MAHENDRA ENGINEERING COLLEGE

(AUTONOMOUS)

Mahendhirapuri, Mallasamudram, Namakkal Dt.637 503.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

PROJECT REPORT ON

HX 8001 PROFESSIONAL READINESS FOR INNOVATION,

EMPLOYABILITY AND ENTREPRENEURSHIP

(Naalaiya Thiran Program)

PROJECT TITLE

Classification Of Arrhythmia by Using Deep Learning With 2-D ECG Spectral Image Representation

TEAM ID: PNT2022TMID16902

TEAM MEMBERS:

1. **ANNAPOORNA M (TEAM LEAD)**
2. **ANUKEERTHANA M**
3. **ANUPRIYA G**
4. **MOHANAPRIYA K**

Classification Of Arrhythmia by Using Deep Learning With 2-D ECG Spectral Image Representation

1.INTRODUCTION

1.1 Project Overview

According to the World Health Organization (WHO), cardiovascular diseases (CVDs) are the number one cause of death today. Over 17.7 million people died from CVDs in the year 2017 all over the world which is about 31% of all deaths, and over 75% of these deaths occur in low and middle-income countries. Arrhythmia is a representative type of CVD that refers to any irregular change from the normal heart rhythms. There are several types of arrhythmia including atrial fibrillation, premature contraction, ventricular fibrillation, and tachycardia. Although a single arrhythmia heartbeat may not have a serious impact on life, continuous arrhythmia beats can result in fatal circumstances. In this project, we build an effective electrocardiogram (ECG) arrhythmia classification method using a convolutional neural network (CNN), in which we classify ECG into seven categories, one being normal and the other six being different types of arrhythmia using deep two-dimensional CNN with grayscale ECG images. We are creating a web application where the user selects the image which is to be classified. The image is fed into the model that is trained and the cited class will be displayed on the webpage.

2.Literature Review:

[1] Classification of Arrhythmia by Using Deep Learning with 2- D ECG Spectral Image Representation Author name: Rizwana Naz Asif, Kiran Sultan, SuleimanAli Alsaif, Sagheer Abbas , Muhammad year:2022

Transfer learning methods to ensure accuracy and time management to detect the ECG in a better way in comparison to the previous and machine learning methods and implementation of the proposed method.

Advantages: The proposed 2- D CNN model attained better accuracy, sensitivity, and specificity (in eight class classification) than the FFNN model, which classified only four kinds of arrhythmia.The computational resources and the simulation time for training and testing modes also increase and this is the main reason for using a carefully selected CNN model.

Disadvantages : limited data transfer and the speed of the convergence was very slow.

[1] Comparing Feature-Based Classifiers and Convolutional Neural
Networks to Detect Arrhythmia from Short Segments of ECG
Author name: Fernando Andreotti, Oliver Carr, Marco A. F. Pimentel, Adam Mahdi, Maarten
Year:2017

In this study, we classify short segments of ECG into four classes (AF, normal, other rhythms or noise) as part of the Physionet/Computing in Cardiology Challenge 2017. We compare a state-ofthe-art feature-based classifier with a convolutional neural network approach. Both methods were trained using the challenge data, supplemented with an additional database derived from Physionet.

Advantages: The featurebased classifier obtained an F 1 score of 72.0% on the training set (5- fold crossvalidation), and 79% on the hidden test set. Similarly, the convolutional neural network scored 72.1%on the augmented database and 83% on the test set and the latter method resulted on a final score of 79% at the competition.

Disadvantages: They are computationally expensive and require large datasets .Inconsistent noise classification and annotating the normal segments

[3] ECG Classification for Detecting ECG ArrhythmiaEmpowered with
Deep Learning App roache Author name: Rizwana Naz,Kiran
Sultan,Suleiman Ali Alsaif year:2022
Transfer learning methods to ensure accuracy and time management to detect the ECG in a better way in comparison to the previous and machine learning methods and to implementation of the proposed method.

