DETECTION OF HEARTBEAT ANOMALY USING DEEP LEARNING

Project report submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

By

PARUL SHARMA (191206)

RIA MAHAJAN (191236)

UNDER THE SUPERVISION OF

Dr EKTA GANDOTRA

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat, 173234, Himachal Pradesh, INDIA

CERTIFICATE

Candidate's Declaration

I hereby declare that the work presented in this report entitled Detection of Heartbeat Anomaly using Deep Learning in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering submitted in the department of Computer Science and Engineering, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2022 to December 2022 under the supervision of Dr Ekta Gandotra (Assistant Professor, Department of Computer Science & Engineering and Information Technology).

I also authenticate that I have carried out the above-mentioned project work under the proficiency stream data science.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature)

Parul Sharma,191206

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)

Dr Ekta Gandotra

Assistant Professor (SG)

Department of Computer Science & Engineering and Information Technology

Dated:

CERTIFICATE

Candidate's Declaration

I hereby declare that the work presented in this report entitled Detection of Heartbeat Anomaly using Deep Learning in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering submitted in the department of Computer Science and Engineering, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2022 to December 2022 under the supervision of Dr Ekta Gandotra (Assistant Professor, Department of Computer Science & Engineering and Information Technology).

I also authenticate that I have carried out the above-mentioned project work under the proficiency stream data science.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature)

Ria Mahajan,191236

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)

Dr Ekta Gandotra

Assistant Professor (SG)

Department of Computer Science & Engineering and Information Technology

Dated:

ACKNOWLEDGEMENT

Firstly, we express our heartiest thanks and gratefulness to almighty God for his divine blessing to make it possible to complete the project work successfully. We are grateful and wish our profound indebtedness to Supervisor **Dr Ekta Gandotra**, **Assistant Professor (SG)**, Department of CSE Jaypee University of Information Technology, Wakhnaghat. The deep knowledge & keen interest of our supervisor in the field of machine learning helped us to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, and reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

We would like to express our heartiest gratitude to **Dr Ekta Gandotra, Department of CSE**, for her kind help to finish our project.

We would also generously welcome each one of those individuals who have helped us straightforwardly or in a roundabout way in making this project a win. In this unique situation, we want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated our undertaking.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

Parul Sharma (191206)

Ria Mahajan (191236)

TABLE OF CONTENT

Content **Page No.**

LIST OF ABBREVIATIONS

ABBREVIATIONS	DEFINITION
CNN	Convolutional Neural Network
CVD	Cardiovascular Disease
ECG	Electrocardiogram
AMI	Acute Myocardial Infarctions
CAD	Coronary Artery Disease
ML	Machine Learning
OS	Operating System
DNN	Deep Neural Networks
PCG	Phonocardiogram
GCG	Gyro cardiography
SCG	Seismocardiography
SVM	Support Vector Machines
GUI	Graphic User Interface
UI	User Interface
UX	User Experience
RAM	Random Access Memory
DFD	Data Flow Diagram
MFCC	Mel-frequency Cepstral Coefficients

LIST OF FIGURES

LIST OF GRAPHS

LIST OF TABLES

ABSTRACT

With 17.9 million deaths per year, cardiovascular diseases (CVDs) are the leading cause of death worldwide. Coronary heart disease, cerebrovascular disease, rheumatic heart disease, and other illnesses are among the classes of heart and blood vessel disorders known as CVDs. Heart attacks and strokes account for more than four out of every five CVD deaths, and one-third of these deaths happen before the age of 70 [1].

The best methods for monitoring someone's heart health currently are electrocardiograms (ECGs) and echocardiograms. However, both approaches are rather expensive. We aim to create a system that anyone with internet access can use and that is reliable, efficient, and economical.

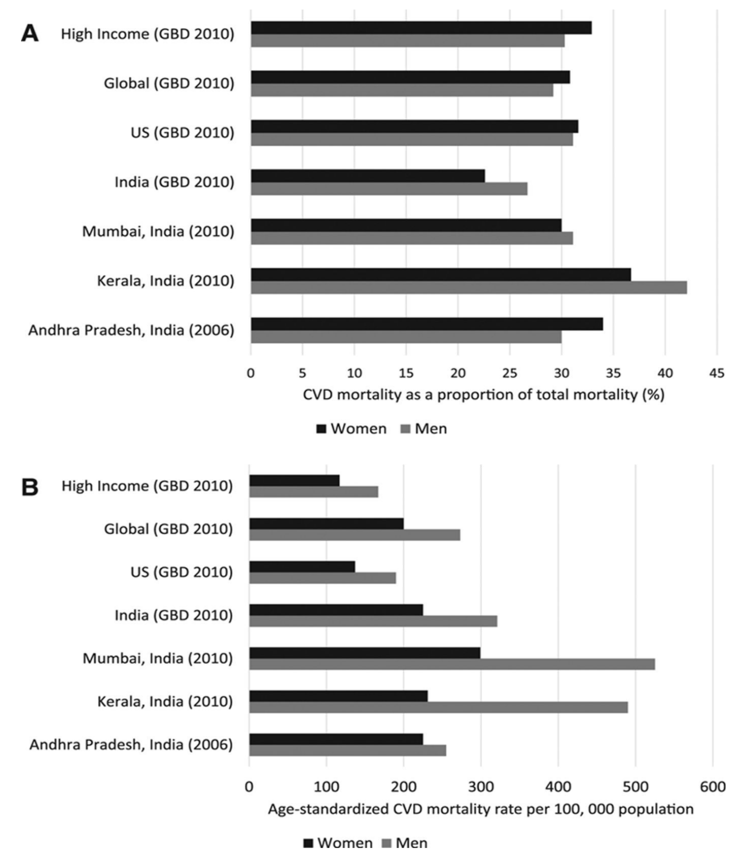
In this study, we use deep learning to identify any abnormalities in heartbeat sounds by including stethoscope sounds as well as waveforms captured using a mobile phone's microphone as input. We outline an automated heart sound categorization system that applies deep convolutional neural networks and time-frequency heat map representations (CNN). The classification model achieved an accuracy of 95.92% using the dataset from the Machine Learning challenge for the classification of heartbeat sounds [2]. Then we hyper-tuned the parameters to find and analyse the best accuracy possible

INTRODUCTION

According to the World Health Organization, India is responsible for one-fifth of these global deaths, particularly among the younger population. According to the findings of the Global Burden of Disease study, India has an age-standardised CVD death rate of 272 per 100,000 people, significantly higher than the global average of 235. Also it is seen that the Indians were affected by CVDs ten years earlier than people in the west [3].

It is suspected that the traditional risk factors, such as hypertension, diabetes mellitus, dyslipidemia, smoking, and obesity, contribute to the greater likelihood of CAD among Indians. Nine common risk variables (including physical inactivity, a low diet of fruits and vegetables, and chronic stress explained more than 90% of acute myocardial infarctions (AMIs) in South Asians, according to the INTERHEART study.

Disease at early age, quick progression, and high death rate are special sources of concern for us.The highest rates of coronary artery disease (CAD) are known to occur in Indians, yet the usual risk factors are unable to account for this elevated risk. For the Indian subcontinent, there are no systematic data gathering methods for cardiac mortality and morbidity, and the bulk of fatalities take place at home without the precise cause of death being known. Data on CV morbidity and mortality from hospitals might not be reflective of the full burden of CV disease.



Graph 1: Proportion of Cardiovascular Disease mortality in India

It is quite expensive to identify the early symptoms of heart problems. These examinations are not at all economical in underdeveloped and developing nations, as well as in nations with poor economic conditions. Therefore, a method that is both inexpensive and useful in identifying any early symptoms of a heart anomaly is urgently needed. Therefore, any strategy that might help eliminate this need will have a big influence on global health.

A stethoscope is a medical tool used for monitoring the body's internal sounds. It is a simple tool for hearing heartbeats. An electronic stethoscope has several advantages over an acoustic stethoscope, including greater sound production, a better frequency range, ambient noise loss, etc. It includes an amplifier that amplifies the heart's low-intensity sound. An electronic stethoscope can be a wireless machine or a recording device because it transmits sound electronically. Additionally, it may display the recorded heart sound visually.

Applications of machine learning to audio formerly relied on conventional methods of digital signal processing to extract features. For instance, phonetics principles could be used to analyse audio data and extract components like phonemes in order to understand human speech. For all of this to be accomplished and the system to be tuned for improved performance, a great deal of domain-specific expertise was needed.

Deep Learning has achieved great success in processing audio in recent years, though, as it becomes more and more prevalent. Deep learning eliminates the requirement for conventional audio processing methods, allowing us to rely on standard data preparation without having to create several manuals and unique feature sets.

Spectrograms are used in deep-learning audio applications to represent sounds. They often start with a wave file containing only raw audio data. Then transform the audio data into the spectrogram that corresponds to it. Use basic audio processing methods if you want to improve the spectrogram data. (The raw audio data can also be enhanced or cleaned before being converted to a spectrogram). We may analyse the image data using common CNN architectures to extract feature maps, which are an encoded representation of the spectrogram image, now that we have it.

Similarly in this work, we discuss an automated heart sound categorization technique that blends deep convolutional neural networks and time-frequency heat maps (CNN). The hyperparameters are then tuned.

1.1 Objective

Our study attempts to provide an innovative method for identifying anomalies in patient data's heart sounds and analysing the best possible hyperparameters to develop a feasible outcome.

1.2 Problem Statement

The public is alarmed about the high mortality rate from cardiovascular diseases. There is a need for an effective and simple system that can monitor the heartbeat, identify any abnormalities, and issue an early warning. In this approach, the general population will be assisted without having to pay exorbitant prices for various heart exams.

1.3 Methodology

The project's methodology is a straightforward five-phase design consisting of data collection, data preparation, feature extraction, model building and hyperparameter tuning and results in analysis.

