

**HEARTBEAT ANOMALY DETECTION
FROM AUDIO FILES**

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

in

Computer Science and Engineering

By

PANKIL PATHANIA (181351)

UNDER THE SUPERVISION OF

Dr. RAJINDER SANDHU



Department of Computer Science & Engineering and Information
Technology

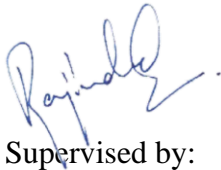
**Jaypee University of Information Technology, Wagnaghat,
173234, Himachal Pradesh, INDIA**

TABLE OF CONTENT

Content	Page No.
Declaration by Candidate	I
Certificate by Supervisor	II
Acknowledgment	III
Abstract	IV
1. Chapter No. 1: INTRODUCTION	1
1.1 Introduction	1
1.2 Objective	2
1.3 Problem Statement	2
1.4 Methodology	2
1.5 Language Used	2
2. Chapter No. 2: Literature Survey	3
3. Chapter No. 3: System Development	5
3.1 Feasibility Study	5
3.2 Requirements on Major Project	6
3.3 Use Case Diagram of Major Project	7
3.4 DFD Diagram	8
3.5 Dataset Used in Major Project	9
3.6 Dataset Features	9
3.7. Algorithm/Pseudo code of model	9
3.8 Screenshots of the various stages of the Project	14
4. Chapter No. 4: Performance Analysis	22
4.1 Predicting the result	22
4.2 Model Testing	22
4.3 Discussion on the result Achieved	23
4.4 Comparison	24
5. Chapter No. 5: Conclusion	26
5.1 Applications of the Project	26
5.2 Limitations of the Project	26
5.3 Future Work	26
REFERENCE	27

DECLARATION

I hereby declare that, this project has been done by me under the supervision of (**Dr. Rajinder Sandhu, Affiliation**), Jaypee University of Information Technology. I also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.



Supervised by:

Dr. Rajinder Sandhu


Assistant Professor (SG)

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology

Submitted by:

Pankil Pathania



(181351)

Computer Science & Engineering Department

Jaypee University of Information Technology

CERTIFICATE

This is to certify that the work which is being presented in the project report titled **Heartbeat Anomaly Detection from Audio Files** in partial fulfilment of the requirements for the award of the degree of B.Tech in Computer Science And Engineering and submitted to the Department of Computer Science And Engineering, Jaypee University of Information Technology, Waknaghat is an authentic record of work carried out by **Pankil Pathania (181351)** during the period from January 2022 to May 2022 under the supervision of **Dr. Rajinder Sandhu** Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat.



Student Name: Pankil Pathania

Roll number: 181351

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




Supervisor Name: Dr. Rajinder Sandhu

Assistant Professor (SG)

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology



Supervisor Name: Darshan Sai

DC Delivery Manager

26-05-2022

AKNOWLEDGEMENT

Firstly, I express my heartiest thanks and gratefulness to almighty God for his divine blessing makes us possible to complete the project work successfully. I really grateful and wish my profound my indebtedness to Supervisor **Dr. Rajinder Sandhu, Assistant Professor (SG)**, Department of CSE Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of my supervisor in the field of “**Heartbeat Anomaly Detection**” to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stage have made it possible to complete this project.

I would like to express my heartiest gratitude to **Dr. Rajinder Sandhu, Department of CSE**, for his kind help to finish my project.

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

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Pankil Pathania

ABSTRACT

Many people die each year due to cardiovascular diseases as compared to any other disease as reported by WHO in 2017. Heart disease has become the leading cause of death all over the world. As rural primary care is handled by unqualified practitioners due to lack of doctors the fatality rate of the rural area has exceeded those of urban areas.

Electrocardiogram (ECG) and echocardiogram are currently the most effective ways to monitor one's heart condition. However, both methods are relatively expensive. Our aim is to develop a reliable, fast and affordable system that can be used by anyone with internet access.

In this work, we apply deep learning to recognize any abnormality in heartbeat sound by taking in input in stethoscope sounds and even waveforms recorded using the microphone of a mobile phone. We describe an automated heart sound classification algorithm that combines the use of time-frequency heat map representations with a deep convolutional neural network (CNN).

Chapter 01: INTRODUCTION

1.1 Introduction

Every year around 17.9 million people die due to cardiovascular conditions. According to a report of WHO, these deaths could have been prevented if the diseases were diagnosed at an early stage and even the disease can be cured. Approximately 610,000 people die of heart disease in the United States each year– that's 1 in every 4 deaths. More than 29% of the entire deaths in 2004 are due to cardiovascular conditions and the number is rising each day. Coronary heart disease (CHD) is the most typical type of heart disease, killing over 370,000 people yearly.

Detecting primary signs of heart abnormalities is quite costly. In countries where the economic condition of the country is not good and under developing and developing countries, these tests are not at all affordable. So, there is an urgent need for a system that is affordable and helpful in detecting any primary signs of an abnormality in the heart. Hence, any method which can help remove this need will have a significant impact on world health.

For monitoring to the internal sounds of human body the medical device used is the stethoscope. It is a basic device to listen to heart sound. The advantage of using an electronic stethoscope over an acoustic stethoscope is that its characteristics like increased sound output, improved frequency range, ambient noise loss, etc. It consists of an amplifier to increase the low intensity heart sound. An Electronic Stethoscope transmitted sound electronically, so, it can be a wireless machine, or can be a recording device. It can also provide visual layout of the recorded heart sound.

1.2 Objective

Our study aims to suggest a smart algorithm to discover the presence of abnormalities in the heart sound of patient's data. Also, along with this, we wanted to build a likely and affordable result.

1.3 Problem Statement

High mortality rate due to cardiovascular diseases has raised a concern among the public. There is a need of an efficient and easily accessible technique that can detect any abnormality in the heartbeat and can warn at an early stage if there is a problem. This way the common public will be helped without paying high costs for different heart tests.

