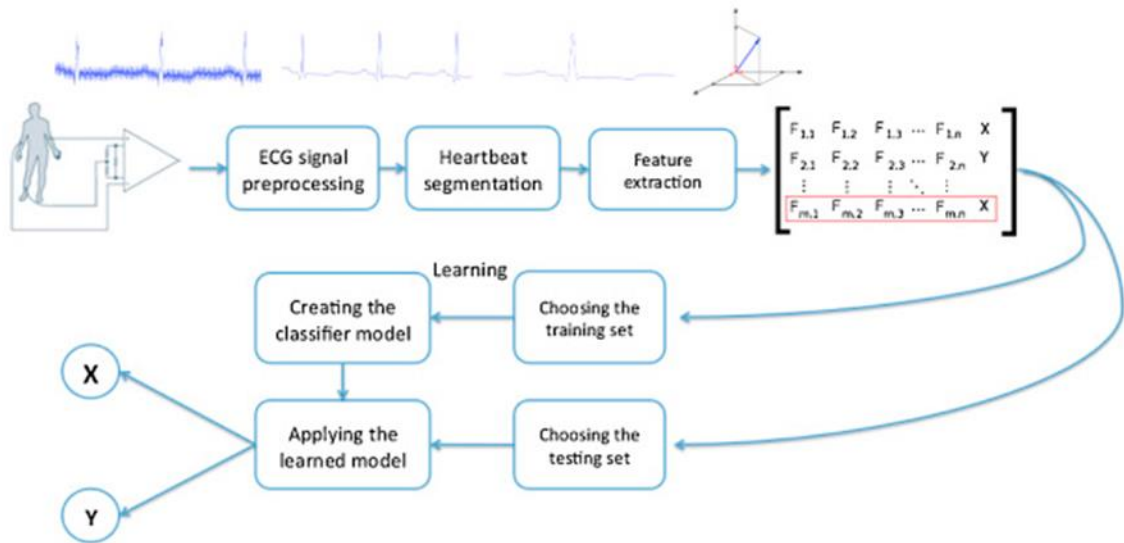


Fig: 4.1 System Architecture

4.2 MODULE DESCRIPTION

4.2.1 Data Collection

The data is gathered in this process in order to analyze the best proposed procedure. The UNSW-NB15 dataset (Pranav Rajpurkar., et al., 2017) was used in our study to evaluate the accuracy of the proposed methods.

4.2.2 Data Preprocessing

- In this step, the collected data is preprocessed. Data Transformation, Data Normalization, and Feature Selection are all part of it.

- The method of converting abstract data into numerical values is known as transformation. Symbolic data such as ftp, http, and so on are used in the KDD Cup 99 dataset. The numerical values of this symbolic data are transformed.
- The method of encoding odd values into a normal range [0-1] is known as data normalization.
- Feature selection is a technique for finding important features and deleting those that aren't in order to speed up the process.

4.2.3 Classifier Training

The supervised learning classifier method (Ian Good fellow. et al., 2017) is used in our research. After selecting the relevant features the proposed classifiers are implemented to identify the anomalies.

4.2.4 CNN Algorithm

A convolutional neural network (CNN) is a type of artificial neural network that analyses data using perceptron, which are a machine learning unit algorithm. Picture processing, natural language processing, and other cognitive functions will all benefit from CNNs. CNN image classifications takes an input image, processes it, and categories it into different groups.

Convolution is divided into four steps:

- Align the function and the display
- Multiply each image pixel by the corresponding attribute pixel
- Add the values and calculate the amount
- Divide the sum by the feature's total number of pixels

4.3 MACHINE LEARNING

Machine learning is a branch of artificial intelligence (AI) (Tara soman. et al., October 2016). That allows computers to learn and develop on their own without having to be directly programmed. Machine learning is concerned with the development of computer programmes that can access data and learn on their own. The learning process starts with insights or evidence, such as experiences, direct experience, or instruction, so that

we can search for correlations in data and make informed choices in the future based on the examples we have. The primary goal is for computers to learn on their own, without the need for human interference, and to adapt their behaviour accordingly.

4.4 MACHINE LEARNING METHODS

There are two types of machine learning algorithms: supervised and unsupervised. (Raghu et al., 2018)

4.4.1 Supervised machine learning algorithms

It will use labelled examples to adapt what it has observed in the past to current data in order to forecast future events. The learning algorithm creates an assumed function to make assumptions about the output values based on the study of a known training dataset. After enough testing, the machine will have goals with any new data. The learning algorithm may also compare its output to the right, expected output and detect errors, allowing the model to be modified as needed.

4.4.2 Unsupervised machine learning algorithms

It's used where the data being used for training isn't categorised or labelled. Unsupervised learning investigates how systems can infer a function from unlabelled data to represent a hidden framework. The method does not determine the correct performance, but it examines the data and can infer hidden constructs from unlabelled data using datasets.

4.4.3 Semi-supervised machine learning algorithms

Since they use both labelled and unlabelled data for preparation – usually a limited number of labelled data and a significant amount of unlabelled data – it falls in supervised and unsupervised learning. This approach will significantly increase learning accuracy in systems that use it. Semi-supervised learning is typically used where the obtained marked data necessitates the use of trained and appropriate tools to train / learn from it. Obtaining unlabelled data, on the other hand, usually does not necessitate additional funding.

4.4.4 Reinforcement machine learning algorithms

It's a kind of learning that communicates with its surroundings by performing activities and detecting errors or rewards. The most important features of reinforcement learning are trial and error search and deferred compensation. This approach enables machines and software agents to automatically evaluate the best behaviour in a given situation in order to increase their performance. For the agent to understand which behaviour is better, simple incentive feedback is required; this is known as the reinforcing signal.

4.4 ADVANTAGES OF MACHINE LEARNING

Machine Learning unquestionably aids people in becoming more imaginative and productive at work. Essentially, you can use Machine Learning to assign very complicated or monotonous tasks to the robot, ranging from scanning, storing, and filing paper documents such as invoices to sorting and editing photographs.

Self-learning machines can handle complicated tasks in addition to these relatively basic tasks. For e.g., recognising error patterns is one of them. This is a significant benefit, particularly in industries that rely on continuous and error-free output, such as manufacturing although even specialists cannot always predict when and how a manufacturing error in a plant fleet occurs, Machine Learning allows for early detection of the error, which saves time and resources. In the medical sector, self-learning systems are also widely used. Apps would be able to alert a patient whether his doctor tries to administer a medication that he cannot handle in the future after "consuming" massive volumes of evidence (medical journals, tests, etc.). This "intelligence" also ensures that the app will suggest alternate solutions, such as those that take into account the patient's genetic criteria.