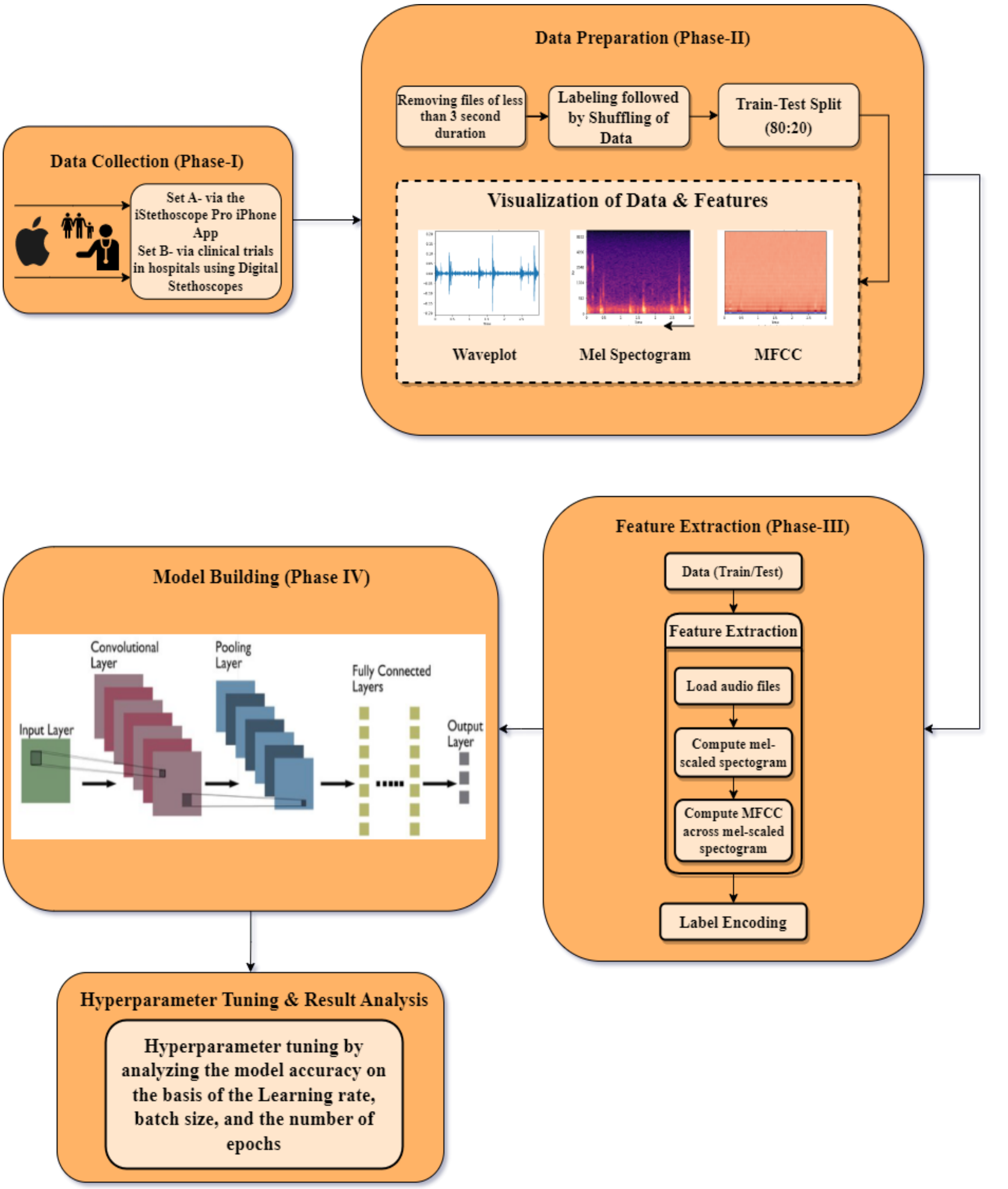


Figure 1: Project Workflow

1.4 Language Used

Python 3 - The main reasons for the use of this programming language for the implementation of the project are stated below:

Simple and dependable:

Python code is simplified and concise that enables programmers to create solid solutions because machine learning and artificial intelligence are based on complex algorithms and adaptable workflows. Developers can concentrate solely on resolving an ML issue rather than on the language's finer technical details.

Extensive selection of libraries and frameworks:

It's imperative to have a well-structured and well-tested environment to offer developers the best coding solutions possible.

A sizable number of libraries and frameworks are made available by Python.

Programmers use these frameworks and libraries to speed up development.

Machine learning frameworks include Keras, TensorFlow, and Scikit-learn.

- NumPy:Package used for scientific computing and data analysis.
- SciPy: Python package for advanced computation.
- Pandas is a tool for data analysis that has many applications.
- Librosa is a Python package for analysing music & audio signals.

Independency of platform

A programming language or framework that is "platform independent" enables programmers to build things on one system and then use them with little (or no) modification on another. The popularity of Python is due to the fact that it is a platform-neutral language. It can be used with a number of operating systems, including Windows, Linux, and macOS. Python software can be shared and used without the requirement for a Python interpreter due to the capacity of Python code to build standalone executable programmes for the majority of popular operating systems.

1.5 Organisation

The remainder of the report is organised as follows: The literature survey on the classification of anomalous heartbeat sounds is summarised in chapter 2. The methodology and system development is described in chapter 3. The experimental results and performance analysis based on several evaluation criteria are shown in chapter 4. The paper's conclusion and future scope are provided in chapter 5.

Chapter 02: LITERATURE SURVEY

Deep neural network architectures have been the subject of a recent study, leading to increasingly sophisticated and adaptive systems that can recognise and categorise objects in images. This subject has been part of ongoing research for the past ten years. For the purpose of detection, these researchers have developed a variety of algorithms and strategies.

While the automated analysis of image and text data has been a major focus of deep learning, improvements are frequently observed in fields that call for handling additional input modalities. One such field is the pharmaceutical industry, where physiologic time series data from a deep learning system may be used. Kaggle's 2014 and 2015 medical challenges, among others, have increased recently, leading to an advance in deep learning architecture. The 2016 challenge required the participants to do research on heartbeat sounds collected using digital stethoscopes. The major goal of this task was to determine whether the sound of the heartbeat was normal or abnormal.

Audio Signal Processing

Audio signal processing is the process of putting complex algorithms and methods to use on audio signals. Audio signals, which can be both digital and analog, are used to represent sound. Binary representations contain digital signals, while electrical signals contain analog signals. The time-frequency bands must be balanced and unwanted noise must be removed using this technique. It has a focus on computational methods for manipulating sounds. It minimises or eliminates undesired noise like echo and overmodulation by utilising a number of strategies.

B. Ming et al. [4] employed modelling and denoising audio signals in order to do spectrum analysis and apply filters to the audio signals. They did this by applying a wide range of core ideas from digital signal processing. The basic principles of digital signal science were used to process audio signals in this work, and signal extraction, amplitude-frequency transform, Fourier transform, filtering, and other techniques were extensively used to handle voice data.

S. Hershey et al. [5] used 30,871 video-level labels and several CNN architectures to classify the soundtracks of a collection of 70M training videos (5.24 million hours).

Fully-linked Deep Neural Networks (DNNs), ResNet, AlexNet, VGG, and Inception were all examined in the study.

F. Rong [6] proposed a revolutionary machine learning-based technique. The authors covered the three main kinds of audio data characteristics—Short time energy, Zero crossing rate, and Mel-Frequency cepstral coefficients—as well as the four-layer hierarchical structure of audio data. These features were then extracted to produce the feature vector. The research's final section described how to classify audio data using an SVM classifier with a Gaussian kernel.

Convolutional Neural Networks

Convolutional neural network (CNN) is a class of deep learning models that are primarily used to analyse visually descriptive data. Without human oversight, CNNs may extract significant information from visual input by using a variety of layers. The data is transformed in various ways by several levels. CNNs perceive data as spatial and can decompose an image's complexity for the computer to more easily comprehend and process, making them a popular tool for pattern identification. Convolutional layers and pooling layers are two examples of the layers that make up a traditional CNN. These layers are used to extract key features from input data. Additionally, it features certain layers at the very bottom that aid to identify the data by taking the output from the first two levels described. CNNs have a wide range of uses, including face detection and recognition, malware application categorization, X-ray picture classification, audio signal processing and others One of the most popular uses of CNN is image classification. The components of CNN, their functions, and all other significant problems were discussed in detail by S. Albawi et al [7]. They also listed the factors that affect CNN effectiveness. The absence of training samples with annotations is the fundamental obstacle to deep learning-based classification of medical images. W. Wang et al [8] showed how, for small training samples, fine-tuning greatly increased the classification accuracy of liver lesions. Based on the EEG spectrogram data, B. Mandhouj et al [9] created a deep convolutional neural network (CNN) model that was effective at identifying and categorising epilepsy seizures. The experimental findings demonstrated the effectiveness of the suggested strategy, which had an average accuracy rate of 98.22% in identifying EEG signals. The approach for classifying ECG arrhythmias proposed by J. Huang et al [10] used a two-dimensional (2D) deep convolutional neural network (CNN). The short-time Fourier transform was used to first convert the five different forms of ECG time domain data into time-frequency spectrograms. Finally, the ECG arrhythmia types were recognised and categorised using the spectrograms of the five different arrhythmia kinds as input to the 2D-CNN. A unique framework based on grid search optimization was suggested by S. Kaur et al [11] to develop an optimised deep learning model to predict the early onset of Parkinson's disease, where several hyperparameters had to be established and tweaked for evaluation of the deep learning model.

Heartbeat Anomaly Detection

Various techniques, such as segmentation, down-sampling, feature extraction, and classification, have been used in numerous research. To predict cardiac disease, these techniques are used. The ability to quickly solve a problem by recognizing an anomaly makes it both a challenge and a requirement to recognise unusual behaviours in a given situation.

B. Omarov et al[12] describe the use of machine learning (MO) techniques for phonocardiogram-based cardiovascular illness detection. The presentation of data on the high prevalence of heart disease among people sparked the creation of models to be used as additional tools for heart disease prediction. It focuses on the development of electronic stethoscopes, improved predictive models, and their implementation in clinical practice as key components supporting the idea that medical decision-making should be based on interdisciplinary collaboration between clinicians and information technology experts.

A. R. Jadhav1 et al. [13] explained an approach based on peak detection using Shannon energy envelope calculation and feature creation for Systoles (S1 or lub) and Diastoles (S2 or dub) segments of a cardiac sound. Next, neural networks are used to categorise these heartbeats. The usage of backpropagation neural networks with an adaptive learning technique significantly improves accuracy.

K.Wolk et al. [14] achieved more than 93% accuracy in the experiments due to the so-called multi-part interactive training method and an additional pre-trained ResNet network. This enables anyone to conduct preventative diagnosis using just a smartphone. Young doctors and students can learn a lot from it as well.