1.4 Methodology

The methodology [1] proposed in this project is a basic three steps architecture data acquisition, pre-processing and classification.

1.5 Language Used

We are working with the Python programming language for the implementation of our project. Python provides a large set of libraries for deep learning, which reduce development time and reduces the complexity of algorithms. Such libraries include Keras, TensorFlow, Flask etc. which provides a wide range of algorithms for deep learning also with GPU support that reduces the training time.

Chapter 02: LITERATURE SURVEY

Recent researches that have been carried out in deep neural network architectures have resulted in more advance and adaptive systems that are able to identify and classify objects in images. For the last decade, this topic is being continuously researched. These researches have introduced many algorithms and techniques for the task of detecting. A large center of deep learning has been on the automated study of image and text data, advances are also frequently being seen in areas that require processing other input modalities. One such area is the pharmaceutical domain where data into a deep learning system could be physiologic time series data. The recent increase in challenges in the medical domain like Kaggle 2014 and 2015 challenges have resulted in improvement to deep learning architecture. The challenge held in 2016 asked the members to complete investigation on heartbeat sounds obtained using digital stethoscopes. The main objective of this challenge was to classify the heartbeat sound as normal or abnormal.

More recent research has worked on applying deep learning techniques to the same sub-tasks. Convolutional Neural Network (CNN) can extract the required features irrespective of the manipulation method. Using deep learning techniques an excellent accuracy of more than 90% is easily achievable.

We have selected some research papers and reviewed them. We have constructed a table for the comparison of the research papers.

Table 1: Comparison of different existing research papers

Sr. No	Title	Author	Year	Approach
1	Recognising Abnormal Heart Sound... [1]	<ul style="list-style-type: none"> Jonathan Rubin Rui Abreu Anurag Ganguli 	2017	The Algorithm used in this paper is to first convert the audio sounds to their respective heat maps and use those heat maps to train the CNN model and predict the output.
2	Stand-alone Heartbeat Detection... [2]	<ul style="list-style-type: none"> Matti Kaisti Mojtaba Jafari Tadi Olli Lahdenoja 	2018	Multidimension beat to beat detection algorithm and uses envelope and Clustering methods to detect beat and then finally merge the beat location.
3	Detection of cardiac abnormality... [3]	<ul style="list-style-type: none"> Samit Ari Koushik Hembram Goutam Saha 	2010	To improve the performance of the least square SVM classifier based on the Least mean square algorithm to detect heartbeat sound, this method was proposed.
4	Early and remote detection... [4]	<ul style="list-style-type: none"> KRZYSZT OF WOŁK AGNIESZ KA WOŁK 	2019	The purposed method is the utilization of CNN for preliminary screening of heart rhythm disturbances by classifying the heartbeat sounds.
5	Heart sounds Segmentation and Classification ... [5]	<ul style="list-style-type: none"> Ashwin R. Jadhav Arun G. Ghontale Anirudh Ganesh 	2017	This paper presents a 2-step approach towards detection of heartbeat anomaly. First the algorithm uses an envelope of Shannon energy and peak detection methods to isolate the peaks. After the segmentation and feature construction of the signal, a neural network is used to train and classify the heartbeat sounds into normal or murmur category.
6	Detection of heartbeat abnormalities from phonocardiography... [6]	<ul style="list-style-type: none"> Batyrkhan Omarov Khaled Gamry Aidar Batyrbekov 	2021	The paper presents the application of machine learning (MO) methods for detecting cardiovascular diseases based on phonocardiograms.

Chapter 03: SYSTEM DEVELOPMENT

3.1 Feasibility Study

The feasibility study is to evaluate the operational, technical, economic and organizational point of view that whether the project is viable i.e., it is possible to do efficiently and conveniently or not.

1. Technical Feasibility:

The heartbeat anomaly detection project has a very user-friendly UI and has been developed on PyCharm which is one of the most famous Integrated development environments (IDE) for Python and Google Colab which is one of the most trusted platforms for development. The technologies used in our project is feasible.

2. Operational Feasibility:

The project is an operational feasible model and could benefit all the common people. It can serve its purpose of detecting heart diseases at an early stage.

3. Economic Feasibility:

The project has been developed using only open-source platforms and libraries. These all libraries are made available for free by the developers on the internet. So, project development does not involve any type of expenses.

4. Market Feasibility:

The project has been developed by keeping the demands of the clients in mind. Clients like digital forensic labs and other news fast check experts. It checks all the constraints of market feasibility.

3.2 Requirements on Major Project

3.2.1 Functional Requirements

The functional requirements are the specific demands the system should offer to the end-user so that user could accomplish their tasks.

The functional requirements of this project are mentioned below.

- i. User-friendly environment for the customer with simple GUI.
- ii. When the user browses the audio file to run the detection test GUI offers a button to run the test.
- iii. Proper detection of the audio file.

3.2.2 Non-Functional Requirements

The non-functional requirements are the non-mandatory requirements that enhance the quality of the project. Some are listed below:

- i. The project provides an accuracy of more than 89% which is quite an excellent number.
- ii. The project can detect the heartbeat abnormality if any within few seconds.

3.3 Use Case Diagram of the Major Project

The graphical representation of the user's interaction with the User Interface (UI), in which the user provides input as an audio file and then runs the detection on that file and gets output on the screen. The use-case diagram is shown in figure 2.