4.5 SYSTEM ANALYSIS

Feasibility study

In this process, (Chandra, et al., 2018.) The project's viability is assessed, and a business idea is presented, along with a very general project plan and some cost estimates. A feasibility study of the planned system is to be carried out during system review. This is to ensure that the proposed system would not cause the corporation any problems. A basic understanding of the system's main specifications is needed for feasibility study. The feasibility study takes into account three major factors. Economical Feasibility

- Technical Feasibility
- Social Feasibility

Economical Feasibility: This research is being carried out to determine the system's economic effects on the enterprise. The amount of money the organization has to invest in the system's research and development is minimal. It is necessary to justify the expenses. As a result, the built device came in under budget, which was made possible by the fact that the majority of the technologies used were readily available. The personalized packages were the only ones that needed to be bought.

Technical Feasibility: This research is being carried out to determine the system's technological viability, or technical specifications. Any system that is created does not place a large demand on the existing technological capital. As a result, the customer would be subjected to heavy demands. A feasibility report assesses a project's chances of completion.

Social Feasibility: The aim of the analysis is to determine the user's level of approval of the scheme. This involves the method of teaching the customer how to effectively use the device. The customer should not be afraid of the device, but rather consider it as a requirement. The approaches used to teach and familiarize the customer with the system are solely responsible for the degree of adoption by the users. His self-esteem

must be boosted so that he can provide positive feedback, which is encouraged because he is the system's final customer.

4.7 REQUIREMENT SPECIFICATION

4.7.1 Hardware requirements

Physical computing tools, also known as hardware, are the most basic set of specifications specified by any operating system or software application. The below are the minimum hardware requirements:

1. Processor : Pentium IV
2. RAM : 8 GB
3. Processor : 2.4 GHz
4. Main Memory : 8GB RAM
5. Hard Disk Drive : 1tb
6. Keyboard : 104 Keys

4.7.2 Software requirements

Software specifications are concerned with specifying the resources and prerequisites that must be implemented on a device in order for an application to work. The below are the minimum programmed requirements:

1. Front end : python
2. Dataset : csv
3. IDE : anaconda
4. Operating System : Windows 10

4.8 TECHNOLOGIES USED:

4.8.1 Python Language

Guido Rossum developed Python in 1989 as an object-oriented programming language. It's perfect for quick prototyping of complicated software. It can be extended to C or C++ and has interfaces to several OS device calls and libraries. NASA, Google, YouTube, Bit Torrent, and other major corporations use the Python programming language. Python programming is commonly used in specialized areas of computer science such as Artificial Intelligence, Natural Language Generation, Neural Networks, and others. Python places a strong emphasis on code readability, and this class will teach you the fundamentals of the language.

4.8.2 Python Programming Characteristics

- It is a platform independent scripted language with full access to operating system API's
- It offers greater run-time freedom than most programming languages
- It has the simple text manipulation facilities of Perl and Awk
- It is a platform independent scripted language with full access to operating system API's
- A Python module can contain one or more classes and free functions.
- Python libraries are cross-platform compliant with Linux, Macintosh, and Windows.
- Python can be compiled to byte-code for massive applications.
- Python embraces both functional and formal programming, as well as object-oriented programming (OOP). It supports interactive mode that allows interacting Testing and debugging of snippets of code
- In Python, since there is no compilation step, editing, debugging and testing is fast.

4.8.3 Applications of Python Programming

- **Web Applications**

- Frameworks and CMS (Content Management Systems) based on Python can be used to build scalable Web Apps. Django, Flask, Pyramid, Plone, and Django CMS are some of the most common tools for developing Web Apps. Python is used to power websites such as Mozilla, Reddit, Instagram, and PBS.

- **Scientific and Numeric Computing**

Python has a plethora of libraries for scientific and numerical programming. In general purpose programming, libraries such as SciPy and NumPy are used. There are also specialized libraries, such as EarthPy for earth science and AstroPy for astronomy, among others. Computer learning, data mining, and deep learning both use the term extensively.

- **Creating software Prototypes**

In comparison to compiled languages such as C++ and Java, Python is sluggish. If resources are scarce and performance is needed, this may not be the best option. Python, on the other hand, is a fantastic tool for prototyping. As an illustration: You can start by making a demo of your game using Pygame (a game-making library). If the prototype appeals to you, you can develop the game using a programming language such as C++.

- **Good Language to Teach Programming**

Python is used by many companies to teach programming to kids and newbies. It is a good language with a lot of features and capabilities. Yet, it's one of the easiest language to learn because of its simple easy-to-use syntax.

4.8.4 Opencv Package

Python (Weifang Sun., et al., 2018) is a general-purpose programming language created by Guido van Rossum that has quickly gained popularity due to its ease of use and readability of code. It allows a programmer to communicate his thoughts in fewer lines of code while maintaining readability.

Python is slower than other languages like C/C++. Python can also be conveniently expanded with C/C++, which is an important feature. This function allows us to write computationally intensive C/C++ code and wrap it in a Python wrapper, which we can then use as Python modules. This provides us with two benefits: first, our code is as quick as original C/C++ code (due to the real C++ code running in the background), and second, Python is very simple to code. OpenCV-Python is a Python wrapper over the original C++ implementation. And the support of Numpy makes the task easier. Numpy is a numerical operations library that is highly structured. It has a syntax similar to that of MATLAB. Both array structures in OpenCV are translated to and from Numpy arrays. So, whatever Numpy operations you can do, you can combine them with OpenCV to expand your arsenal of weapons. Other libraries that allow Numpy, such as SciPy and Matplotlib, can also be used for this.

As a result, OpenCV-Python is an excellent platform for quickly prototyping computer vision issues.

4.8 ANACONDA

Anaconda is a free and open source Python and R programming language distribution that is simple to set up. Anaconda is a software environment for mathematical computation, computer science, predictive analysis, and deep learning.

4.9.1 Anaconda Navigator

Anaconda Navigator is a graphical user interface (GUI) for the desktop that comes with the Anaconda distribution. It helps us to use the Anaconda distribution's software and control conda packages, environments, and networks without having to use command-line commands. It's compatible for Windows, Mac OS X, and Linux.