J.Rubin et al. [15] explained an automated heart sound categorization technique that integrates deep convolutional neural networks with time-frequency heat map representations (CNN). The trade-off between sensitivity and specificity is directly optimised during CNN architecture training using a modified loss function. The 2016 PhysioNet Computing in Cardiology challenge's goal was to accurately detect normal and pathological heart sounds from single, brief, possibly noisy recordings, and its method was tested against that goal.

We have selected and reviewed a few research papers. In order to compare the research papers, we have made a comparison table.

Author(s)	**Journal/ conference year**	**Published By (IEEE, Elsevier, Springer)**	**Methodology**	**Disadvantage**
Batyrkhan Omarov Khaled Gamry Aidar Batyrbekov [12]	2021	IEEE	The paper describes the use of machine learning (MO) techniques for phonocardiogram-based cardiovascular disease detection.	This method's drawback is that it only considers the time domain when estimating the energy of the signal. As a result, it performs noticeably worse when there is a lot of noise present.
KRZYSZTOF WOŁK AGNIESZKA WOŁK [14]	2019	IEEE	The suggested method involves using CNN to classify heartbeat sounds in order to perform a preliminary screening for cardiac rhythm abnormalities.	The amount of training and testing instances in the datasets used for these investigations was fairly small.

Matti Kaisti Mojtaba Jafari Tadi Olli Lahdenoja [16]	2018	IEEE	After using envelope and clustering techniques to detect beats, a multidimensional beat-to-beat detection algorithm finally merges the beat location.	The information gaps about the link between the GCG waveforms and cardiac motion are the most significant restrictions. standardisation requirements, motion noise susceptibility, and worse temporal accuracy of GCG peaks than in SCG
Jonathan Rubin Rui Abreu Anurag Ganguli [15]	2017	IEEE	In order to train the CNN model and forecast the output, the algorithm in this paper first converts the audio sounds to their corresponding heat maps.	The strategy used could only produce binary choice results, such as either normal or abnormal heart sounds. An architecture that took signal quality into account as output would probably enhance performance.
Ashwin R. Jadhav Arun G. Ghontale Anirudh Ganesh [13]	2017	IEEE	To extract the peaks, the algorithm first employs a Shannon energy envelope and peak detection techniques. A neural network is utilised to train and categorise the heartbeat sounds once the signal has been segmented and its features constructed.	Although the process of computing an envelogram for classification is not new, the information provided by the envelopes has never been sufficient for the classifiers to make a valid classification.

Samit Ari Koushik Hembram Goutam Saha[17]	2010	Elsevier	This technique was proposed to enhance the performance of the least square SVM classifier based on the Least Mean Square algorithm to identify heartbeat sound.	Since the 515 characteristics in this investigation were complicated and could not be linearly separated, the SVM classifier performed poorly. It also performed poorly when employing a polynomial kernel.

Table 1: Comparison of Related Works

Chapter 03: SYSTEM DEVELOPMENT

3.1 Feasibility Study

The purpose of the feasibility study is to assess if the project is feasible—that is, whether it can be executed effectively and conveniently—from an operational, technical, economic, and organisational perspective.

Economic Feasibility:

The project was created exclusively with open-source software and libraries. The creators of these libraries have all made them freely accessible online. As a result, there are no costs associated with project development.

Operational Feasibility:

The initiative is a feasible operating concept that could help all ordinary people. It can accomplish its goal of early detection of heart problems.

Technical Feasibility:

The heartbeat anomaly detection project was created using Google Colab, one of the most reliable development platforms, and features a very user-friendly user interface. Our project's technologies are practical.

Market Feasibility:

The project was created with the needs of the customers in mind. Clients favour news fast check specialists and digital forensic labs. It examines all market viability restrictions.

3.2 Technical Requirements

3.2.1 Software Requirements:

For the reasons described above, Python 3 was utilised to construct the project. The dataset was obtained using Kaggle. In the event of programming syntax mistakes, GitHub and StackOverflow were used as resources.

We used Google Collaboratory which is an open source and a free Jupyter notebook with a great GPU performance. It runs fully on cloud and needs no prior setup. We can develop and access advanced computational resources, write and run code, store and share analysis. Lines of code may be created and executed one at a time rather than writing and rewriting an entire program.

3.2.2 Hardware Requirements:

- **Processor:** Intel® Core™ i5-10300H
- **Installed memory (RAM):**8.00GB
- **System Type:** 64-bit Operating System

3.3 Model Development

On a broad scale, model development may be categorised into five discrete phases: model design, data engineering, model assembly, model validation, and model deployment.
The process of creating an ML model includes gathering data from a variety of reliable sources, processing it so that it is fit for modelling, selecting a modelling technique, building the model, computing performance measures, and selecting the best performing model. When a model is put into production, model upkeep is extremely important.

3.4 Use Case Diagram of the Major Project

The visual representation of how a user interacts with a user interface (UI), where the user inputs an audio file, runs detection on that file, and then receives output on the screen. Figure 2 displays a use-case diagram.

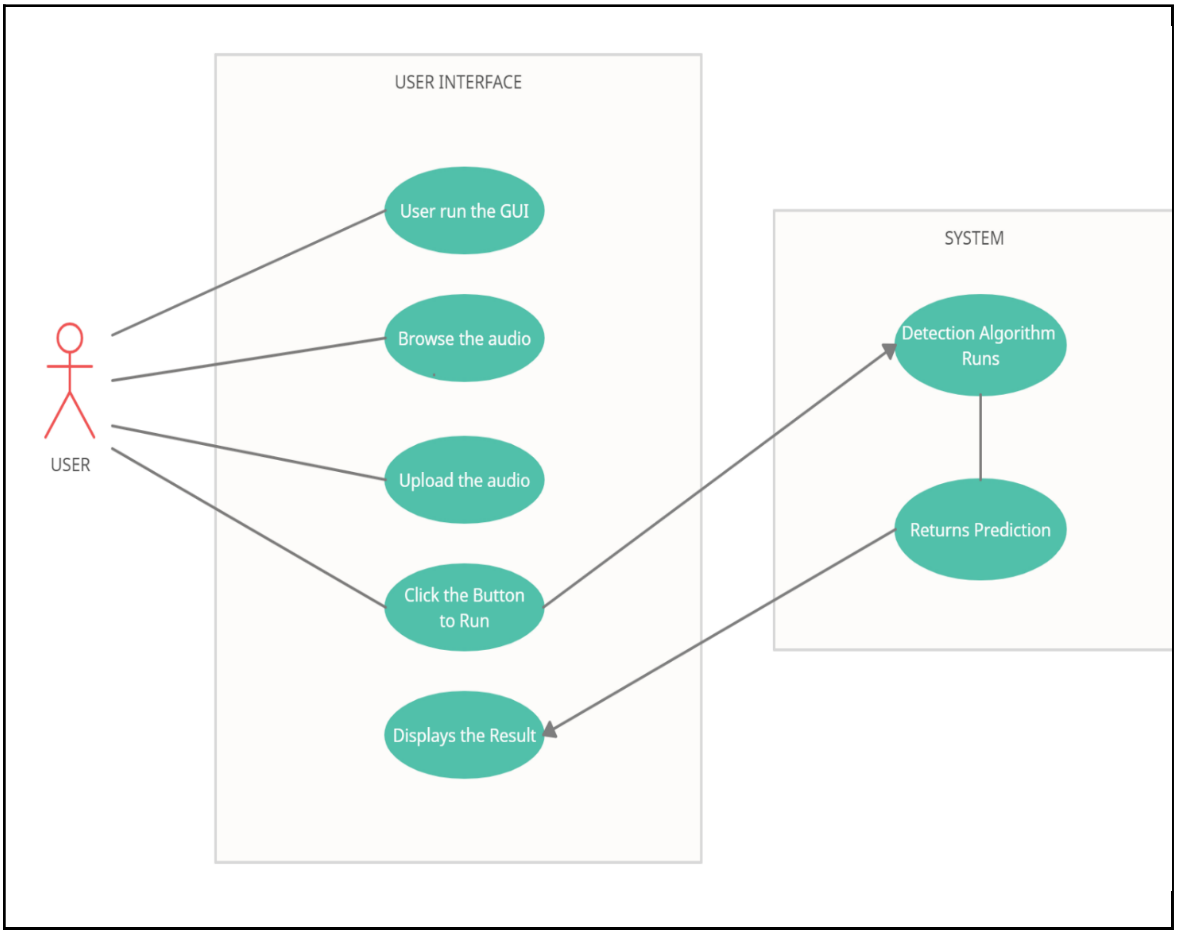


Figure 2: Use-case Diagram

3.5 DFD Diagram

The system's data flow is graphically depicted in the Data Flow Diagram (DFD) diagram. When a user inserts an audio file, data flows from the pre-processing stage to the trained CNN, which then predicts the outcome.

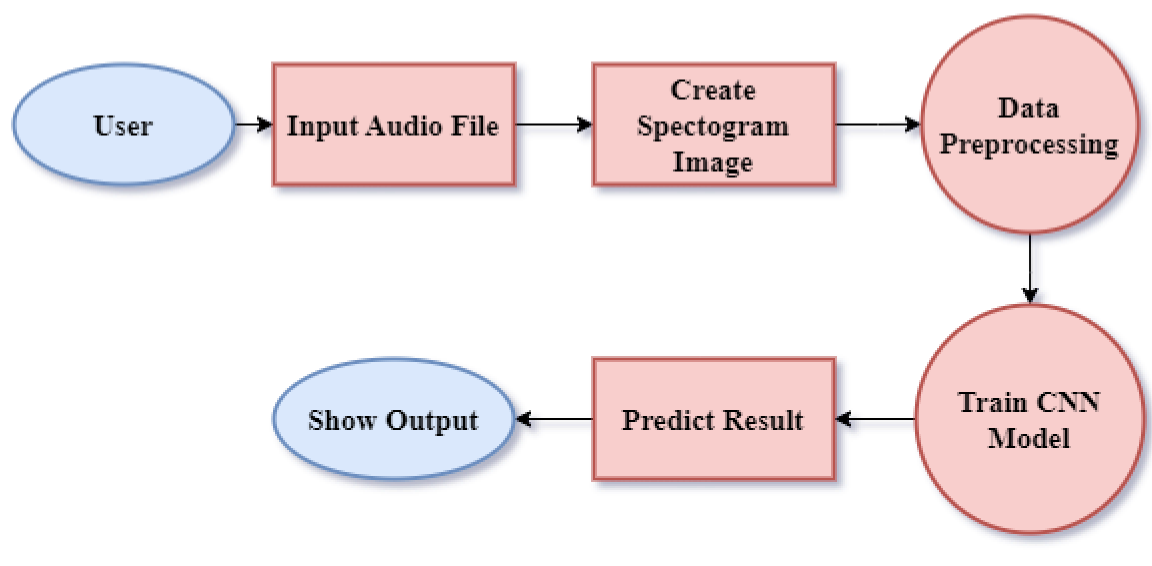


Figure 3: Data Flow Diagram

3.6 Data Set Used in the Major Project

This dataset was first used in a Machine Learning competition for the classification of heartbeat sounds, which was acquired through Kaggle. The dataset is separated into two sets based on the sources used to collect it. The I-Stethoscope Pro iPhone app was used to collect Set A data from the general public, and Set B data came from a clinical trial in hospitals using the DigiScope digital stethoscope. Four categories of heartbeat sounds exist

1. Normal i.e. the healthy heart sound.A typical heartbeat has the rhythm "lub dub, lub dub." The interval between "lub" and "dub" is less than the interval between "dub" and the next "lub".