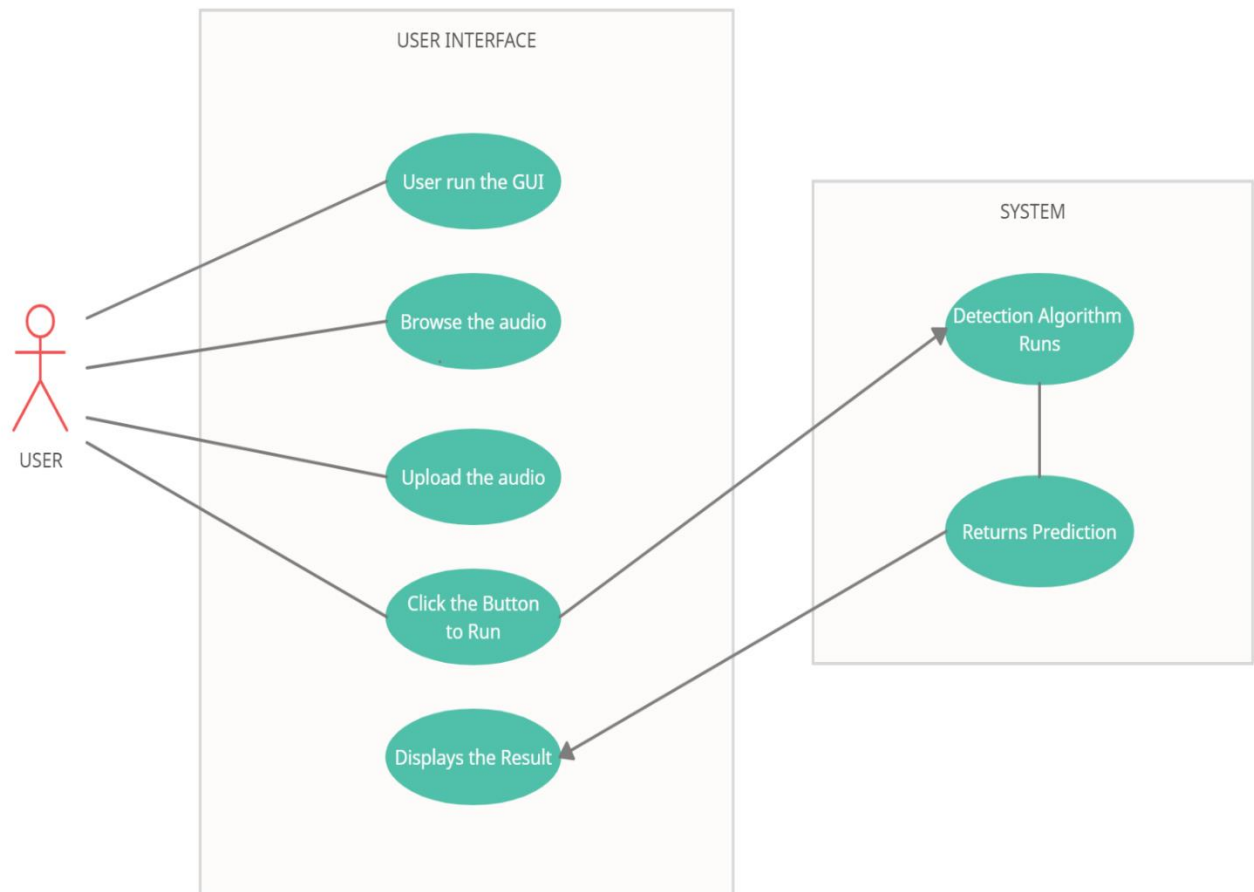


Figure 2. Use-Case Diagram

3.4 DFD Diagram

The Data Flow Diagram (DFD) diagram is the graphical representation of the flow of data in the system. It is the flow of data from the user inserting an audio file to the pre-processing and then passing it to the trained CNN to predict the result.

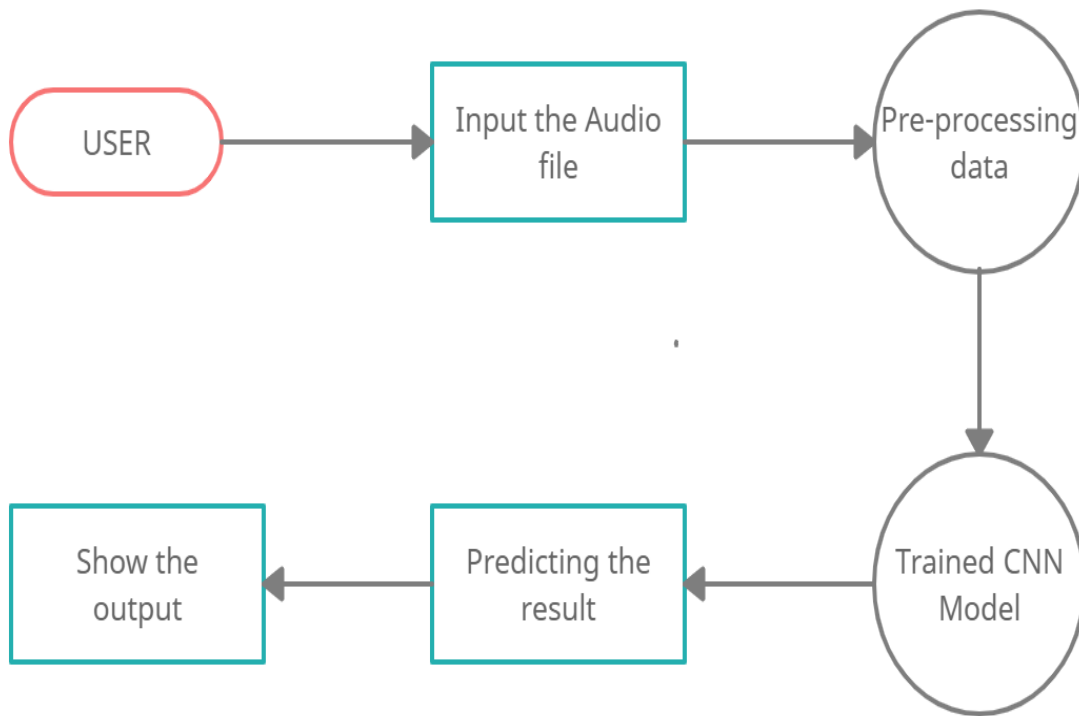


Figure 3. DFD Diagram

3.5 Date Set Used in the Major Project

This dataset was first used in a Machine Learning challenge for the classification of heartbeat sounds. Depending on the sources from where it was collected the dataset is divided into 2 sets. Set A data were obtained from the common public via the I-Stethoscope Pro iPhone app and Set B from a clinical trial in hospitals using the digital stethoscope DigiScope. There are 4 classes of heartbeat sounds:

1. Normal: healthy heart sounds
2. Murmur: extra sounds that occur when there is turbulence in blood flow that causes the extra vibrations that can be heard.
3. Extrahls: heartbeats with an additional sound.
4. Extrasystoles: additional heartbeats that occur outside the physiological heart rhythm and can cause unpleasant symptoms.