Advantages: There are a lot of problems like loss of data, data size limitations, redundancy Deep learning is already in practice to do a variety of tasks in pattern and image recognition and motivated the medical research to work in this state of the art. Blood Pressure (BP) estimation can be evaluated by the ECG signals

Disadvantages: The quality of morphological features extraction in the ECG greatly a & ects the recognition and classification rate of ECG signals Detection of irregular heartbeats from ECG signals is a significant task for the automatic diagnosis of cardiovascular diseases

[4] A Review on Cardiac Abnormalities Classificationusing
Electrocardiogram with Machine Learning and Deep Learning
Classification Techniques. Author name: Muhammad Adnan Khan,Amir Mosavi,Shashank Yadav,Upendra Kumar. year:2020

This survey is focusing on the latest research papers in which machine learning and deep learning classification techniques are applied in different manners.

Advantages: They found highest accuracy rate 99.3% by using k-NN classification by feeding genetic algorithm features.They recorded ECG signals in two differentsituationstechnique on the WEKA software for clas sification and they utilized MIT-BIH arrhythmia database. During classification they found accuracy rate of 88.49%.

Disadvantages: The adoption of featureds specifically switched theextraction features manually and this approach couly help in examining cardiac patient efficiently by the doctors. Training and testing sets, they transformed one dimensional ECG signals to two- dimensional image and classified the ECG data into five classes with 99.21% average accuracy. **[5]** Arrhythmia Classification Techniques Using Deep Neural Network
Author name: Ali Haider Khan,Muzammil Hussain,Muhammad Kamran Malik year:2021

The Cardiac disorder and arrhymia detection, analysis of electrocardiogram (ECG) Signals has become the focus of numerous reserches. 2D Graph Fourier transforn (GFT) was developed

Advantages: All the classes of ECG Arrhythmia and give the accuracy against training set and validation dataset and tables shows the percentage

Accuracy of different transfer learning approaches for the proposed CAATL model and discovered that all three different transfer learning approaches performed well.

Disadvantages: E most ECG databases are not specific to their clinical context. E description of the patient population in which these ECGs were obtained is lacking. important in interpreting the methodology and clinical utility in context.

[6] Classification of Arrhythmia in Heartbeat Detection Using Deep Learning Comput Intell Neurosci. Author name: Wusat Ullah, Imran Siddique, Rana Muhammad Zulqarnain, Mohammad Mahtab Alam, Irfan Ahmad, and Usman Ahmad Raza Year 2021

Aims to apply deep learning techniques dataset to classify arrhythmia. By using two kinds of the dataset. One dataset is the MIT-BIH arrhythmia database and the second database is PTB Diagnostic ECG Database.

Advantages: The result achieved by using these three techniques shows the accuracy of 99.12% for the CNN model, 99.3% for CNN + LSTM, and 99.29% for CNN + LSTM + Atten tion Model.

Disadvantages: Challenges in designing and adjusting CNN models, the high computational cost of neural networks and requires a large dataset for successful training.

REFERENCE:

1. Classification of Arrhythmia by Using Deep Learning with 2- D ECG Spectral Image Representation year 2022. Computer and Electronics Systems Engineering, Hankuk University of Foreign Studies, Yongin-si 17035, Korea Information and Communication Technology Department, School of Electrical and Computer Engineering, Xiamen University Malaysia, Sepang 43900, Malaysia.