2. Murmur: sounds that exist when there is turbulence in the blood flow. They typically produce a whooshing noise. Blood flowing through or close to the heart causes a sound known as a murmur. A murmur is typically caused by a high blood velocity.The murmur noises might come from either a lub or a dub, or from a dub and a lub.

3. Extrahls: heartbeats with an extra sound.

4. Extrasystoles: Unpleasant symptoms might result from additional heartbeats that happen outside of the natural heart rhythm.

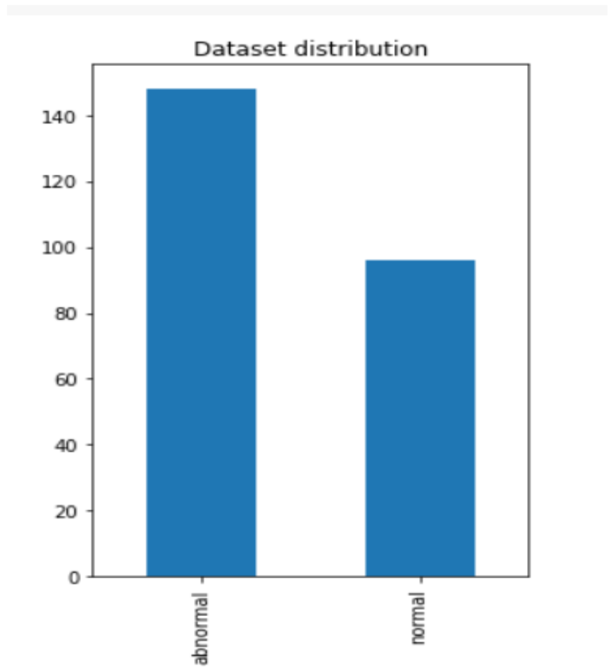
3.7 Data Set Features

3.7.1 Number of attributes, fields, description of data set

Set_A	Set_B
176	656

Table 2: Number of audios in dataset

Then after removing sound files of duration less than 3 seconds the dataset has two types of audios:



Graph 2: Dataset Distribution

3.8 Algorithm/Pseudo code of the model

In this project, we classify inputs as either normal or abnormal using the PCG waveform as an input to our deep convolutional neural network and the audio files as an input.

Pseudocode of the proposed method:

I. **Importing the dataset**
II. **Pre-processing the dataset:** Background noise is frequently present in heartbeat recordings made with stethoscopes and mobile phones. Due to insufficient data points, audio files with a duration of less than three seconds are removed from the analysis. The big audio files are divided into smaller ones while keeping the original label. The audio files have been altered to reduce the background noise brought on by the microphone's physical contact with the body for around half a second from both the front and the back.
III. **Feature Extraction:** The.wav type (Waveform Audio File Format) was used for the raw audio data, which was in the time domain's amplitude vs. time frame. This one-dimensional time-series signal was transformed into a two-dimensional heat map (mel-spectrograms & MCFF) that captures the signal's time-frequency pattern.
IV. **Dataset Split:** Training and test datasets have been created from the dataset. There are 195 images in the training dataset and 49 images in the test dataset. The graphics were created by converting audio data into the correct heat maps so that CNN architecture could use them as input.
V. **Training CNN:** The CNN architecture is built, and the training dataset is used to train the CNN.
VI. **Testing the model:** Using the test dataset, the saved model is now tested, with the results appropriately recorded.
VII. **Optimizer:**CNN model is trained using the Adam optimizer.
VIII. **Hyperparameter Tuning:** To accurately anticipate, we analyse various sets of hyperparameters. However, it is challenging to pick which hyperparameter to utilise due to the abundance of options.

To accurately anticipate, we analyse various sets of hyperparameters. However, it is challenging to pick which hyperparameter to utilise due to the abundance of options.

Learning Rate: The learning rate is one of the optimizer's hyperparameters. We will adjust the learning rate as well. The step size required for a model to achieve the minimal loss function is controlled by learning rate. Although the model learns more quickly with a higher learning rate, it may miss the minimum loss function and merely reach its surroundings. Obtaining a minimum loss function has a greater chance with a lower learning rate. Lower learning rates require larger epochs, or more time and memory resources, as a trade-off.

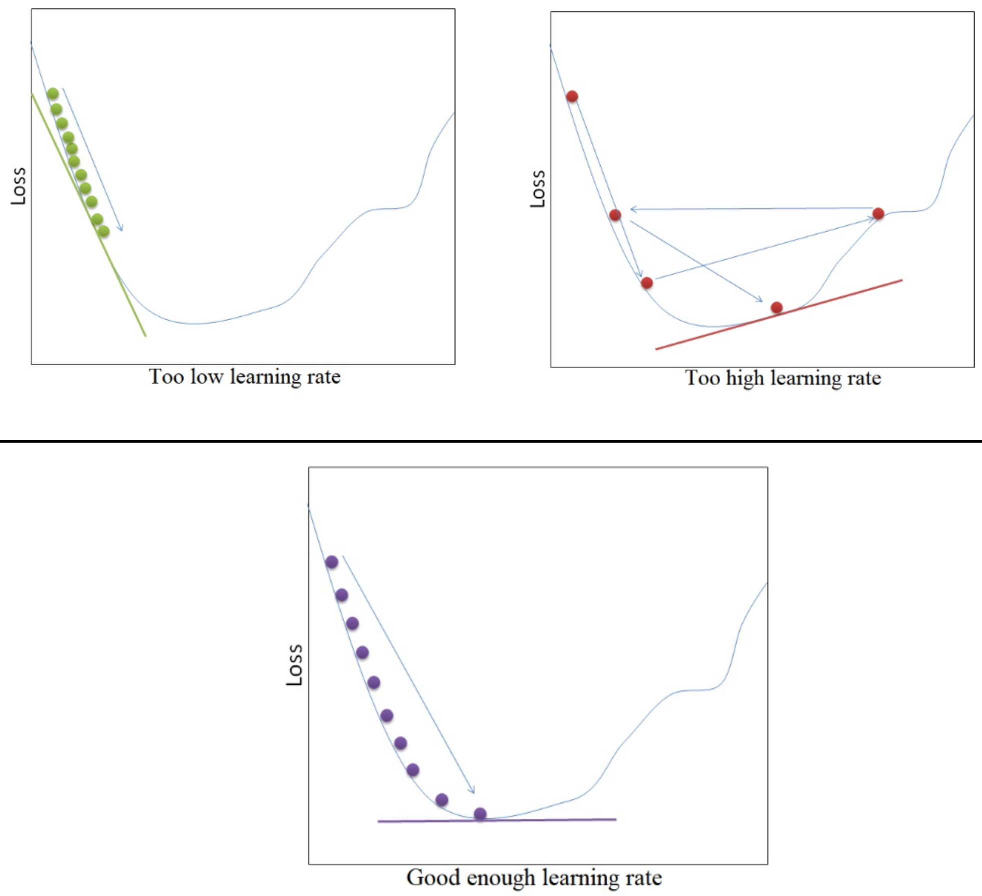


Figure 4: Learning Rate as Hyperparameter

Batch-size: The batch size is the number of input training data sub-samples. Although the learning process is accelerated by the reduced batch size, the variance of the validation dataset accuracy is increased. Although the learning process is slower with a larger batch size, the variance in the accuracy of the validation dataset is reduced.

Epochs: Underfitting is caused by a too-small number of epochs because the neural network has not learnt enough. Numerous iterations of the training dataset or epochs are necessary. On the other hand, too many epochs will result in overfitting, where the model can accurately forecast the data but is unable to adequately predict brand-new, unforeseen data. To achieve the best outcome, the number of epoch must be adjusted.

In the frequency domain, the following are used for feature extraction:

- Melspectrogram: The mel scale is a range of pitches that, to the human ear, appear to be equally distant from one another. The hertz (or just "mels") between mel scale values (or frequency) increases as frequency rises.The frequencies in a spectrogram are translated to the mel scale to create a mel spectrogram.
- MFCC (Mel Frequency Cepstral Components): As MFCCs collect information from audio files that more closely resemble how people perceive loudness and tone, we chose to use them to achieve this conversion. In speech recognition, MFCCs are frequently used as a characteristic type.

The primary distinction between the two approaches is the use of a quasi-logarithmic spaced frequency scale by MFCC as opposed to the spectrogram's linear spaced frequency scale (i.e., Short Time Fourier Transform (STFT)).

For the aforementioned, we utilise built-in librosa library functions in Python as follows:

- Using the built-in feature technique of librosa, the preprocessed sound files are converted into magnitude spectrograms and then mapped onto the mel-scale to obtain the mel-scale spectrogram.
- The Mel-Frequency Cepstral Coefficients (MFCC) are calculated by doing a cepstral analysis on the Mel-Spectrum using the log-power Melspectogram as an argument to the MFCC function.
- As a result, the audio file is now shown as a series of Cepstral vectors. The model is then given these Cepstral vectors for anomaly identification.