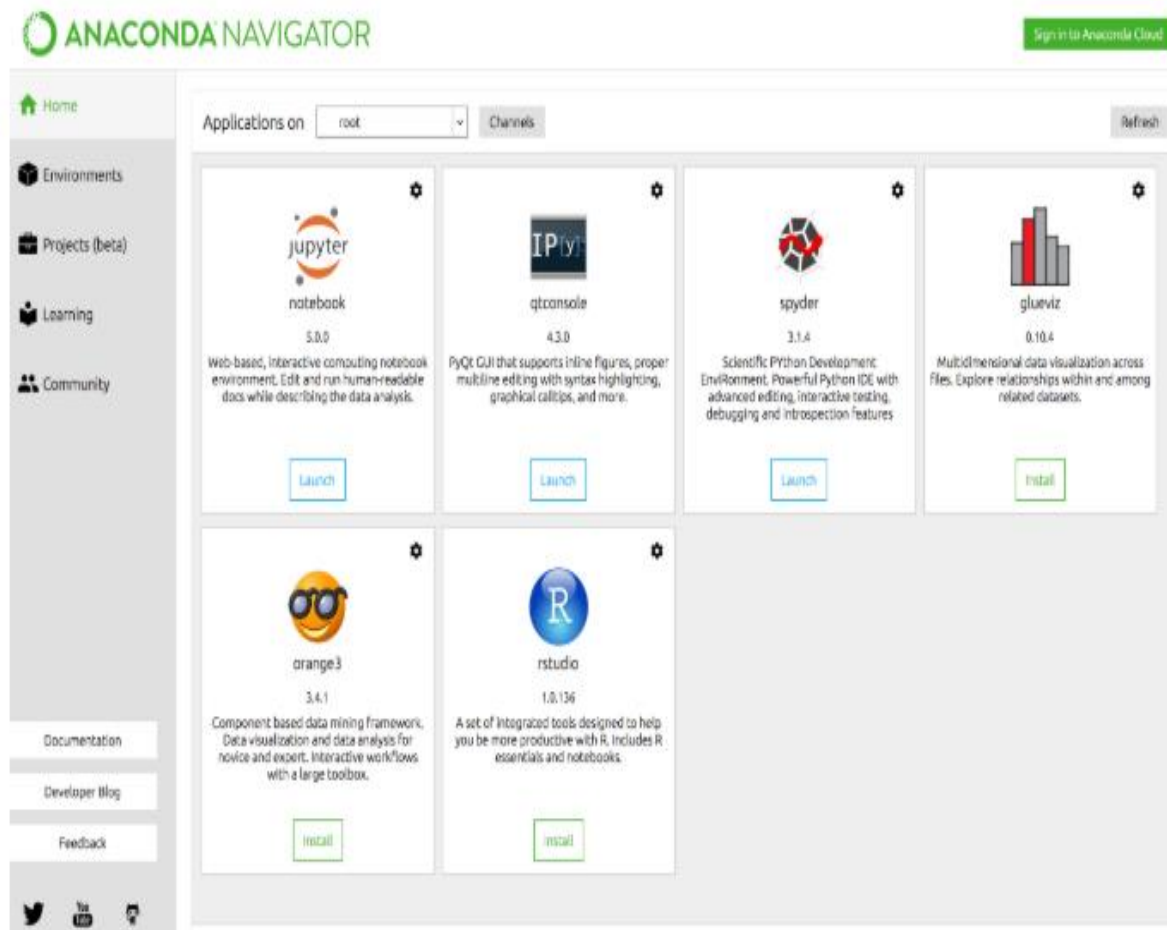


Fig 4.2 Anaconda Navigator

4.10 ANACONDA DISTRIBUTION

Anaconda 5.3 is the most recent distribution, which was launched in October of 2018. It includes the conda module, an environment manager, and a library of over 1000 open source packages, all of which come with free community support.

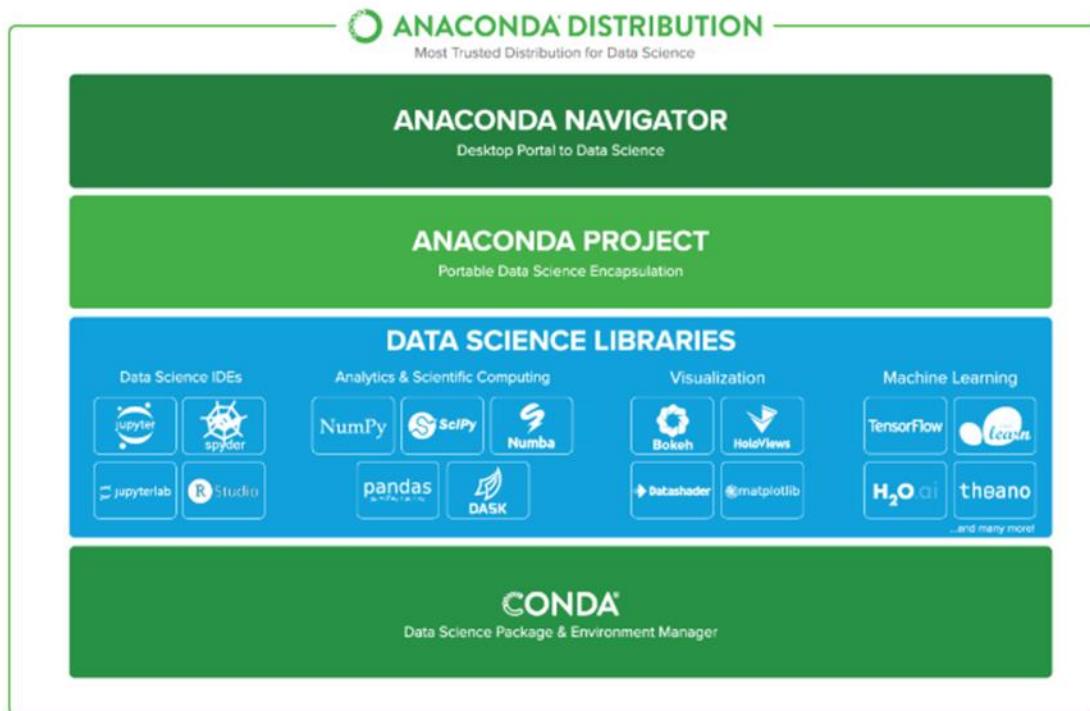


Fig 4.3 Anaconda Distribution

4.10.1 Applications Provided In Anaconda Distribution

Along with Anaconda Navigator, the Anaconda distribution includes the following features.

- Jupyter Notebook
- Qt Console
- Spyder
- Glueviz
- Orange3
- RStudio
- Visual Studio Code

JupyterLab: Based on the Jupyter Notebook and Architecture, this is an extensible working platform for collaborative and reproducible computing.