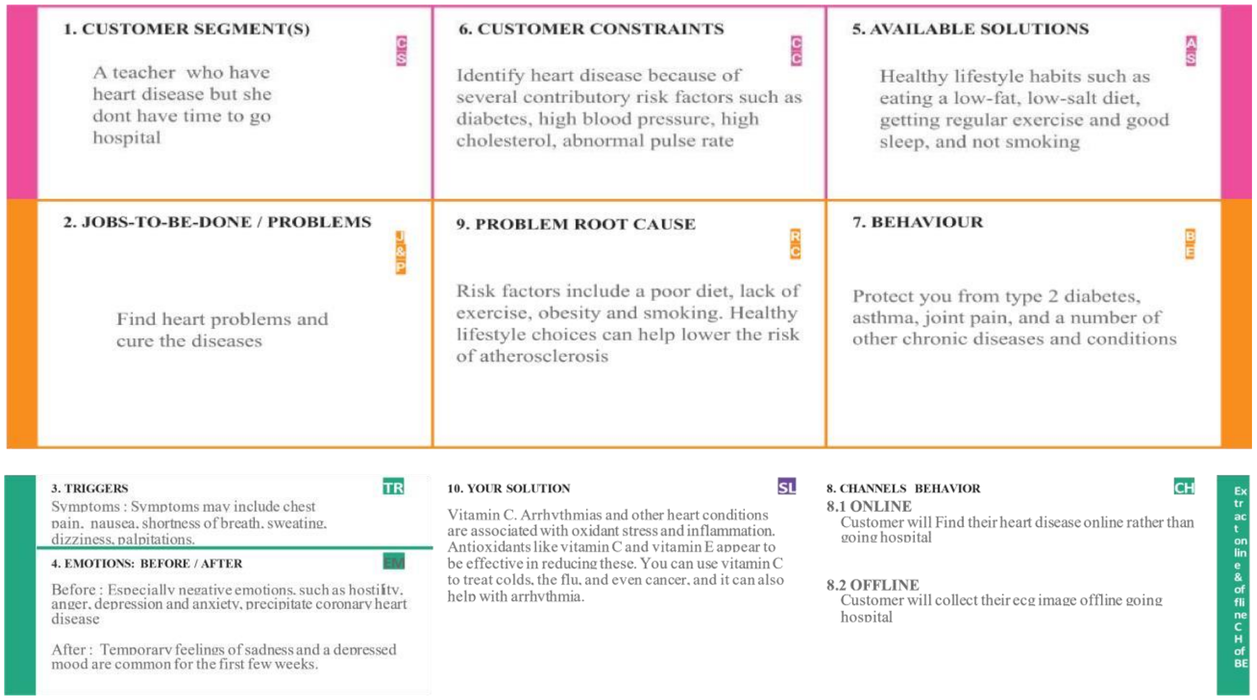
2. Classification of Arrhythmia in Heartbeat Detection Using Deep Learning Comput Intell Neurosci. 2021; 2021: 2195922.

3. Comparing Feature-Based Classifiers and Convolutional Neural Networks to Detect Arrhythmia from Short Segments of ECG
Fernando Andreotti, Oliver Carr, Marco A. F. Pimentel, Adam Mahdi, Maarten De Vos Institute of Biomedical Engineering, University of Oxford, Oxford, United Kingdom.

4. A Review on Cardiac Abnormalities Classification using Electrocardiogram with Machine Learning and Deep Learning Classification Techniques December 2020 INTERNATIONAL JOURNAL OF COMPUTER SCIENCES AND ENGINEERING 8(12):74-84 DOI:10.26438/ijcse/v8i12.7484.

5. Comparing Feature Based Classifiers and Convolutional Neural Networks to Detect Arrhythmia from Short Segments of ECG September 2017 DOI:10.22489/CinC.2017.360-239 Conference: 2017 Computing in Cardiology Conference.

2.3 PROBLEM STATEMENT



1. CUSTOMER SEGMENT(S) CS

A teacher who have heart disease but she dont have time to go hospital

6. CUSTOMER CONSTRAINTS CC

Identify heart disease because of several contributory risk factors such as diabetes, high blood pressure, high cholesterol, abnormal pulse rate

5. AVAILABLE SOLUTIONS AS

Healthy lifestyle habits such as eating a low-fat, low-salt diet, getting regular exercise and good sleep, and not smoking

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Find heart problems and cure the diseases

9. PROBLEM ROOT CAUSE RC

Risk factors include a poor diet, lack of exercise, obesity and smoking. Healthy lifestyle choices can help lower the risk of atherosclerosis

7. BEHAVIOUR BE

Protect you from type 2 diabetes, asthma, joint pain, and a number of other chronic diseases and conditions

3. TRIGGERS TR

Symptoms : Symptoms may include chest pain, nausea, shortness of breath, sweating, dizziness, palpitations.