Below we have the visual representation of the features that were extracted from the audio files of both normal as well as abnormal heartbeat.

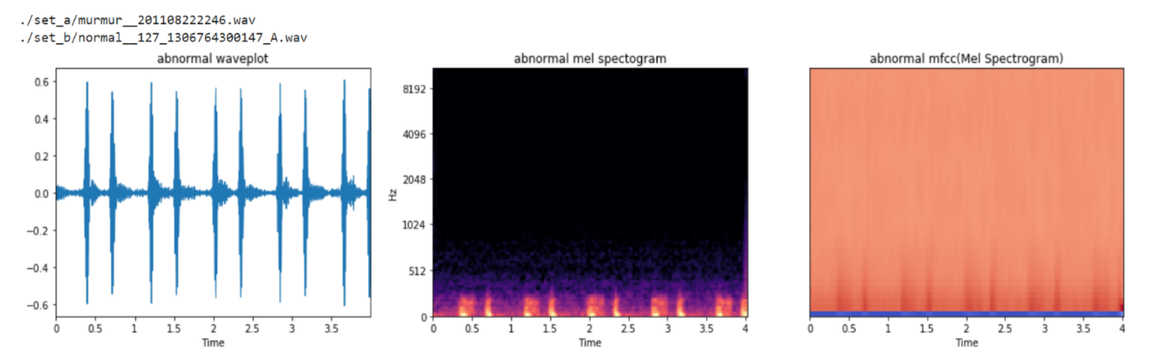


Figure 5: Abnormal Heartbeat features

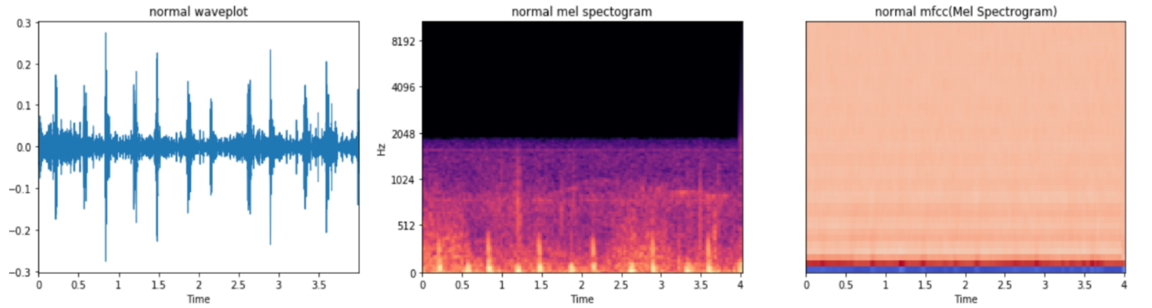


Figure 6: Normal Heartbeat Features

Below we define the architecture of CNN network.

CNN Architecture

Convolutional layers, fully linked layers, and a SoftMax classifier make up the core components of CNN.

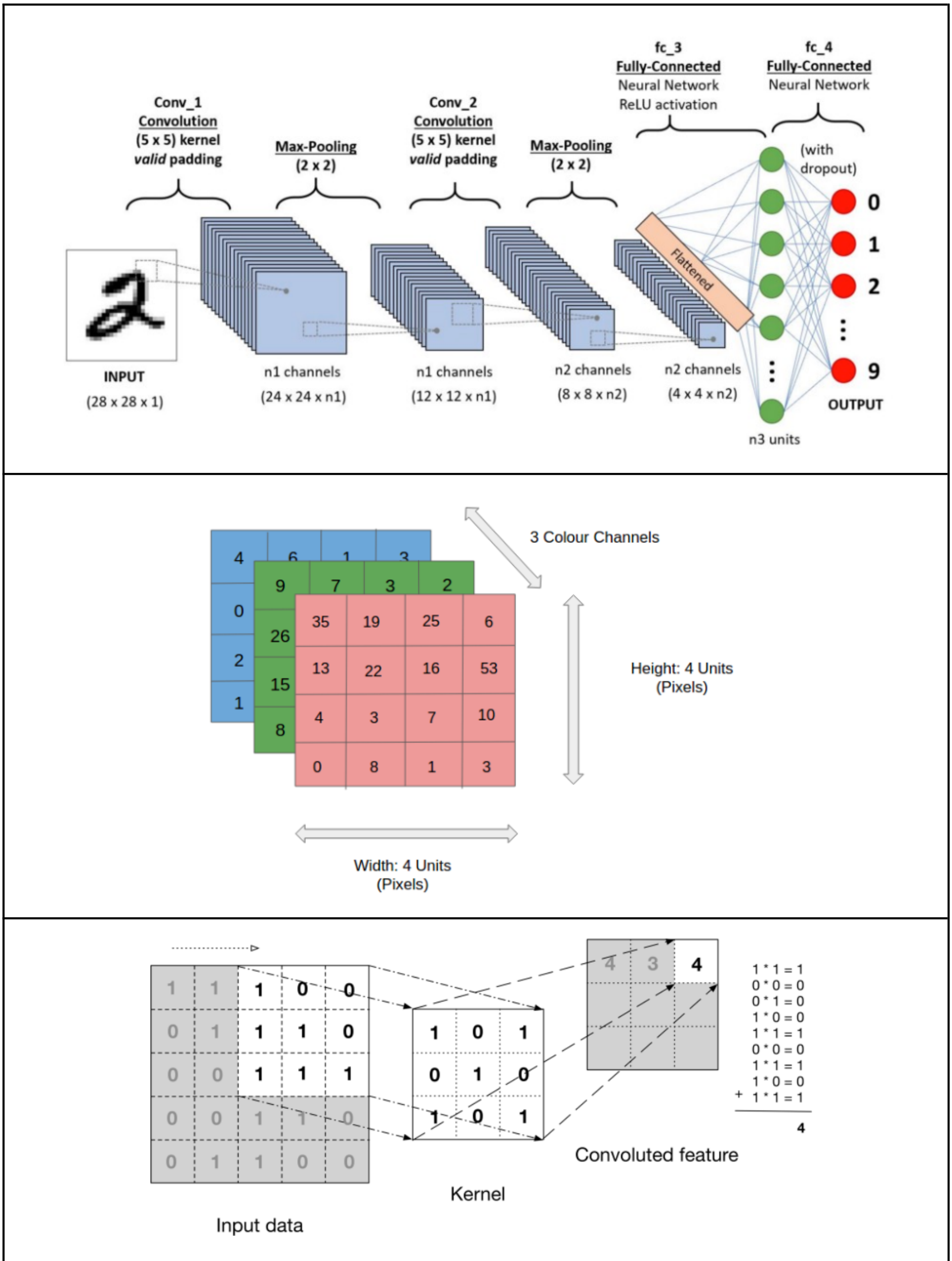


Figure 7: General Working Of CNN in Image Classification

The neural network gets a single channel of a 40 x 130 MFCC heat map as input, and it generates a binary classification that specifies whether the input file segment represents a normal or abnormal heart sound. The model uses one layer that uses the global average pool, four convolutional layers, four max-pooling layers, four dropout layers, and finally one dense layer. The activation function used in convolutional layers is called ReLU. ReLU offers the advantage of taking minimal time for both testing and training.

Function	*Explanations*
Convolution2D	Convolutional layer, sliding window convolution to 2-dimensional input information;
MaxPooling2D	Maximum pooling layer， imposing a maximum pooling on the spatial domain signal;
RELU	Rectified Linear Unit, which performs linear rectification activation on the input vector of the upper layer neural network and outputs nonlinear results.
Flatten	The Flatten layer is used to translate the multidimensional input information into one-dimensional information.
Dropout	It is an regularization layer to prevent overfitting;
Softmax	It is an activation function for multi-class neural network output.

Table 3: CNN layers & functions explained

43570 different parameters must be trained in total. The CNN architecture that has been developed for this project is summarised below.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d (Conv2D)              (None, 39, 129, 16)       80        
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 19, 64, 16)        0         
_________________________________________________________________
dropout (Dropout)            (None, 19, 64, 16)        0         
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 18, 63, 32)        2080      
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 9, 31, 32)         0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 9, 31, 32)         0         
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 8, 30, 64)         8256      
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 15, 64)         0         
_________________________________________________________________
dropout_2 (Dropout)          (None, 4, 15, 64)         0         
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 3, 14, 128)        32896     
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 1, 7, 128)         0         
_________________________________________________________________
dropout_3 (Dropout)          (None, 1, 7, 128)         0         
_________________________________________________________________
global_average_pooling2d (Gl (None, 128)               0         
_________________________________________________________________
dense (Dense)                (None, 2)                 258       
=================================================================
Total params: 43,570
Trainable params: 43,570
Non-trainable params: 0
_________________________________________________________________
```

Figure 8: CNN Parameters

3.9 Screen shots of the various stages of the Project

1. Libraries are imported.