Jupyter Notebook: This is an immersive programming notebook that runs on the internet. When explaining the data processing, we can edit and run human-readable docs.

Qt Console: The PyQt GUI supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Spyder: Spyder is a Python Development Environment for scientists. It's a robust Python IDE with functionality such as advanced editing, dynamic checking, debugging, and introspection.

VS Code: It's a streamlined code editor that includes features for debugging, mission execution, and version control.

Glueviz: This is used to visualize multidimensional data spanning several files. It looks for connections between and beyond similar datasets.

Orange 3: It's a data-mining platform built on components. This will be used for data processing and visualization. Orange 3's workflows are very collaborative and have a wide toolbox.

Rstudio: It's a set of resources that work together to help you get things done with R. It contains R basics as well as notebooks.

4.11 NEW FEAYTURES OF ANACONDA 5.3

Anaconda 5.3 is compiled with Python 3.7, which takes advantage of Python's speed and feature enhancements.



Fig 4.4 Anaconda 5.3

- ***Better Reliability:*** Anaconda's reliability has been enhanced with the new update, which now captures and stores package metadata for installed packages.

Enhanced CPU Performance: In Anaconda 5.3, the Intel Math Kernel Library 2019 for Deep Neural Networks (MKL 2019) was added. MKL 2019 for Deep Neural Networks is available to Tensor flow users. These Python binary packages are offered to help you get the most out of your CPU.

- **New packages are added:** The latest release includes over 230 packages that have been revised or added.

- **Work in Progress:** There is a casting bug in Numpy with Python 3.7 but the team is currently working on patching it until Numpy is updated.

4.12 TESTING

In a broad sense, we may define system testing as a type of testing whose primary goal is to ensure that a system runs smoothly and efficiently. The testing method is applied to a program with the aim of discovering an unheard-of mistake, an error that may have harmed the software's future. Good test cases are those that have a high chance of finding and correcting errors. This good test aids in the identification of errors that are still unknown. (Varnagar C et al., 2019)

4.12.1 Test case

As previously mentioned, testing is the process of identifying all potential flaws in the finalized software product. Testing aids in the verification of sub-assemblies, parts, assembly, and the final product. The program is put through a series of tests with the aim of ensuring that it meets business requirements and user expectations and that it does not fail unexpectedly. Today, a variety of tests are used. Each test form is designed to address a particular testing need.

4.12.2 Testing Techniques

A test plan is a document that outlines the strategy, scope, resources, and schedule for conducting targeted testing exercises. It assists in identifying almost every other test object, the features to be evaluated, the tasks, how everyone will perform each task, how autonomous the tester is, the setting in which the test is taking place, the test method, as well as both the end criteria that are used, as well as their rationale for choosing and any type of danger that necessitates emergency planning. It can also be referred to as a record of the test planning process. In most cases, test plans are created with the help of significant feedback from test engineers.

- **Unit testing:** Unit testing entails the development of test cases (Nusrat Zahan et al. 2019) which aids in the verification of internal programme logic. The internal code and all of the decision branches are validated. It occurs after the completion of each individual unit. It is also considered after the person has been unified and before integration. As a result, the unit test conducts a simple level test at the component level, testing the specific business operation, device configurations, and so on. The unit test ensures that the process's one-of-a-kind path is followed exactly as documented, with clearly specified inputs and expected outcomes.
- **Functional testing:** The functional tests help in providing the systematic representation that functions tested are available and specified by technical requirement, documentation of the system and the user manual.
- **System testing:** System testing, as the name suggests, is the type of testing in which ensure that the software system meet the business requirements and aim. Testing of the configuration is taken place here to ensure predictable result and thus analysis of it. System testing is relied on the description of process and its flow, stressing on pre driven process and the points of integration.
- **Integration testing:** These measures are intended to assess whether the combined software items truly execute as a single program or application. Since the testing is event-driven, it is concerned with the fundamental outcome of the field. The integration tests show that the components were individually satisfied, as shown by good unit testing, and that the components are appropriate and functional. This method of testing is designed to reveal any problems that may arise as a result of the components' combination.
- **White box testing:** White box testing is a method of testing in which the device software's internal components are exposed and can be processed by the tester. As a result, it is a difficult research procedure. To detect a potential flaw or error, the

tester tests the entire data structure, components, and so on. When the black box is unable to detect a bug, this method is employed. It is a more complicated method of testing that takes longer to implement.

- **Black box testing:** Black box testing is a method of software testing in which the internal components of the software are concealed and only the system's input and output are used to detect bugs. As a result, it is a basic method of research. This form of testing can also be handled by a programmer with basic programming skills. When opposed to white box-checking, it takes less time. It works well with software that is less complicated and straightforward in nature. In addition, it is less expensive than white box testing.
- **Acceptance testing:** Acceptance by the users Testing is an important aspect of every project, and it necessitates active involvement from the end-user. It also ensures that the device satisfies the functional specifications.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 RESULTS

This paper proposes an automatic classification for arrhythmia detection is presented, and its performance is evaluated by measuring the system sensitivity, specificity and error after the all MIT-BIH positive and negative cases have been normalized. The system errors are caused by two main reasons: 1-The pre-processing and filtering processes in some cases, are not enough to remove all type of noise affecting the ECG signal, and sometimes they cause a signal distortion, so the QRS detection and the feature extraction procedure don't perform as well as they are supposed to do; 2- Other types of arrhythmia are rarely present in some records; thus we haven't sufficient cases to be inserted in the dataset and they resulted as unrecognized patterns from our system.

The future work will focus on developing our ECG filtering method to reduce the first cause of system error, and we will try to find more data containing different cases of arrhythmia to overcome the second one. We hope this system can be further developed, in order to realize a low- cost, high performance, simple to use and portable equipment for ECG signal monitoring.(Jun., et al., 2018)

```
In [1]: import os
        for dirname, __, filenames in os.walk(r'C:\Users\RE_CS_2\Downloads\Input'):
            for filename in filenames:
                print(os.path.join(dirname, filename))

        # Any results you write to the current directory are saved as output.

C:\Users\RE_CS_2\Downloads\Input\mitbih_test.csv
C:\Users\RE_CS_2\Downloads\Input\mitbih_train.csv
C:\Users\RE_CS_2\Downloads\Input\ptbdb_abnormal.csv
C:\Users\RE_CS_2\Downloads\Input\ptbdb_normal.csv
```

```
In [2]: import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        from sklearn.metrics import classification_report
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import f1_score
        from sklearn.metrics import confusion_matrix
        from keras.utils.np_utils import to_categorical
        from sklearn.utils import class_weight
        import warnings
        warnings.filterwarnings('ignore')
```

Using TensorFlow backend.