4. EMOTIONS: BEFORE / AFTER EM

Before : Especially negative emotions, such as hostility, anger, depression and anxiety, precipitate coronary heart disease

After : Temporary feelings of sadness and a depressed mood are common for the first few weeks.

10. YOUR SOLUTION SL

Vitamin C. Arrhythmias and other heart conditions are associated with oxidant stress and inflammation. Antioxidants like vitamin C and vitamin E appear to be effective in reducing these. You can use vitamin C to treat colds, the flu, and even cancer, and it can also help with arrhythmia.

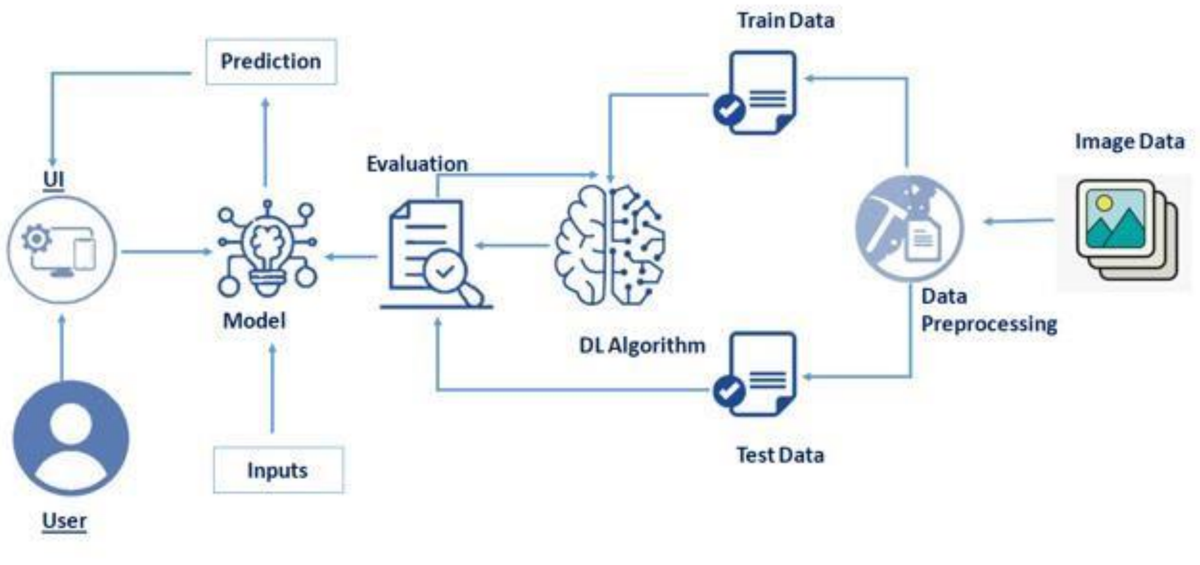
8. CHANNELS BEHAVIOR CH

8.1 ONLINE
Customer will Find their heart disease online rather than going hospital

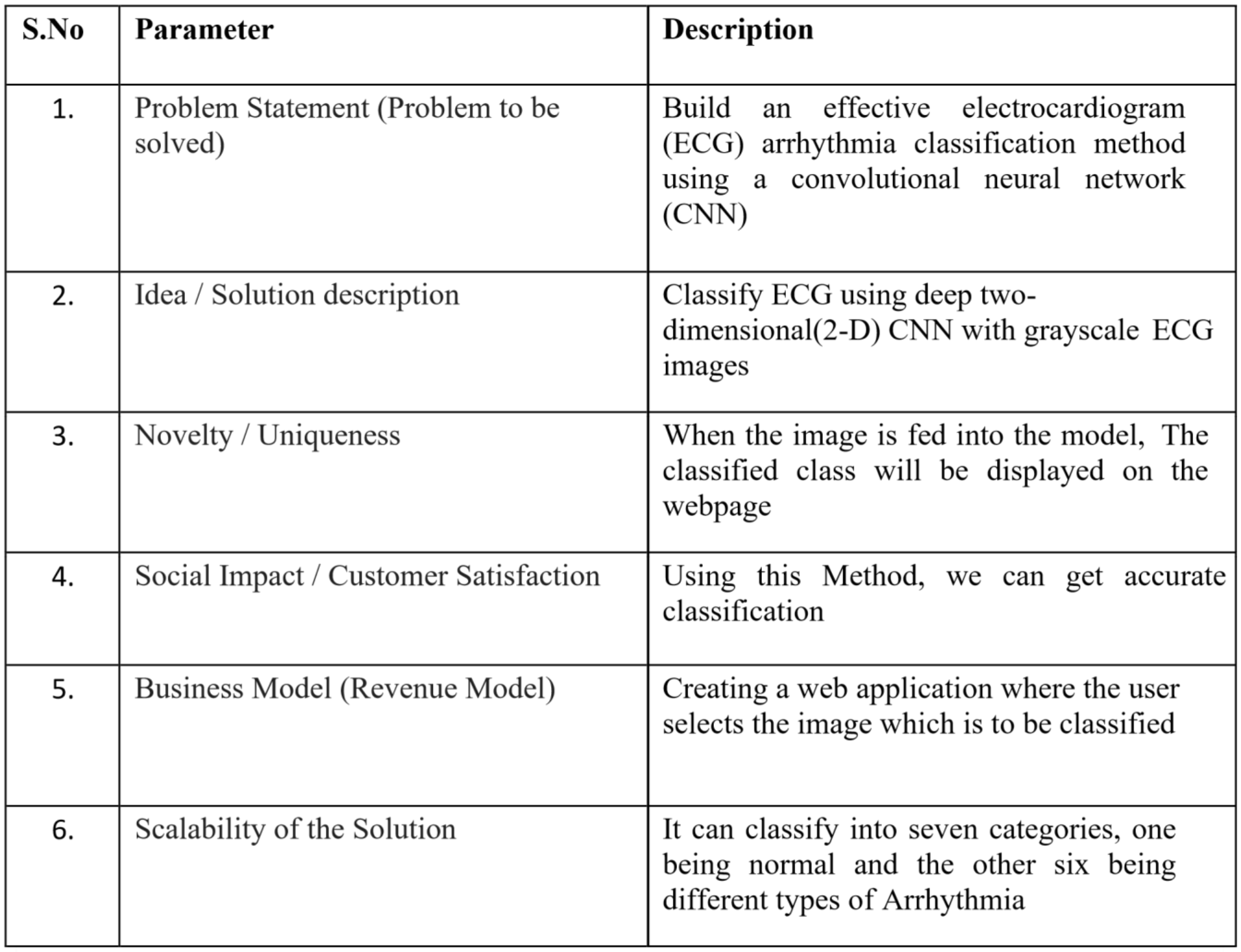
8.2 OFFLINE
Customer will collect their ecg image offline going hospital

Extract online & offline CH of BE

Technical Architecture:



Proposed Solution Template:



S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	Build an effective electrocardiogram (ECG) arrhythmia classification method using a convolutional neural network (CNN)
2.	Idea / Solution description	Classify ECG using deep two-dimensional(2-D) CNN with grayscale ECG images
3.	Novelty / Uniqueness	When the image is fed into the model, The classified class will be displayed on the webpage
4.	Social Impact / Customer Satisfaction	Using this Method, we can get accurate classification
5.	Business Model (Revenue Model)	Creating a web application where the user selects the image which is to be classified
6.	Scalability of the Solution	It can classify into seven categories, one being normal and the other six being different types of Arrhythmia

Project Objectives:

By the end of this project you will:

- Know fundamental concepts and techniques of the Artificial Neural Network and Convolution Neural Networks
- Gain a broad understanding of image data.
- Work with Sequential type of modeling
- Work with Keras capabilities
- Work with image processing techniques
- know how to build a web application using the Flask framework.