```
import os
import glob
import numpy as np
from tqdm import tqdm
import itertools
import matplotlib.pyplot as plt
import pandas as pd

# Audio
import librosa
import librosa.display

# Scikit learn
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.utils import shuffle
from sklearn.utils import class_weight

# Keras
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D, Conv2D, MaxPooling2D, GlobalAveragePooling2D
from tensorflow.keras.utils import to_categorical

import warnings
warnings.filterwarnings('ignore', category=FutureWarning)
```

Figure 9: Libraries used

2. Loading Dataset and Applying Pre-processing

```python
dataset = []
for folder in ["/content/drive/MyDrive/Major Project /set_a.zip/**","/content/drive/MyDrive/Major Project /set_b.zip/**"]:
    for filename in glob.iglob(folder):
        if os.path.exists(filename):
            label = os.path.basename(filename).split("_")[0]
            duration = librosa.get_duration(filename=filename)
            # skip audio smaller than 3 secs
            if duration>=3:
                slice_size = 3
                iterations = int((duration-slice_size)/(slice_size-1))
                iterations += 1
                #initial_offset = (duration % slice_size)/2
                initial_offset = (duration - ((iterations*(slice_size-1))+1))/2
                if label not in ["Aunlabelledtest", "Bunlabelledtest", "artifact"]:
                    for i in range(iterations):
                        offset = initial_offset + i*(slice_size-1)
                        if (label == "normal"):
                            dataset.append({
                                "filename": filename,
                                "label": "normal",
                                "offset": offset
                            })
```

```python
                        else:
                            dataset.append({
                                "filename": filename,
                                "label": "abnormal",
                                "offset": offset
                            })

dataset = pd.DataFrame(dataset)
dataset = shuffle(dataset, random_state=45)
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 244 entries, 220 to 203
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   filename  244 non-null    object
 1   label     244 non-null    object
 2   offset    244 non-null    float64
dtypes: float64(1), object(2)
memory usage: 7.6+ KB
```

Figure 10: Pre-processing dataset

```python
plt.figure(figsize=(20,10))
idx = 0
for label in dataset.label.unique():
    y, sr = librosa.load(dataset[dataset.label==label].filename.iloc[33], duration=3)
    print(dataset[dataset.label==label].filename.iloc[33])

    # Wave plot
    idx+=1
    plt.subplot(2, 3, idx)
    plt.title("%s waveplot" % label)
    librosa.display.waveplot(y, sr=sr)

    # Mel Spectrogram
    idx+=1
    plt.subplot(2, 3, idx)
    S = librosa.feature.melspectrogram(y, sr=sr, n_fft=2048, hop_length=512, n_mels=128)
    S_DB = librosa.power_to_db(S, ref=np.max)
    librosa.display.specshow(S_DB, sr=sr, hop_length=512, x_axis='time', y_axis='mel')
    plt.title("%s mel spectogram" % label)

    # MFCC (Mel spectrogram)
    idx+=1
    mfccs = librosa.feature.mfcc(S=librosa.power_to_db(S), n_mfcc=40)
    plt.subplot(2, 3, idx)
    librosa.display.specshow(mfccs, x_axis='time')
    plt.title("%s mfcc(Mel Spectrogram)" % label)
plt.show()
```

Figure 11: Visualisation of Dataset

3. Feature Extraction

Converting Audio files to their respective heat maps.

```
] def extract_features(audio_path,offset):
#     y, sr = librosa.load(audio_path, duration=3)
    y, sr = librosa.load(audio_path, offset=offset, duration=3)
#     y = librosa.util.normalize(y)

    S = librosa.feature.melspectrogram(y, sr=sr, n_fft=2048,
                                        hop_length=512,
                                        n_mels=128)
    mfccs = librosa.feature.mfcc(S=librosa.power_to_db(S), n_mfcc=40)

#     mfccs = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=40)
    return mfccs
```

Figure 12: Feature Extraction

```
] # Encode Labels
encoder = LabelEncoder()
encoder.fit(train.label)

y_train = encoder.transform(train.label)
y_test = encoder.transform(test.label)

# Compute class weights
from sklearn.utils.class_weight import compute_class_weight
class_wts = compute_class_weight(class_weight = "balanced", classes= np.unique(y_train), y= y_train)
print(class_wts)
class_weights= {}
for index,value in enumerate(class_wts):
  class_weights[index]= value
print(class_weights)

[0.8125 1.3   ]
{0: 0.8125, 1: 1.3}
```

Figure 13: Encode Labels

4. Train/Test Split

```
x_train = x_train.reshape(x_train.shape[0], x_train.shape[1], x_train.shape[2], 1)
x_test = x_test.reshape(x_test.shape[0], x_test.shape[1], x_test.shape[2], 1)
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)

print("X train:", x_train.shape)
print("Y train:", y_train.shape)
print("X test:", x_test.shape)
print("Y test:", y_test.shape)
```

```
X train: (195, 40, 130, 1)
Y train: (195, 2)
X test: (49, 40, 130, 1)
Y test: (49, 2)
```

Figure 14: Splitting of Dataset

5. Training CNN.

5.1. CNN Layer

The CNN architecture has 4 convolutional layers and 4 Max pooling layers and a global pooling layer. Every convolutional layer uses the ReLU activation function and then a fully-connected layer with a SoftMax classifier.

```
model = Sequential()
model.add(Conv2D(filters=16, kernel_size=2, input_shape=(x_train.shape[1],x_train.shape[2],x_train.shape[3]), activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.2))

model.add(Conv2D(filters=32, kernel_size=2, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.2))

model.add(Conv2D(filters=64, kernel_size=2, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.2))

model.add(Conv2D(filters=128, kernel_size=2, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.5))
model.add(GlobalAveragePooling2D())

model.add(Dense(len(encoder.classes_), activation='softmax'))
model.summary()
```

Figure 15: Architecture of CNN model

6. Fitting the model

The model uses Adam optimizer and a learning rate of 0.01, a batch size of 16 and an epoch of 300.

```
adam = keras.optimizers.Adam(learning_rate=0.01)
model.compile(loss='categorical_crossentropy', metrics=['accuracy'], optimizer='adam')
```

```
%%time
history = model.fit(x=x_train, y=y_train,
                    validation_data=(x_test, y_test),
                    batch_size=16,
                    epochs=300,
                    class_weight=class_weights,
                    shuffle=True)
```

```
13/13 [==============================] - 1s 72ms/step - loss: 0.0274 - accuracy: 0.9897 - val_loss: 0.2121 - val_accuracy: 0.9184
Epoch 293/300
13/13 [==============================] - 1s 70ms/step - loss: 0.0504 - accuracy: 0.9795 - val_loss: 0.2146 - val_accuracy: 0.8776
Epoch 294/300
13/13 [==============================] - 1s 70ms/step - loss: 0.2077 - accuracy: 0.9487 - val_loss: 0.4212 - val_accuracy: 0.8367
Epoch 295/300
13/13 [==============================] - 1s 69ms/step - loss: 0.2584 - accuracy: 0.9077 - val_loss: 0.2305 - val_accuracy: 0.9184
Epoch 296/300
13/13 [==============================] - 1s 71ms/step - loss: 0.0911 - accuracy: 0.9641 - val_loss: 0.3627 - val_accuracy: 0.8367
Epoch 297/300
13/13 [==============================] - 1s 71ms/step - loss: 0.0960 - accuracy: 0.9641 - val_loss: 0.2309 - val_accuracy: 0.8571
Epoch 298/300
13/13 [==============================] - 1s 71ms/step - loss: 0.0969 - accuracy: 0.9641 - val_loss: 0.2151 - val_accuracy: 0.9184
Epoch 299/300
13/13 [==============================] - 1s 71ms/step - loss: 0.1352 - accuracy: 0.9333 - val_loss: 0.4630 - val_accuracy: 0.7551
Epoch 300/300
13/13 [==============================] - 1s 71ms/step - loss: 0.1524 - accuracy: 0.9385 - val_loss: 0.1514 - val_accuracy: 0.9592
CPU times: user 6min 56s, sys: 23.8 s, total: 7min 20s
Wall time: 5min 22s
```

Figure 16: Model Fitting

7. Trained CNN

The CNN trained is now saved as a .h5 file and then will be used to further test and also to predict some new data.

```
model_name = "heartbeat_classifier (normalised).h5"
model.save(model_name)
```

```
# # load and evaluate a saved model
from keras.models import load_model

# # load model
model = load_model("heartbeat_classifier (normalised).h5")

# File to be classified
classify_file = "my_heartbeat.wav"
x_test = []
x_test.append(extract_features(classify_file,0.5))
x_test = np.asarray(x_test)
x_test = x_test.reshape(x_test.shape[0], x_test.shape[1], x_test.shape[2], 1)
pred = model.predict(x_test,verbose=1)