Fig 5.1 Python programming

```
In [3]: train_df=pd.read_csv(r'C:\Users\RE_CS_2\Downloads\Input\mitbih_train.csv',header=None)
test_df=pd.read_csv(r'C:\Users\RE_CS_2\Downloads\Input\mitbih_test.csv',header=None)
train_df.head(10)
```

Out[3]:

	0	1	2	3	4	5	6	7	8	9	...	178	179	180	181	182	183	184	185	186	18
0	0.977941	0.926471	0.681373	0.245098	0.154412	0.191176	0.151961	0.085784	0.058824	0.049020	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
1	0.960114	0.863248	0.461538	0.196581	0.094017	0.125356	0.099715	0.088319	0.074074	0.082621	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
2	1.000000	0.659459	0.186486	0.070270	0.070270	0.059459	0.056757	0.043243	0.054054	0.045946	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
3	0.925414	0.665746	0.541436	0.276243	0.196133	0.077348	0.071823	0.060773	0.066298	0.058011	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
4	0.967136	1.000000	0.830986	0.586854	0.356808	0.248826	0.145540	0.089202	0.117371	0.150235	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
5	0.927461	1.000000	0.626943	0.193437	0.094991	0.072539	0.043178	0.053541	0.093264	0.189983	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
6	0.423611	0.791667	1.000000	0.256944	0.000000	0.277778	0.465278	0.520833	0.472222	0.555556	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
7	0.716814	0.539823	0.283186	0.129794	0.064897	0.076696	0.026549	0.032448	0.008850	0.044248	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
8	0.874214	0.849057	0.480084	0.058700	0.090147	0.310273	0.387841	0.385744	0.389937	0.392034	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
9	1.000000	0.996086	0.694716	0.336595	0.238748	0.268102	0.191781	0.174168	0.164384	0.187867	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.

10 rows x 188 columns

Fig 5.2 Mit-bih dataset training

The collected data mentioned above is taken from UNSW-NB15.