3.6 Data Set Features

3.6.1 Number of attributes, fields, description of data set

Table 2: Number of images in dataset

Set_A	Set_B
176	656

3.7 Algorithm/Pseudo code of the model

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

Pseudocode of the proposed method:

- i. Importing the dataset.
- ii. Pre-processing the dataset: The heartbeat sound collected from mobile phone and stethoscope often contains background noises. Audio files having a span of fewer than 3 seconds are cut out because they don't contain enough data points to correctly classify heartbeat. The large audio files are sliced into smaller files while retaining their original label. About half a second from the front as well as the back has been cut from the audio

files to remove the noise that is caused by physical touch of the microphone with the body.

- iii. Feature Extraction: The raw audio data utilized .wav type (Waveform Audio File Format) was in the amplitude vs time frame in the time domain. We modified this one-dimensional time-series signal into a two-dimensional heat map that catches the time-frequency pattern of the signal.
- iv. Dataset Split: The dataset is split into training and test dataset. The training dataset contains 1044 images and the test dataset contains 261 images. The images were the result of conversion of audio files to their corresponding heat maps so, that it can be given as an input to CNN architecture.
- v. Training CNN: The CNN architecture is constructed and the CNN is trained with the training dataset.
- vi. Testing the model: The saved model is now tested using the test dataset and the results are duly noted.
- vii. CNN is trained using the Adam optimizer.
- viii. A proper web page is then made for this model using flask that is deployed in cloud.

Below are the details of the Feature Extraction from audio files

In the frequency domain, we have used the following for feature extraction:

- Melspectrogram: It describes an acoustic time-frequency description of a sound. It is a spectrogram with the Mel Scale as its y-axis. This Mel Scale is constructed so that sounds of the similar in distance from each other on the Mel Scale, also “sound” to humans as they are similar in distance from one another
- MFCC (Mel Frequency Cepstral Components): We decided to use Mel Frequency Cepstral Coefficients to achieve this conversion, as MFCCs capture features from audio file that more firmly match how human beings understand loudness and tone. MFCCs are commonly used as a characteristic type in speech recognition.
- For the above we used inbuilt functions from the librosa library in python as follows:
 1. The preprocessed sound files are transformed into magnitude spectrogram and then mapped onto the mel-scale by using the inbuilt feature method of librosa to get the mel-scaled spectrogram.
 2. A cepstral analysis is performed on the Mel-Spectrum to obtain Mel-Frequency Cepstral Coefficients (MFCC) by passing the log-power Melspectrogram as an

argument to the MFCC function.

- Thus, the audio file is now represented as a sequence of Cepstral vectors. These Cepstral vectors are then given to the model for anomaly detection.

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

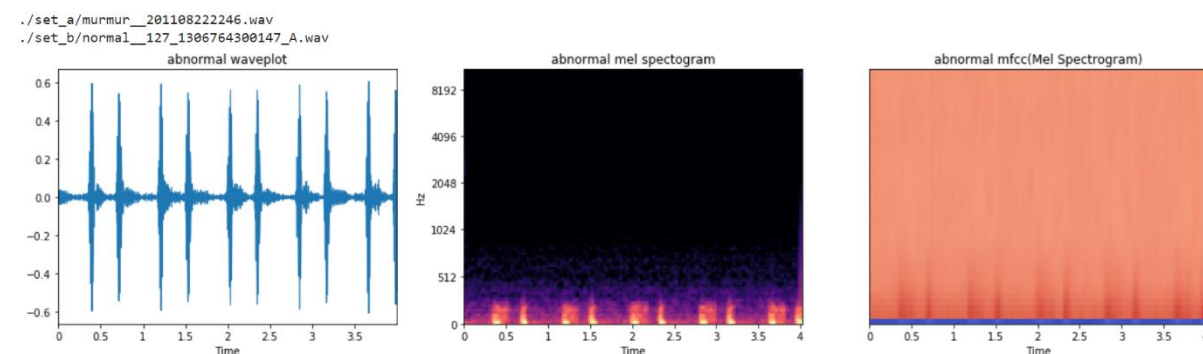


Figure 4: Abnormal Heartbeat features

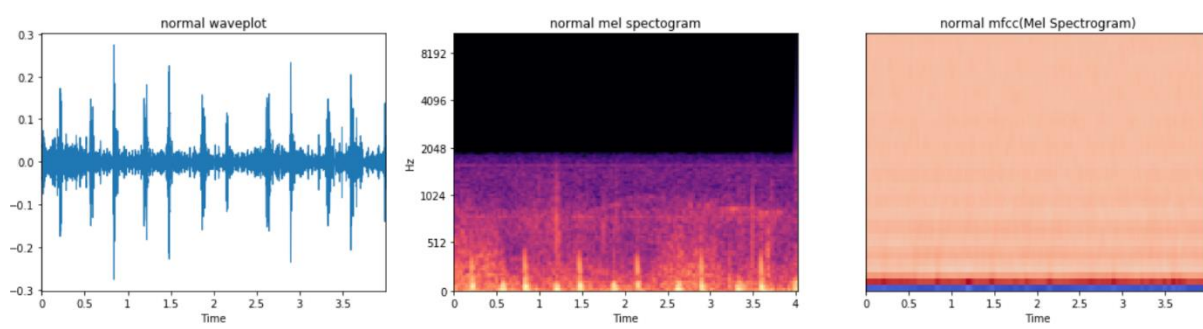


Figure 5: Normal Heartbeat Features

Below we define the architecture of CNN network.