print(pred)
```

Figure 17: Model saving & testing

Chapter 4 - Performance Analysis

4.1 Evaluation Metrics

We try to assess the classification performance of our model in this section using two metrics, i.e. accuracy and loss.

The ratio of correctly classified samples to all of the test samples is defined as accuracy. Mathematically it is defined as,

$$Accuracy(\%) = \frac{TP + TN}{TP + TN + FP + FN} \times 100$$

where TP stands for "true positive," indicating that the heartbeat sound falls in the normal category; FP stands for false positive, which denotes the abnormal heartbeat sounds categorised as normal; TN is for true negative, which denotes correct classification as abnormal; False negative, or FN, stands for mistaken classification as abnormal.

The difference between a single sample's true value and the model's predicted value is the metric of loss, according to its definition. There are various diverse sorts of mathematical expressions for this metric. We decided to use the categorical cross entropy loss function in this study.

$$loss = -\frac{1}{n}\sum_{i=1}^{n} \hat{y}_{i1} \ln y_{i1} + \hat{y}_{i2} \ln y_{i2} + \ldots + \hat{y}_{im} \ln y_{im},$$

where n stands for the sample size, m for the number of categories,

ŷ for the predicted output value, and y for the actual value.

4.2. Model Parameter Optimization

The learning rate, batch size and the number of epochs are the three key variables in this suggested 2D-CNN model. The phase of model parameter optimization is necessary to attain the optimum classification performance of PCG heartbeat signals.

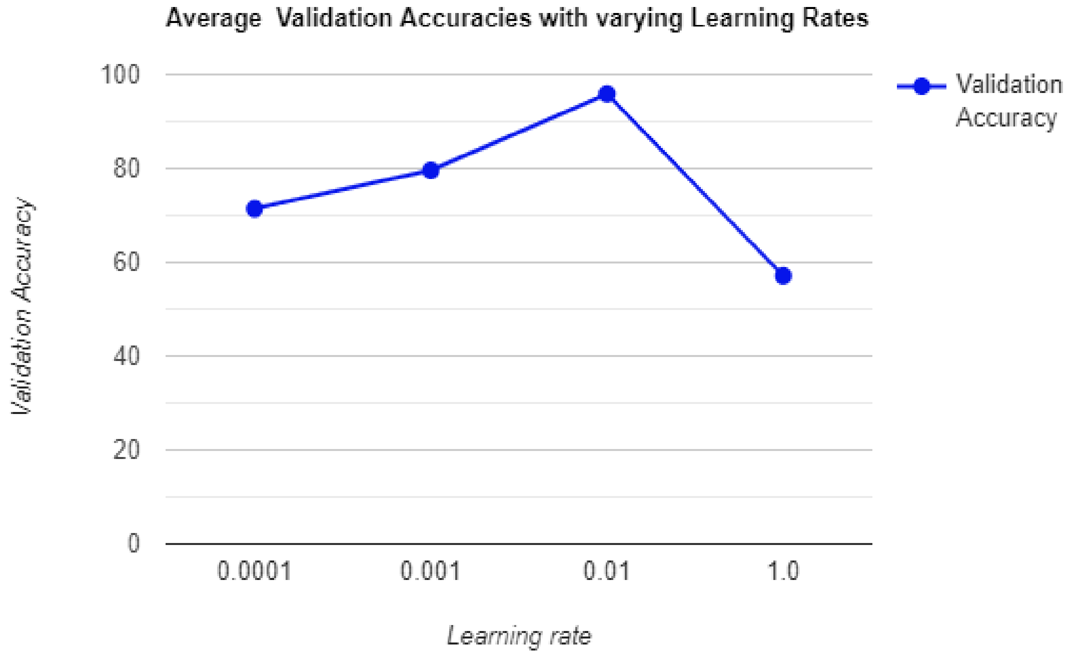
A series of contrast experiments using various parameter configurations were carried out to assess the significance of the learning rate and batch size within the proposed 2D-CNN model. On the one hand, while maintaining the value of the batch size, we tested the contrast experiments using various learning rates. In Table 4, the comprehensive parameter set is displayed. Contrarily, we tried the contrast tests with various batch sizes while maintaining the same learning rate value. In Table 5, the comprehensive parameter sets are displayed.

Further, on determination of the optimum learning rate and batch size, the model accuracy was then comprehended with the gradual increase in the number of epochs during model training. In Table 6, the corresponding results are displayed.

Batch size	Epochs	Learning Rate	Validation Accuracy
128	300	0.0001	0.71428
128	300	0.001	0.79591
128	300	0.01	0.95918
128	300	1.0	0.57142

Table 4: Validation accuracy with varying learning rate

Table 4 displays the variation in the model's validation accuracy while keeping the batch size constant (batch size=128), for 300 epochs and varying the learning rate from 0.0001 to 1.0. It is evident that when we increase the learning rate, the validation accuracy rises. The validation accuracy reaches its maximum, or 95.91%, at learning rate=0.01, which is a widely used number. Following this, the validation accuracy drops dramatically to just 57.14% at learning rate=1.0, making learning rate=0.01 the model's most optimal setting. This variance in the model's validation accuracy with the changing learning rate is depicted visually in Graph 3.

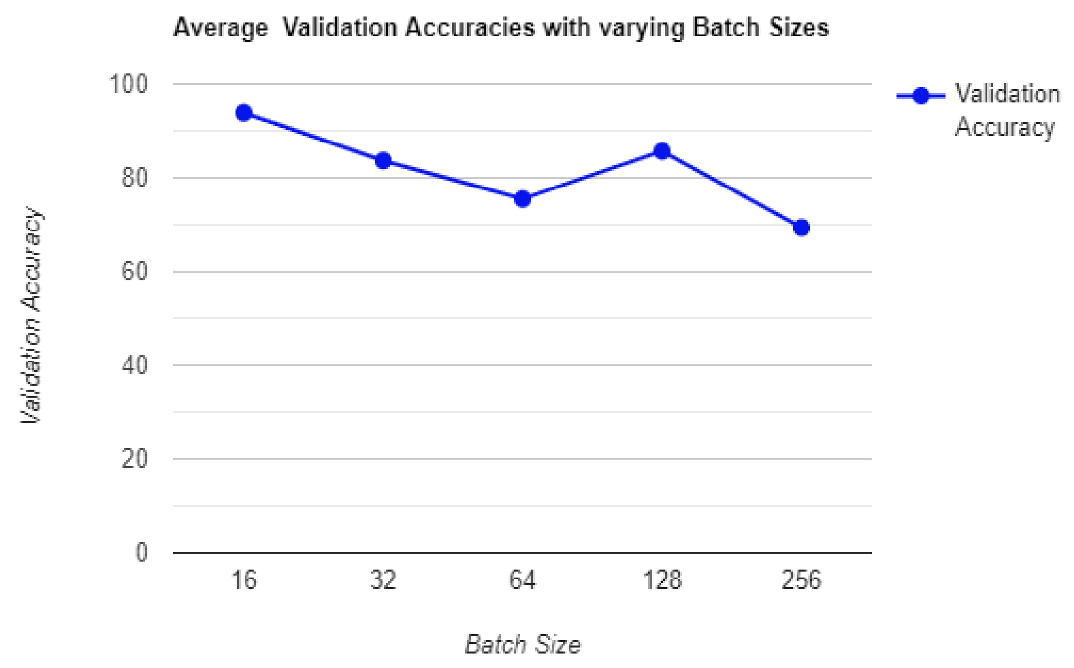


Graph 3: Average validation accuracy with varying learning rates

Batch size	Epochs	Learning Rate	Validation Accuracy
16	300	0.01	0.93877
32	300	0.01	0.83673
64	300	0.01	0.75510
128	300	0.01	0.85714
256	300	0.01	0.69387

Table 5: Validation accuracy with varying batch size

We also tried adjusting the batch size after figuring out the model's optimal learning rate. In order to do this, the model was trained with the parameters epochs = 300, learning rate = 0, and batch size varying between 16–256. As can be seen, the model's validation accuracy was at its highest, or 93.87%, at batch size=16. The validation accuracy then gradually declined till batch size = 64. The validation accuracy suddenly increased at batch size=128, but it severely declined to only 69.38% at batch size=256. As a result of the aforementioned studies, it was determined that batch size=16 was the ideal batch size for the model. Table 5 displays the variation in validation accuracy as batch size increases, whereas Graph 4 illustrates the same information graphically.

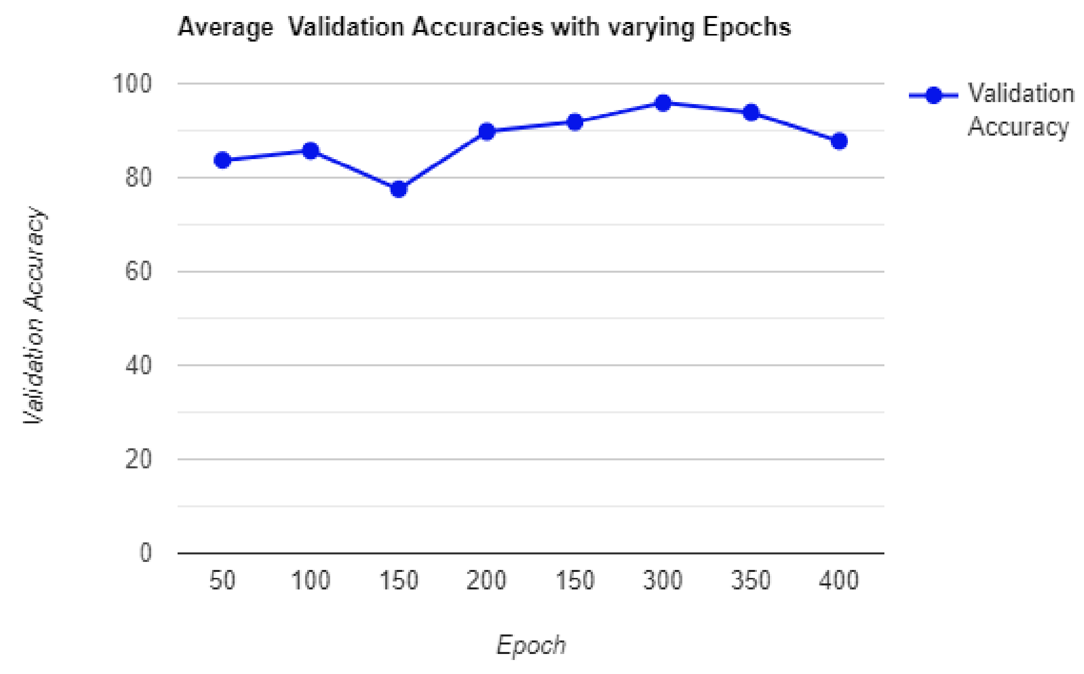


Graph 4: Average validation accuracy with varying batch size

We experimented with the model by changing the number of epochs after figuring out the appropriate learning rate and batch size for our model. We changed the epoch settings between 50 and 400 while keeping the batch size=16 and learning rate=0.01. The model showed its best performance at 300 epochs, yielding an accuracy of 95.92%. The fluctuation in model accuracy with changing epoch counts is shown in Table 6, and the same is visually shown in Graph 5.

Batch size	Epochs	Learning Rate	Validation Accuracy
16	50	0.01	0.83691
16	100	0.01	0.85714
16	150	0.01	0.77551
16	200	0.01	0.89795
16	250	0.01	0.91836
16	300	0.01	0.95918
16	350	0.01	0.93877
16	400	0.01	0.8776

Table 6: Validation accuracy with varying epochs



Graph 5: Average validation accuracy with varying epochs

The hyperparameter tuning performed above gave us the optimal parameter values to be as follows:

Parameters	Values
Learning rate	0.01
Optimiser	Adam
Loss	Categorical Crossentropy
batch_size	16
Epochs	300
Metrics	Accuracy

Table 7: Parameters of final CNN Model

With a loss of 15.14 percent, the aforementioned model produced a validation accuracy of 95.92%. The final model's accuracy and loss curves are displayed in the figures below.

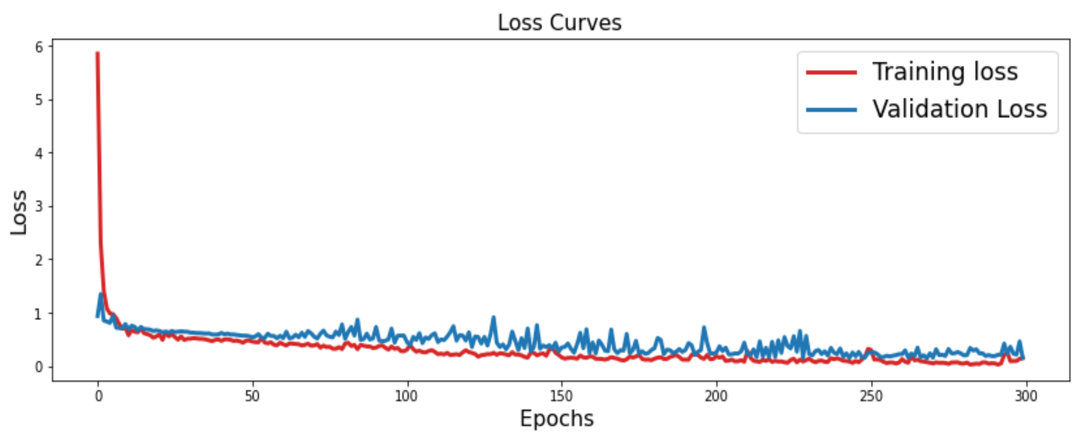


Figure 18 : Loss Curves for CNN Model

The loss gradually decreases from the first epoch until it stabilises in the vicinity of the last few. The graph which was plotted made it quite clear to see.

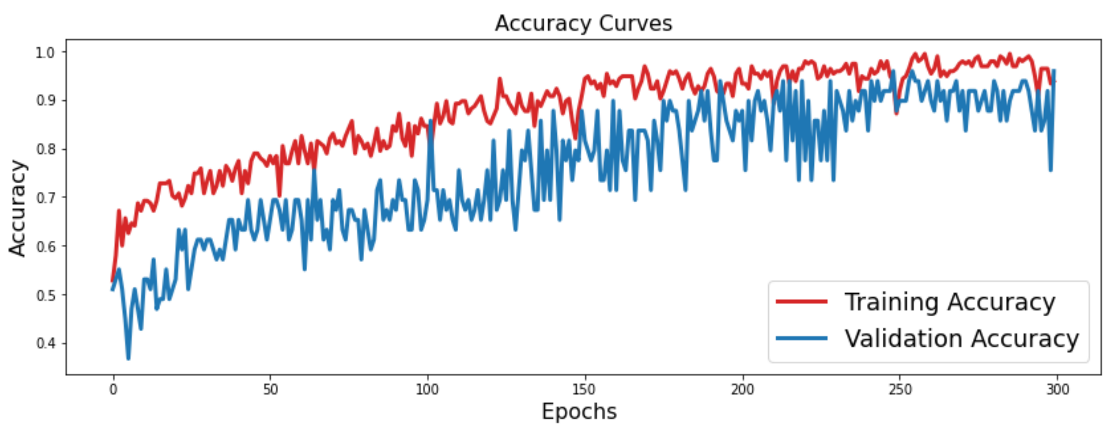


Figure 19: Accuracy Curves for CNN Model

From about 50% during the initial epochs, accuracy starts to rise. It maintains growing at a steady rate until it stabilises near the end epochs, or around 90-95%. The graph in figure, which was plotted, made it quite clear to see.

The testing of the developed model using a test heartbeat sound is shown in the ensuing graphics. The sample sound "my heartbeat.wav" is classified using the saved model "heartbeat classifier (normalised).h5". On prediction, the model provides a confidence of 99.82%.