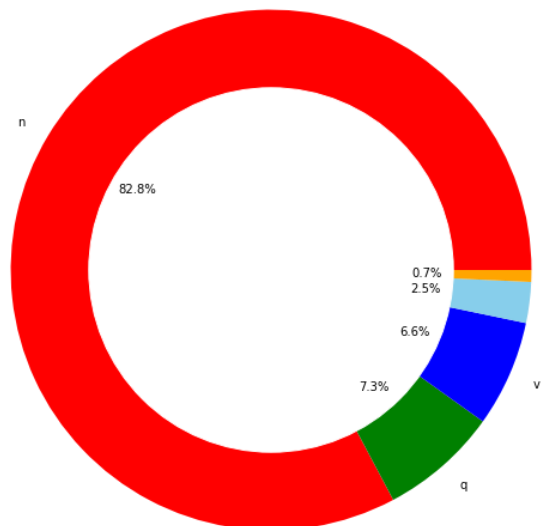


Fig 5.3 Removing null values

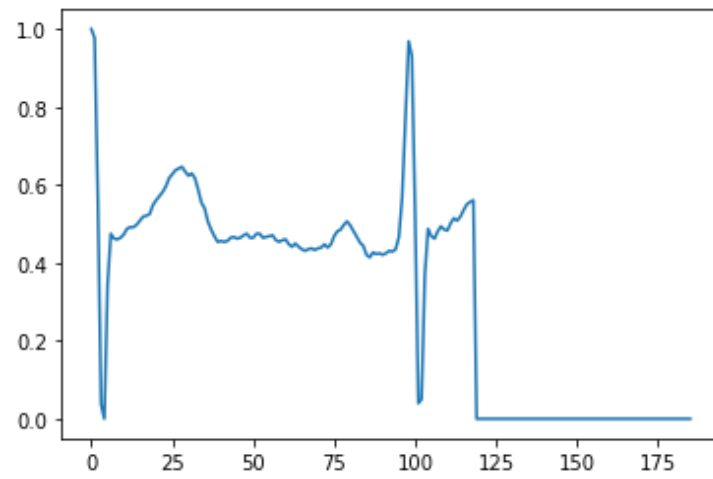


Fig 5.4 Filtered values

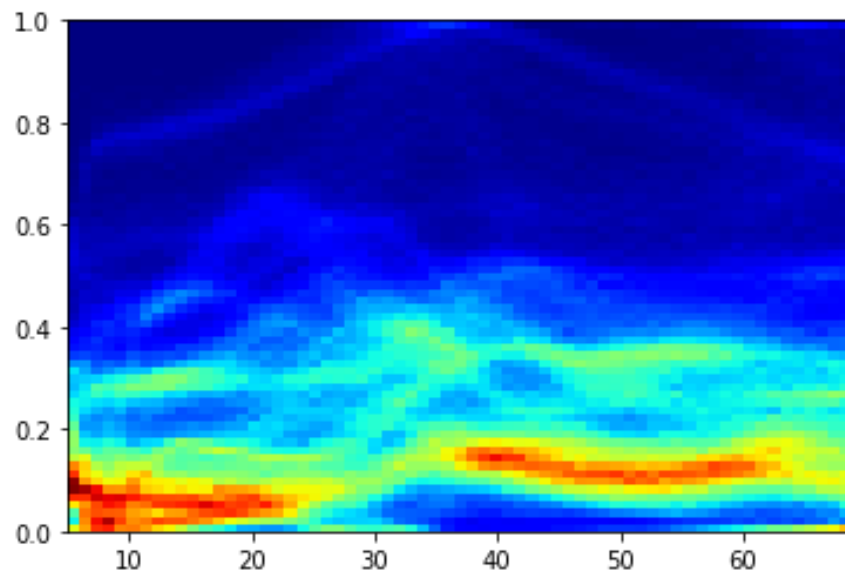


Fig 5.5 Histogram ECG signal

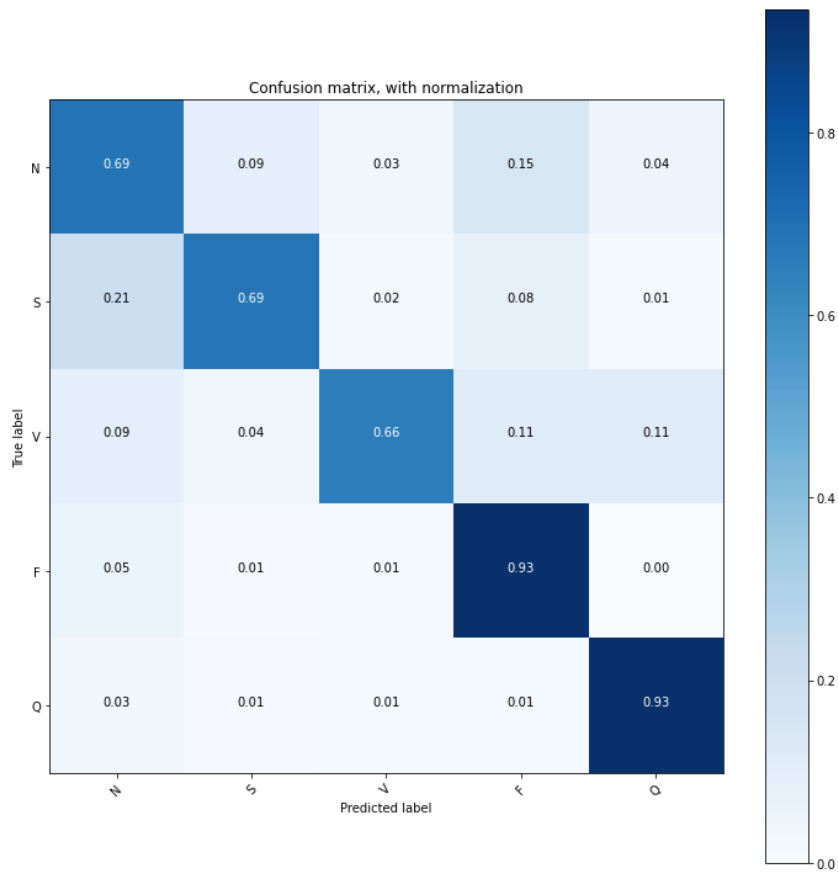


Fig 5.6 Confusion matrix with normalization

```

Epoch 1/10
44/44 [=====] - 699s 16s/step - loss: 0.8700 - acc: 0.6842 - val_loss: 0.7655 - val_acc: 0.7661
Epoch 2/10
44/44 [=====] - 636s 14s/step - loss: 0.6459 - acc: 0.7379 - val_loss: 1.5847 - val_acc: 0.5919
Epoch 3/10
44/44 [=====] - 635s 14s/step - loss: 0.5317 - acc: 0.7971 - val_loss: 0.8290 - val_acc: 0.7358
Epoch 4/10
44/44 [=====] - 682s 15s/step - loss: 0.4019 - acc: 0.8369 - val_loss: 0.6475 - val_acc: 0.8428
Epoch 5/10
44/44 [=====] - 697s 16s/step - loss: 0.3454 - acc: 0.8584 - val_loss: 0.6966 - val_acc: 0.7813
Epoch 6/10
44/44 [=====] - 682s 16s/step - loss: 0.2342 - acc: 0.9044 - val_loss: 0.3765 - val_acc: 0.8892
Epoch 7/10
44/44 [=====] - 682s 15s/step - loss: 0.2173 - acc: 0.9141 - val_loss: 1.0270 - val_acc: 0.8201
Epoch 8/10
44/44 [=====] - 683s 16s/step - loss: 0.1962 - acc: 0.9245 - val_loss: 3.0318 - val_acc: 0.6847
Epoch 9/10
44/44 [=====] - 683s 16s/step - loss: 0.1530 - acc: 0.9432 - val_loss: 0.4752 - val_acc: 0.8930
Epoch 10/10
44/44 [=====] - 680s 15s/step - loss: 0.1607 - acc: 0.9384 - val_loss: 0.3236 - val_acc: 0.8778

```

Fig 5.7 Training

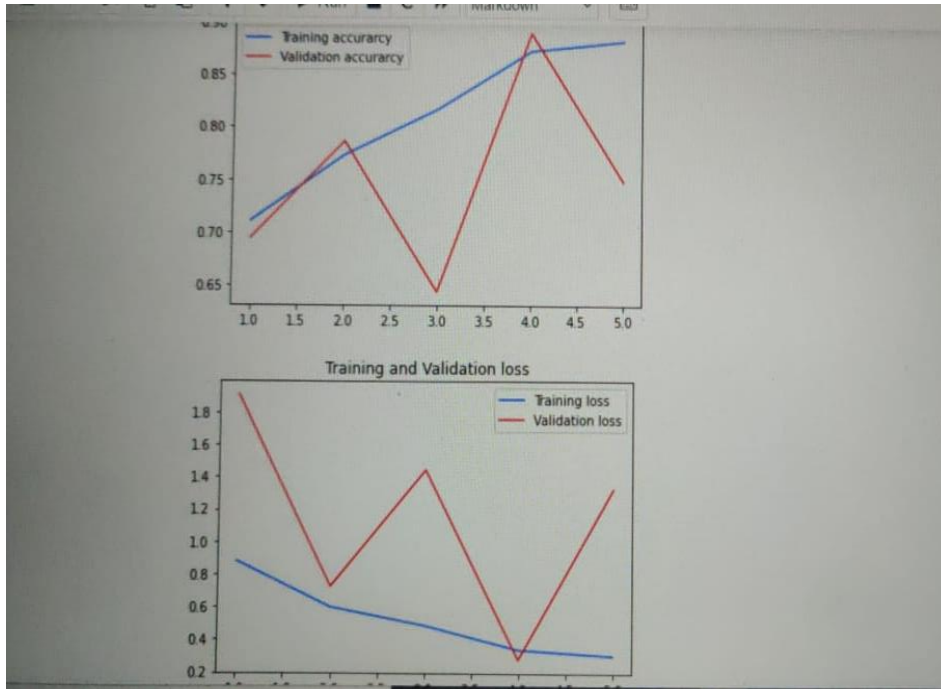


Fig 5.8 Training accuracy and loss

The dataset mentioned is collected from UNSW-NB15. This data was trained and uploaded as fig 5.2 MIT-BIH Dataset training. In this dataset, null values were removed. This removed nullset is mentioned fig 5.3 Removing nullset. After this, the ECG signals of filtered values are mentioned in fig 5.4 filtered values. Histogram of ECG is mentioned in fig 5.5. A confusion matrix is used for summarizing the performance of classification mentioned in fig 5.6. Accuracy loss is obtained as 99.11. This is mentioned in the fig 5.8 training accuracy and loss.