CNN Architecture

The basic structure of a CNN is several convolutional layers succeeded by fully connected layers and a SoftMax classifier.

The neural network accepts input as a single channel of 40 X 130 MFCC heat map and gives a binary classification, predicting whether the input file segment is normal or abnormal heart sound.

CNN in this project uses 4 convolution layers, 4 max-pooling layers, 4 dropout layers, 1 global average pooling layer and finally a dense layer. The activation function used in convolution layers is ReLU. The advantage of using ReLU is that its time efficient for both training and testing.

Dropout Layer: The dropout layer provides for regularization by randomly placing any neurons in previous layers to zero while training.

Max Pooling: Max pooling layer intends to down-sample a data representation. It helps in decreasing dimensionality and eases feature extraction. It decreases the computational load and the number of parameters to be learned.

Dense layer: Here every input is connected to every output by weight. And we are using softmax as the non-linear activation function after this layer.

Proposed CNN:

- i. Comprises 4 convolutional, 4 max-pooling layers, 4 dropout and 1 global average pooling layer.
- ii. The input of the network is a 40x130x1.
- iii. All the layers use a kernel size of 2.
- iv. The Filter size in first CNN layer is 16 then in the next layer increases to 32 and then 64 and finally 128.
- v. Every convolutional layer uses the ReLU activation function.
- vi. Finally, the dense layer uses the softmax activation function.

The total parameter to be trained are 43570. Below is the summary of the CNN architecture that has been built for this project.

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 (MaxPooling2D)	(None, 9, 31, 32)	0
dropout_1 (Dropout)	(None, 9, 31, 32)	0
conv2d_2 (Conv2D)	(None, 8, 30, 64)	8256
max_pooling2d_2 (MaxPooling2D)	(None, 4, 15, 64)	0
dropout_2 (Dropout)	(None, 4, 15, 64)	0
conv2d_3 (Conv2D)	(None, 3, 14, 128)	32896
max_pooling2d_3 (MaxPooling2D)	(None, 1, 7, 128)	0
dropout_3 (Dropout)	(None, 1, 7, 128)	0
global_average_pooling2d (GlobalAveragePooling2D)	(None, 128)	0
dense (Dense)	(None, 2)	258

Total params: 43,570
 Trainable params: 43,570
 Non-trainable params: 0

Figure 6: CNN Parameters

3.8 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

```

Figure 7: Libraries used

2. Loading Dataset and Applying Pre-processing

```
dataset = []
for folder in ["/set_a/**", "/set_b/**"]:
    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
                        })
```

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

dataset = pd.DataFrame(dataset)
dataset = shuffle(dataset, random_state=42)
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1305 entries, 1172 to 1126
Data columns (total 3 columns):
#   Column      Non-Null Count  Dtype
---  -
0   filename    1305 non-null   object
1   label       1305 non-null   object
2   offset      1305 non-null   float64
dtypes: float64(1), object(2)
memory usage: 40.8+ KB
```

Figure 8: Pre-processing dataset

4. 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 9: Feature Extraction

5. Train/Test Split

```
from keras.utils.np_utils import to_categorical
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: (1044, 40, 130, 1)
Y train: (1044, 2)
X test: (261, 40, 130, 1)
Y test: (261, 2)
```

Figure 10: Splitting of Dataset

6. Training CNN.

6.1. 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]),
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'))

```

Figure 11: Architecture of CNN model

6.2. Fitting the model

The model uses Adam optimizer and a learning rate of 0.001, a batch size of 128 and an epoch of 300.

```

import tensorflow
adam = tensorflow.keras.optimizers.Adam(learning_rate=0.001)
model.compile(loss='categorical_crossentropy', metrics=['accuracy'], optimizer='adam')

history = model.fit(x_train, y_train,
                    batch_size=128, epochs=300, validation_data=(x_test, y_test),
                    shuffle=True)

```

Figure 12: Model Fitting

6.3. 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.

7. Flask Website

As, we want that this model be available to the common people easily we will deploy our CNN model. For that purpose, first we have created a simple user-friendly page using html and css and finally using the flask framework we have connected our trained CNN model with the page.

Below is how our page looks like,

Heart Anomaly Detection

With the healthy heart,
the beat goes on.

Choose File No file chosen

Detect



Figure 13: Website

The code for the above page is as follows.

App.py which connects the trained model with html page

```
from flask import Flask, render_template, request, redirect

from test import run_model

app = Flask(__name__)

@app.route('/', methods=["GET", "POST"])
def index():
    transcript = ""
    if request.method == "POST":
        print("data received")

        if "file" not in request.files:
            return redirect(request.url)

        file = request.files["file"]
        if file.filename == "":
            return redirect(request.url)

        if file:
            transcript = run_model(file)

    return render_template("index.html", transcript=transcript)
```

Figure 14: flask code

The html code is as follows.