```
# # load and evaluate a saved model
from keras.models import load_model

# # load model
model = load_model("heartbeat_classifier (normalised).h5")

# File to be classified
classify_file = "my_heartbeat.wav"
x_test = []
x_test.append(extract_features(classify_file,0.5))
x_test = np.asarray(x_test)
x_test = x_test.reshape(x_test.shape[0], x_test.shape[1], x_test.shape[2], 1)
pred = model.predict(x_test,verbose=1)

print(pred)
```

```
1/1 [==============================] - 0s 106ms/step
[[0.00172758 0.9982724 ]]
```

```
#@title
pred_class=model.predict(x_test)
classes_x=np.argmax(pred_class,axis=1)
if classes_x[0]:
    print("Normal heartbeat")
    print("confidence:",pred[0][1])
else:
    print("Abnormal heartbeat")
    print("confidence:",pred[0][0])
```

```
1/1 [==============================] - 0s 23ms/step
Normal heartbeat
confidence: 0.9982724
```

Figure 20: Testing of CNN Model on New Data

Chapter 05: Conclusions

The study presented an automated heart sound classification algorithm that blended deep convolutional neural networks and time-frequency heat maps (CNN). Testing and training accuracy for the proposed model were 95.92% and 93.85%, respectively. When the suggested model was evaluated on individual input data, it also produced optimal results with high confidence scores (99.82%) and accurate classification.

5.2 Application of the Major Project

A good, dependable system that can quickly identify abnormal heartbeats is required. Heart failure is among the leading causes of death in the modern world.

Any anomaly in the heartbeat sound can be recognised by our model even if the audio was captured using a mobile phone.

Due to its straightforward GUI design, which covers a variety of people, it might be used by any medical experts as well as by non-professionals.

5.3 Limitation of the Major Project

Even though our model had a high accuracy rate, some incorrect classifications could still be generated.

If the dataset had been larger, the accuracy of the model might have been increased even more because CNN works well with huge datasets because it can learn from practically all case scenarios.

The model can only indicate whether there is a heartbeat issue, thus precise detection would necessitate testing carried out by a medical practitioner.

5.4 Future Work

Future work will involve gathering more diverse datasets for the model's training. Additionally, because the output of our model was binary, i.e., normal or abnormal, future work could eliminate this restriction, offer more precise explanations, correctly identify the heartbeat sound, and provide more accurate findings. Additionally, we can take into account signal quality as an input to increase the performance of our architecture and the dependability of our system.

REFERENCES

[1] WHO . World Health Organization; Geneva: 2016. Global Health Estimates 2015: Deaths by Cause, Age, Sex, by Country and by Region, 2000–2015. [Google Scholar]

[2] P. Bentley, G. Nordehn, M. Coimbra and S. Mannor, *The PASCAL Classifying Heart Sounds Challenge*, 2011, [online] Available: http://www.peterjbentley.com/heartchallenge/index.html.

[3] Prabhakaran D., Jeemon P., Roy A. Cardiovascular diseases in India. *Circulation.* 2016;133:1605–1620. [PubMed] [Google Scholar]

[4] B. Ming and P. Wu, "Research on Audio Signal Denoising and Simulation Processing," 2019 International Conference on Communications, Information System and Computer Engineering (CISCE), 2019, pp. 192-194, doi: 10.1109/CISCE.2019.00051.

[5] S. Hershey et al., "CNN architectures for large-scale audio classification," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 131-135, doi: 10.1109/ICASSP.2017.7952132.

[6] F. Rong, "Audio Classification Method Based on Machine Learning," 2016 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), 2016, pp. 81-84, doi: 10.1109/ICITBS.2016.98.

[7] S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.

[8] Wang, W. *et al.* (2020). Medical Image Classification Using Deep Learning. In: Chen, YW., Jain, L. (eds) Deep Learning in Healthcare. Intelligent Systems Reference Library, vol 171. Springer, Cham. https://doi.org/10.1007/978-3-030-32606-7_3

[9] Mandhouj, B., Cherni, M.A. & Sayadi, M. An automated classification of EEG signals based on spectrogram and CNN for epilepsy diagnosis. *Analog Integr Circ Sig*

Process 108, 101–110 (2021). https://doi.org/10.1007/s10470-021-01805-2

[10] J. Huang, B. Chen, B. Yao and W. He, "ECG Arrhythmia Classification Using STFT-Based Spectrogram and Convolutional Neural Network," in IEEE Access, vol. 7, pp. 92871-92880, 2019, doi: 10.1109/ACCESS.2019.2928017.

[11] Kaur, S., Aggarwal, H. & Rani, R. Hyper-parameter optimization of deep learning model for prediction of Parkinson's disease. *Machine Vision and Applications* 31, 32 (2020). https://doi.org/10.1007/s00138-020-01078-1

[12] Omarov, B., Gamry, K., Batyrbekov, A., Alimzhanova, Z., Dauytova, Z., & Seitbekova, G. (2021, January). Detection of heartbeat abnormalities from phonocardiography using machine learning. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (pp. 843-847). IEEE.

[13] A. R. Jadhav, A. G. Ghontale and A. Ganesh, "Heart sounds segmentation and classification using adaptive learning neural networks," 2017 International Conference on Signal Processing and Communication (ICSPC), 2017, pp. 33-38, doi: 10.1109/CSPC.2017.8305881.

[14] K. Wołk and A. Wołk, "Early and Remote Detection of Possible Heartbeat Problems With Convolutional Neural Networks and Multipart Interactive Training," in IEEE Access, vol. 7, pp. 145921-145927, 2019, doi: 10.1109/ACCESS.2019.2919485.

[15] Jonathan Rubin, et al. 2017 Recognizing Abnormal Heart Sounds Using Deep Learning arXiv:1707.04642v2

[16] M. Kaisti et al., "Stand-Alone Heartbeat Detection in Multidimensional Mechanocardiograms," in IEEE Sensors Journal, vol. 19, no. 1, pp. 234-242, 1 Jan.1, 2019, doi: 10.1109/JSEN.2018.2874706.

[17] Ari, S., Hembram, K., & Saha, G. (2010). Detection of cardiac abnormality from PCG signal using LMS based least square SVM classifier. *Expert Systems with Applications*, *37*(12), 8019-8026. https://doi.org/10.1016/j.eswa.2010.05.088

[18] Colony, B., Maabreh, M., AlFuqaha, A.: Parameters optimization of deep learning models using Particle swarm optimization. In: International Conference of Wireless Communications, and Mobile Computing (2017)

[19] Bind, S., Tiwari, A., Sahani, A.: A survey of machine learning-based approaches for Parkinson disease prediction. Int. J. Inf. Technol. Comput. Sci. 6, 1648–1655 (2015)

[20] Tahmassebi, A., Gandomi, A., Fong, S.: Multi-stage optimization of a deep model: a case study on ground motion modeling. Plos One. 3, 1–23 (2018)

Appendix A: Glossary

Algorithm:

An algorithm is a process used to carry out a computation or solve a problem. In either hardware-based or software-based routines, algorithms function as a detailed sequence of instructions that carry out predetermined operations sequentially.

UI:

The point of human-computer contact and communication in a device is the user interface. This can include desktop displays, keyboards, mouse, and other pointing devices.

GUI:

A set of interactive visual elements for computer software is known as a "graphical user interface." A GUI presents information-conveying and action-representative items for the user to interact with. When the user interacts with the items, they alter their colour, size, or visibility.

IDE:

Integrated Development Environment (IDE) software combines standard developer tools into a single graphical user interface and is used to create programmes (GUI).

Deep Learning:

It is a form of machine learning that uses numerous layers of processing to extract increasingly more complex properties from data. It is based on artificial neural networks.

Neural Network:

These are also known as artificial neural networks (ANNs) or simulated neural networks (SNNs). They are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

Heatmap:

A heatmap is a type of graph where values are represented by colours.