5.2 DISCUSSION

(Dutkiewicz, et al., 2018) developed a heart disease detection model based on non-laboratory risk factors is critical in preventive treatment, particularly for high-risk patients. To create a non-invasive risk prediction model for automatic heart disease detection. We looked at four public databases that included 1071 people who were sent for a special X-ray of their heart arteries. We observed a prediction accuracy of 72 percent also without laboratory-based data. Varnagar C et al., (2019) Data Mining is being used to develop a method for early detection and prevention of ischemic heart disease. Finding the precise cause of a disease will aid in the cure of a patient at an early stage of risk, they used five data mining algorithms: Logistic Regression, Decision Tree, K nearest Neighbor, Nave Bayes, and SVM, with the Support Vector Machine algorithm achieving the maximum precision of 97.91 percent. The proposed CNN algorithms with other classification methods of arrhythmia using ECG signals it achieved 97.91% average sensitivity, 99.61% specificity, 99.11% average accuracy, and 98.59% positive predictive value (precision).

CHAPTER 6

SUMMARY AND CONCLUSION

6.1 SUMMARY

In this project we have developed a cardiac arrhythmia detector using 2D CNN.

The data is gathered in this process in order to analyses the best proposed procedure.

The collected data is preprocessed. The numerical values of this symbolic data are transformed .Encoding odd values into a normal range [0-1]. For finding important features and deleting those that aren't in order to speed up the process. The supervised learning classifier method is used in our research. After selecting the relevant features the proposed classifiers are implemented to identify the anomalies. A convolutional neural network (CNN) is a type of artificial neural network that analyses data using perceptron's, which are a machine learning unit algorithm.

6.2. CONCLUSION

In this paper, we proposed a CNN-based classification model for automatic classification of cardiac arrhythmias using ECG signals. An accurate taxonomy of ECG signals is extremely helpful in the prevention and diagnosis of CVDs. Deep CNN has proven useful in enhancing the accuracy of diagnosis algorithms in the fusion of medicine and modern machine learning technologies. The proposed CNN-based classification algorithm, using dataset, can classify eight kinds of Arrhythmia and it achieved 97.91% average sensitivity (Varnagar C et al., (2019), 99.61% specificity, 99.11% average accuracy, and 98.59% positive predictive value (precision). These results indicate that the prediction and classification of arrhythmia with ECG representation as spectrograms and the CNN model is a reliable operative technique in the diagnosis of CVDs. The proposed scheme can help experts diagnose CVDs by referring to the automated classification of ECG signals. The present research uses only a single-lead ECG signal. The effect of multiple lead ECG data to further improve experimental cases will be studied in future work.

REFERENCES

1. AKAN, "Arrhythmia Detection on ECG Signals by Using Empirical Mode Decomposition" in Proceedings of the third international workshop on advanced issues of e-commerce and web-based information systems, IEEE 2018.
2. Anam Mustaqeem, Syed Muhammad Anwar, Muahammad Majid, "Multiclass classification of cardiac arrhythmia using improved feature selection and svm invariants", March 2018.
3. Billeci.L, Costi, Lombardi, Chiarugi, and Varanini, "Automatic detection of atrial fibrillation and other arrhythmias in ECG recordings acquired by a smartphone device, Electron" vol. 7, (Issue). 9, 2018.
4. Chandra, Sastry and Jana S, "Robust Heartbeat Detection from Multimodal Data via CNN-based Generalizable Information Fusion" in Journal of the American Society for Information Science and Technology, 29th June 2019
5. Chandra, A. Sharma, and G. K. Singh, "Feature extraction of ECG signal," J. Med. Eng. Technol., vol. 42, no. 4, pp. 306–316 , 2018.
6. Deep Learning Approach to Cardiovascular Disease Classification Employing Modified ECG Signal from Empirical Mode Decomposition, ELSEVIER, Biomedical Signal Processing and Control, 2019.
7. Dohare.A.K, V. Kumar, and R. Kumar, Detection of myocardial infarction in 12 lead ECG using support vector machine, Applied Soft Computing, vol. 64, pp. 138-147 , 2018.

8. Elif IZCI, Mehmet Akif OZDEMIR, Reza SADIGHZADEH, Aydin Akan, Arrhythmia Detection on ECG signal by using empirical mode decomposition, medical Technologies congress, TIPTEKNO'18, December 2018.
9. Ian Goodfellow, Y. Bengio, and A. Courville, Deep learning. 2017
10. Izci.E, Ozdemir .M. A, Degirmenci.M, and Akan. A, "Cardiac arrhythmia detection from 2d ecg images by using deep learning technique," TIPTEKNO 2019 - Tip Teknol. , pp. 1–4, 2018.
11. Izci.E, M. A. Ozdemir, M. Degirmenci, and A. Akan, "Cardiac arrhythmia detection from 2d ecg images by using deep learning technique," TIPTEKNO 2019 - Tip Teknol. Kongresi, pp. 1–4, 2019.
12. Jun.T.J, H. M. Nguyen, D. Kang, D. Kim, D. Kim, and Y.-H. Kim, "ECG arrhythmia classification using a 2-D convolutional neural network," 2018.
13. Karthik.R, Dhruv Tyagi, Amogh Raut, Soumya Saxena, Rajesh Kumar M, "Implementation of Neural Network and feature extraction to classify ECG signals", EP Europace, Volume 21, Issue 8, August 2019.
14. Kuo.P.H and C. J. Huang, "A high precision artificial neural networks model for short-Term energy load forecasting," Energies, vol. 11, (Issue). 1, pp. 1–13, 2017.
15. Milad Salem, Shayan Sean taheir, ECG "Arrhythmia Classification Using Transfer Learning from 2-Dimensional Deep CNN Features" IEEE, October 2018.

16. Oh.S.L, E. Y. K. Ng, R. S. Tan, and U. R. Acharya, "Automated diagnosis of arrhythmia using combination of CNN and LSTM techniques with variable length heart beats," Comput. Biol. Med., no. June 2018.
17. Padhy.S and S. Dandapat, Third-order tensor based analysis of multilead ECG for classification of myocardial infarction, ELSEVIER, Biomedical Signal Processing and Control, vol. 31, pp. 71-78, 2017.
18. Pranav Rajpurkar, Awni Y Hannun, Masoumeh Haghpanahi, Codie Bourn, Andrew Y Ng, "Cardiologist level arrhythmia detection with convolution neural network", July 2017.
19. Pranav Rajpurkar, Awni Y Hannun, Masoumeh Haghpanahi, Codie Bourn, Andrew Y Ng, Nature Medicine 2019, Mintu P Turakhia, Geoffrey H Tison, "Cardiologist Level Arrhythmia detection and classification in ambulatory electrocardiograms using deep learning Neural Network", January 2019.
20. Raghu N, Manjunathan K N, "Arrhythmia Detection Using machine Learning techniques", August 2019
21. Sai Krishna.V, Nithya Kalyani.A, "Prediction of Cardiac Arrhythmia using Artificial Neural Network" in International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8, Issue-1S4, June 2019.
22. Selen Ozdalilib, Ali Isina "Cardiac arrhythmia detection using deep learning" in 9th International Conference on Theory and Application of Soft Computing, Computing with Words and Perception, ICSCCW 2017, 24-25 Hungary , August 2017.