```
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Heart Anomaly Detection</title>
  <link rel="stylesheet" href="{{url_for('static', filename='styles/style.css')}}">
</head>
<body>

  <div class="main">
    <div class="content" id="beatContainer">
      <div class="heading">Heart Anomaly Detection</div>
      <br>
      <br>
      <div class="sub-heading">
        With the healthy heart, the beat goes on.
      </div>
      <form method="post" enctype="multipart/form-data">
        <input class="upload_file" type="file" name="file"/>
        <br>
        <input type="submit" id="submitButton" class="submitButton" value="Detect"/>
      </form>
```

Figure 15: Html code


```
        {% if transcript!=" " %}
<div id="beatSubmitContainer">
    <p style="font-size: 2rem;">Result</p>
    <p id="beatText" style="font-size: 2rem;">{{transcript}}</p>
</div>
{% endif %}

</div>
<div class="banner"...>

</div>
</body>
</html>
```

Figure 16: Html code

Let's finally see our html page doing our work and predicting the audio sound.

Heart Anomaly Detection

With the healthy heart,
the beat goes on.

Choose File Bunlabelledt...23364_A.wav

Detect

Result

Normal heartbeat



Figure 17: Working of website

8. Deployment of Model on AWS EC2 Instance

In order to deploy our model, we first created a website which can be seen above. So, to deploy we used AWS ec2 instance which will give us a server where we can deploy our flask website and hence it can be used by anyone with the link of the website. AWS ec2 instance is a free of cost and easy to use. We created an ubuntu instance using ec2 and after completing all the necessary requirements we were able to deploy our model correctly.

Ubuntu server running

```
ubuntu@ip-172-31-81-175: ~  
/home/ubuntu/.local/lib/python3.6/site-packages/numba/core/errors.py:154: UserWarning: Insufficiently recent colorama version found. Numba requires colorama >= 0.3.9  
  warnings.warn(msg)  
2021-12-06 05:13:17.202388: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'libcudart.so.11.0'; dLError: libcudart.so.11.0: cannot open shared object file: No such file or directory  
2021-12-06 05:13:17.202561: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.  
* Serving Flask app 'app' (lazy loading)  
* Environment: production  
  WARNING: This is a development server. Do not use it in a production deployment.  
  Use a production WSGI server instead.  
* Debug mode: off  
* Running on all addresses.  
  WARNING: This is a development server. Do not use it in a production deployment.  
* Running on http://172.31.81.175:8080/ (Press CTRL+C to quit)  
14.139.240.50 - - [06/Dec/2021 05:13:50] "GET / HTTP/1.1" 200 -  
14.139.240.50 - - [06/Dec/2021 05:13:50] "GET /static/styles/style.css HTTP/1.1" 200 -  
14.139.240.50 - - [06/Dec/2021 05:13:51] "GET /favicon.ico HTTP/1.1" 404 -
```

Figure 18: Ubuntu

Working website deployed in AWS ec2

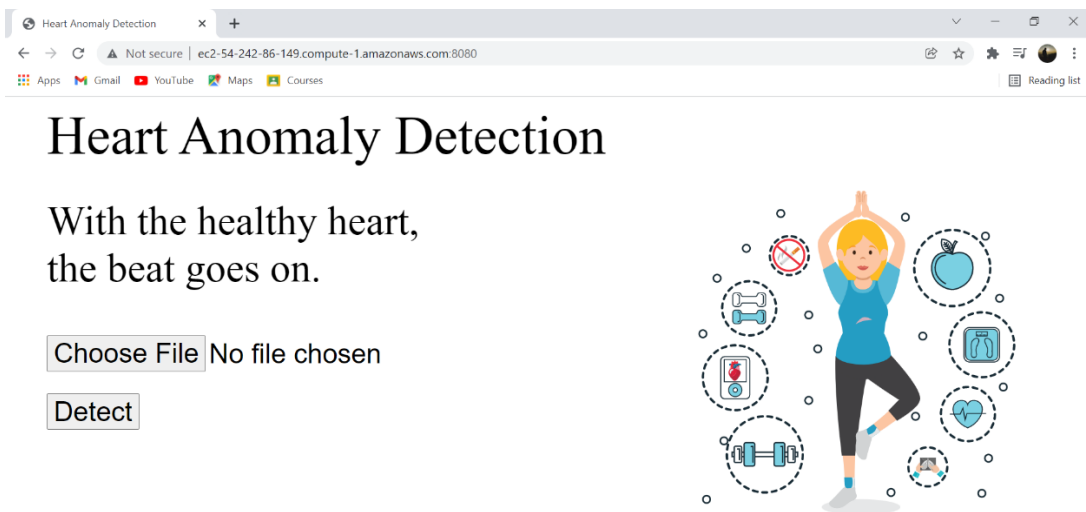


Figure 19: Working Website

Chapter 04: Performance Analysis

4.1. Predicting the result

The training dataset is now used to predict the accuracy of our trained model. The test accuracy achieved was 89%.

```
scores = model.evaluate(x_test, y_test, verbose=1)
print('Test loss:', scores[0])
print('Test accuracy:', scores[1])

9/9 [=====] - 0s 27ms/step - loss: 0.3202 - accuracy: 0.8927
Test loss: 0.32022568583488464
Test accuracy: 0.8927202820777893
```

Figure 20: Prediction

Prediction result was as follows

```
9/9 [=====] - 0s 27ms/step
      precision    recall  f1-score   support

 abnormal    0.87     0.88     0.88     113
   normal    0.91     0.90     0.90     148

 accuracy                   0.89     261
 macro avg    0.89     0.89     0.89     261
weighted avg    0.89     0.89     0.89     261
```

Figure 21: Result

4.2. Model Testing

Now we will provide our model a new data and see if it is able to perform well and predict whether the heartbeat sound is normal or abnormal.