23. THARA SOMAN, PATRICK O. BOBBIE, "Classification of Arrhythmia Using Machine Learning Techniques" in International journal of man-machine studies, October 2016,
24. Varghees.V.N and K. Ramachandran, Effective heart sound segmentation and murmur classification using empirical wavelet transform and instantaneous phase for electronic stethoscope, IEEE Sensors Journal, vol. 17, (Issue). 12, pp. 3861-3872, 2017.
25. Wang.H et al., "A high-precision arrhythmia classification method based on dual fully connected neural network," Biomed. Signal Process. Control, vol. 58, p. 101874, 2020.
26. Weifang sun, nianyin zeng, and yuchao he, "Morphological Arrhythmia Automated Diagnosis Method Using Gray-Level Co-Occurrence Matrix Enhanced Convolutional Neural Network" in IEEE Access 2019.
27. Yang.J, Y. Bai, F. Lin, M. Liu, Z. Hou, and X. Liu, "A novel electrocardiogram arrhythmia classification method based on stacked sparse auto-encoders and softmax regression," Int. J. Mach. Learn. Cybern, vol. 9, no. 10, pp. 1733–1740, 2018.
28. Yildirim.O, P. Pławiak, R. S. Tan, and U. R. Acharya, "Arrhythmia detection using deep convolutional neural network with long duration ECG signals," Comput. Biol. Med., vol. 102, no. September, pp. 411– 420, 2018.
29. Yonghui Dai, Bo Xu, Siyu Yan, Jing Xu, "Study of cardiac arrhythmia classification based on convolution neural network", January 2020.

APPENDICES

```
import os
for dirname,_,filenames in
os.walk(r'C:\Users\RE_CS_2\Downloads\Input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.metrics import confusion_matrix
from keras.utils.np_utils import to_categorical
from sklearn.utils import class_weight
import warnings
warnings.filterwarnings('ignore')
train_df[187]=train_df[187].astype(int)
equilibre=train_df[187].value_counts()
print(equilibre)
plt.figure(figsize=(20,10))
my_circle=plt.Circle( (0,0), 0.7, color='white')
plt.pie(equilibre,labels=['n','q','v','s','f'],
colors=['red','green','blue','skyblue','orange'],autopct='%1.1f%%')
```

```

p=plt.gcf()
p.gca().add_artist(my_circle)
plt.show()

```

```

def plot_hist(class_number,size,min_,bins):
    img=train_df.loc[train_df[187]==class_number].values
    img=img[:,min_:size]
    img_flatten=img.flatten()

    final1=np.arange(min_,size)
    for i in range (img.shape[0]-1):
        tempo1=np.arange(min_,size)
        final1=np.concatenate((final1, tempo1), axis=None)
    print(len(final1))
    print(len(img_flatten))
    plt.hist2d(final1,img_flatten, bins=(bins,bins),cmap=plt.cm.jet)
    plt.show()

```

```

def network(X_train,y_train,X_test,y_test):

    im_shape=(X_train.shape[1],1)
    inputs_cnn=Input(shape=(im_shape), name='inputs_cnn')
    conv1_1=Convolution1D(64,      (6),      activation='relu',
input_shape=im_shape)(inputs_cnn)

```

```

conv1_1=BatchNormalization()(conv1_1)
pool1=MaxPool1D(pool_size=(3),                strides=(2),
padding="same")(conv1_1)
conv2_1=Convolution1D(64,      (3),      activation='relu',
input_shape=im_shape)(pool1)
conv2_1=BatchNormalization()(conv2_1)
pool2=MaxPool1D(pool_size=(2),                strides=(2),
padding="same")(conv2_1)
conv3_1=Convolution1D(64,      (3),      activation='relu',
input_shape=im_shape)(pool2)
conv3_1=BatchNormalization()(conv3_1)
pool3=MaxPool1D(pool_size=(2),                strides=(2),
padding="same")(conv3_1)
flatten=Flatten()(pool3)
dense_end1 = Dense(64, activation='relu')(flatten)
dense_end2 = Dense(32, activation='relu')(dense_end1)
main_output      =      Dense(5,      activation='softmax',
name='main_output')(dense_end2)

model = Model(inputs= inputs_cnn, outputs=main_output)
model.compile(optimizer='adam',
loss='categorical_crossentropy',metrics = ['accuracy'])

```

```
callbacks = [EarlyStopping(monitor='val_loss', patience=8),
              ModelCheckpoint(filepath='best_model.h5',
                              monitor='val_loss', save_best_only=True)]
```

```
history=model.fit(X_train,
y_train,epochs=2,callbacks=callbacks,
batch_size=32,validation_data=(X_test,y_test))
model.load_weights('best_model.h5')
return(model,history)
```

```
def evaluate_model(history,X_test,y_test,model):
    scores = model.evaluate((X_test),y_test, verbose=0)
    print("Accuracy: %.2f%%" % (scores[1]*100))
```

```
print(history)
fig1, ax_acc = plt.subplots()
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Model - Accuracy')
plt.legend(['Training', 'Validation'], loc='lower right')
plt.show()
```

```
fig2, ax_loss = plt.subplots()
```

```

plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Model- Loss')
plt.legend(['Training', 'Validation'], loc='upper right')
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.show()
target_names=['0','1','2','3','4']

for element in y_test:
    y_true.append(np.argmax(element))
prediction_proba=model.predict(X_test)
prediction=np.argmax(prediction_proba,axis=1)
cnf_matrix = confusion_matrix(y_true, prediction)
from keras.layers import Dense, Convolution1D, MaxPool1D,
Flatten, Dropout
from keras.layers import Input
from keras.models import Model
from keras.layers.normalization import BatchNormalization
import keras
from keras.callbacks import EarlyStopping, ModelCheckpoint

model,history=network(X_train,y_train,X_test,y_test)

```