```

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

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

# File to be classified
classify_file = "/content/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 93ms/step
[[0.02669311 0.9733069 ]]

```

Figure 22: Model testing

4.3. Discussion on the Results Achieved

Our model was successful in detecting heartbeat anomaly. The model achieved a training accuracy of approx. 89%.

These results complete our objective of the project that is to detect heartbeat anomaly using CNN achieving good accuracy.

The accuracy starts to increase from around 50% during the starting epochs. With a steady pace, it keeps on increasing till it becomes stable near the ending epochs i.e., around 88-89%. It could be seen clearly in the plotted graph in figure.

While the loss starts to decrease from the first epoch and then it continues decreasing at a gradual rate until it becomes steady near the last epochs. It could be seen clearly in the plotted graph in figure.

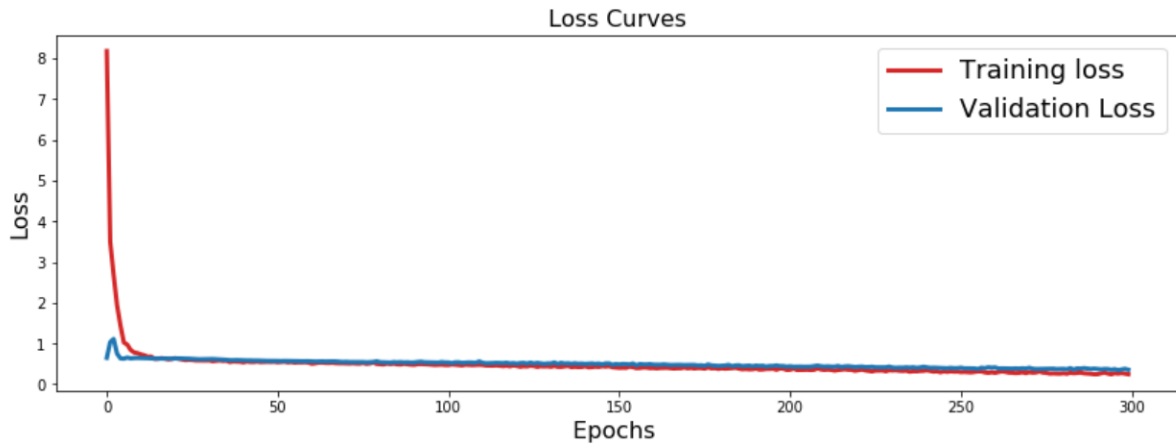


Figure 23: Loss Vs Epochs

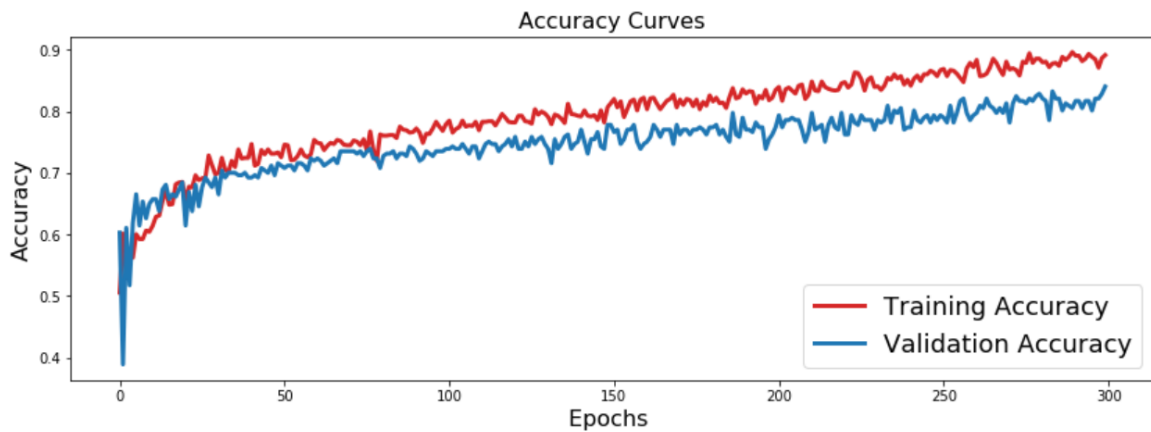


Figure 24: Accuracy Vs Epochs

4.4. Comparison

Now we compare our model with previously present models and finally came to a conclusion that our model was able to perform well just because it uses CNN and CNN has a very high accuracy rates and hence we were able to finally achieve our goal. We compared our model with a system which used LinearSVC as the algorithm to predict the result. The comparison chart of the accuracy is shown below.

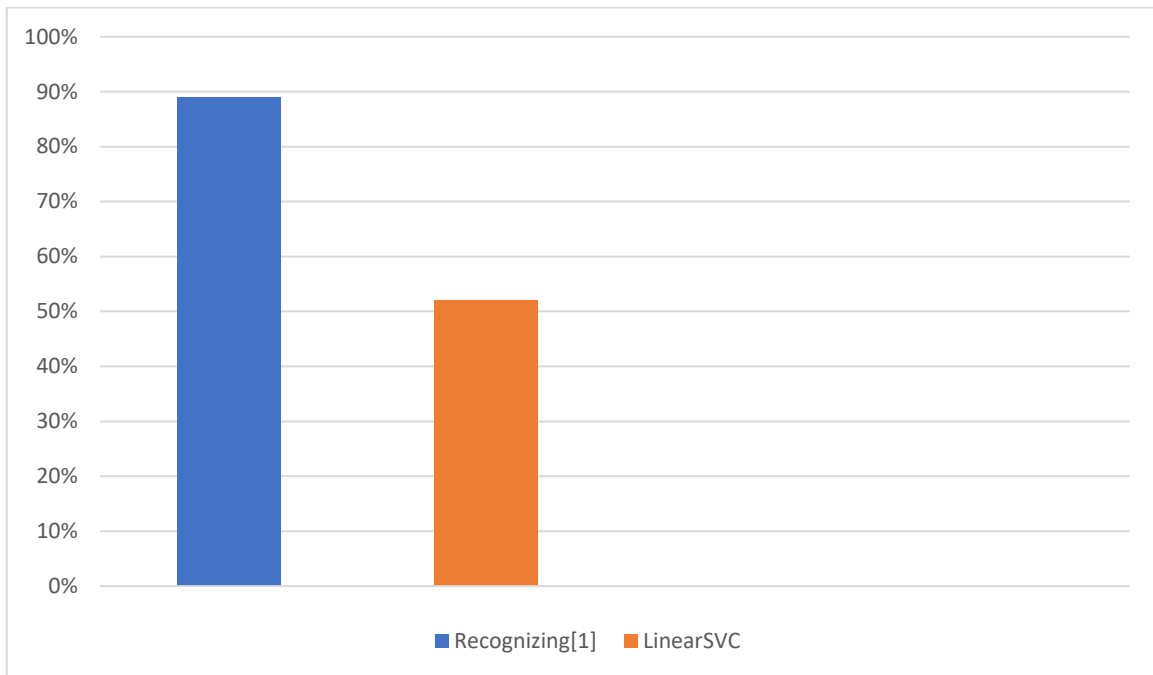


Figure 25: Comparison Chart

4.4.1. Extracting Features for LinearSVC model

We need to find more statistical parameters that we can add for our LinearSVC model to train itself. Normal statistical parameters include the min, max and mean. We know that normal audio file may contain fluctuations that can affect the accuracy of our model so, it is suggested to make envelope of sound wave amplitude and it will help to learn different patterns.

In order to create envelopes we can make use of the python rolling window of 3000 and hence find the statistical values of these envelope and merge them to the original extracted values. Hence we added 3 new features using the envelope that is mean, standard deviation and max of the envelope.

Finally we can also use tempo graphs to extract features from the audio files. Using the librosa library we can extract the tempo of the sound. We will add the extracted tempo features data to the original data and now we have total of 9 features for our model to learn from.

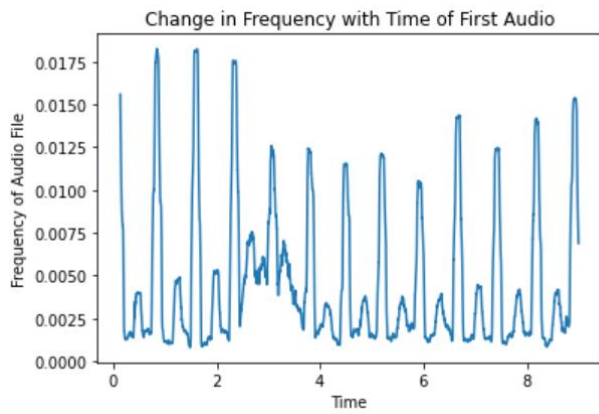


Figure 26: Envelope created

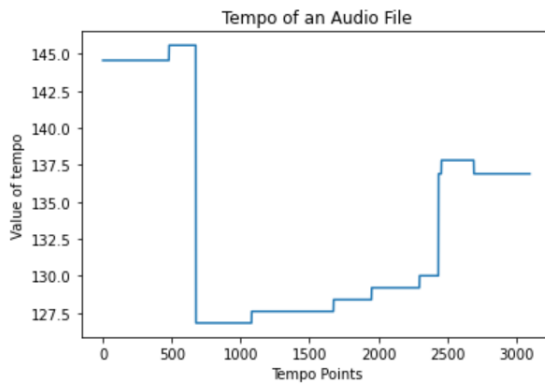


Figure 27: Tempo of the sound

```
Out[ ]:
```

	dataset	label	sfreq	min	mean	max	class	env_mean	env_std	env_max	tempo_mean	tempo_std	tempo_max
key_0													
set_a/artifact_201012172012.wav	a	artifact	22050	-0.730766	-1.04662e-05	0.712701	1	0.001324	0.005395	0.043410	117.414923	21.062747	136.899834
set_a/artifact_201105040918.wav	a	artifact	22050	-0.0173211	-1.10528e-06	0.0158929	1	0.000299	0.000534	0.003567	112.012669	9.969253	153.125000
set_a/artifact_201105041959.wav	a	artifact	22050	-0.0204537	4.74498e-07	0.0195355	1	0.000485	0.000565	0.003529	130.103762	22.025754	187.926136
set_a/artifact_201105051017.wav	a	artifact	22050	-0.920064	-1.36942e-05	0.762148	1	0.010100	0.004798	0.027056	111.872001	17.118809	134.232955
set_a/artifact_201105060108.wav	a	artifact	22050	-0.905326	-2.86369e-05	0.924389	1	0.012662	0.017345	0.125319	124.033688	24.434358	161.499023

Figure 28: Features Extracted

Chapter 05: Conclusions

5.1 Application of the Major Project

There's a need for a good and stable system that can easily detect heartbeat anomaly. In the current world scenario, one of the leading causes of death is from heart diseases.

Our model is capable of detecting any abnormality in the heartbeat sound. Even when the audio sound has been recorded from a mobile phone.

It could be used by any medical professionals and even by non-professionals due to its simple GUI design, it covers a range of people.

5.2 Limitation of the Major Project

Even though our model achieved a good accuracy rate however it is still vulnerable to some false classifications.

The accuracy of the model could have further been improved if the dataset had been large, as CNN performs well on large dataset as it is able to learn from almost all case scenarios.

As the model can only tell if there is a problem with the heartbeat, the exact detection would require tests conducted by a medical professional.

5.3 Future Work

The future work includes to collect more varying type of dataset for training of the model. Also, as the output of our model was binary i.e., normal or abnormal, our future work could remove this restriction and include some more specific reasons and diagnose properly the heartbeat sound and give more precise results. Also, we can consider signal quality as an input for our architecture can improve the performance and our system will be more reliable.

References

[1] [Jonathan Rubin, et al. 2017] Recognizing Abnormal Heart Sounds Using Deep Learning

<https://arxiv.org/abs/1707.04642v2>

[2] 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.

<https://ieeexplore.ieee.org/document/8485687>

[3] 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>

[4] Wołk, K., & Wołk, A. (2019). Early and remote detection of possible heartbeat problems with convolutional neural networks and multipart interactive training. *IEEE Access*, 7, 145921-145927.

<https://ieeexplore.ieee.org/abstract/document/8723396>

[5] Jadhav, A. R., Ghontale, A. G., & Ganesh, A. (2017, July). Heart sounds segmentation and classification using adaptive learning neural networks. In *2017 International Conference on Signal Processing and Communication (ICSPC)* (pp. 33-38). IEEE.

<https://ieeexplore.ieee.org/abstract/document/8305881>

[6] 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.

<https://ieeexplore.ieee.org/abstract/document/9377164>