Project Flow:

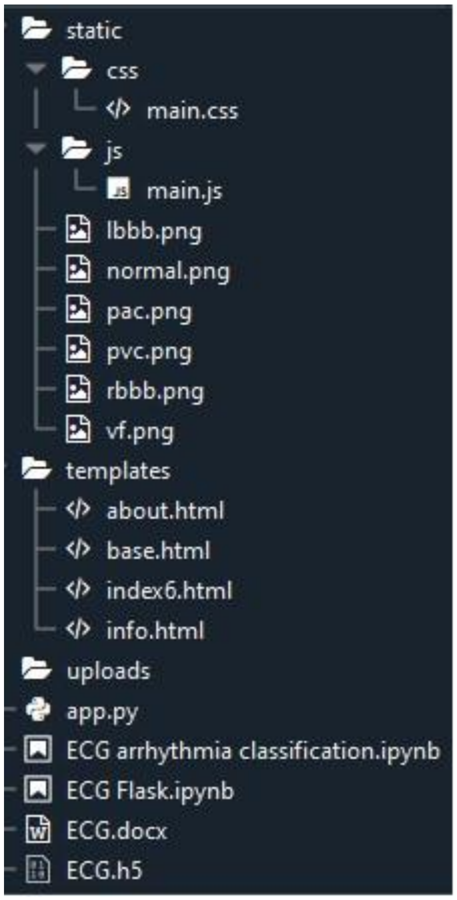
- User interacts with User interface to upload image
- Uploaded image is analyzed by the model which is integrated
- Once model analyses the uploaded image, the prediction is showcased on the UI

To accomplish this, we have to complete all the activities and tasks listed below

- Data Collection. ○ Collect the dataset or Create the dataset
- Data Preprocessing.
    - ○ Import the ImageDataGenerator library ○ Configure ImageDataGenerator class
    - ○ Apply ImageDataGenerator functionality to Trainset and Testset
- Model Building ○ Import the model building Libraries ○ Initializing the model ○ Adding Input Layer ○ Adding Hidden Layer ○ Adding Output Layer ○ Configure the Learning Process ○ Training and testing the model ○ Optimize the Model ○ Save the Model
- Application Building ○ Create an HTML file ○ Build Python Code

Project Structure

Create a Project folder which contains files as shown below



```
static
  css
    main.css
  js
    main.js
  lbbb.png
  normal.png
  pac.png
  pvc.png
  rbbb.png
  vf.png
templates
  about.html
  base.html
  index6.html
  info.html
uploads
app.py
ECG arrhythmia classification.ipynb
ECG Flask.ipynb
ECG.docx
ECG.h5
```

- We are building a Flask Application that needs HTML pages stored in the templates folder and a python script app.py for serverside scripting
- we need the model which is saved and the saved model in this content is ECG.h5
- The static folder will contain js and CSS files.
- Whenever we upload an image to predict, those images are saved in the uploads folder.

Prerequisites:

To complete this project you should have the following software and packages

Anaconda Navigator :

Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS.Conda is an open-source, cross-platform, package management system. Anaconda comes with so very nice tools like JupyterLab, Jupyter Notebook, QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code. For this project, we will be using Jupiter notebook and spyder

To install Anaconda navigator and to know how to use Jupyter Notebook a Spyder using Anaconda watch the video

To build Deep learning models you must require the following packages

Tensor flow: TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers can easily build and deploy ML powered applications.

Keras: Keras leverages various optimization techniques to make high level neural network API easier and more performant. It supports the following features:

- Consistent, simple and extensible API.
- Minimal structure - easy to achieve the result without any frills.
- It supports multiple platforms and backends.
- It is user friendly framework which runs on both CPU and GPU.
- Highly scalability of computation.

Flask: Web frame work used for building Web applications
Watch the below video to Install the necessary Packages

Prior Knowledge:

Supervised and unsupervised learning:

Watch the below video to know about the types of machine learning

Dataset Collection:

Artificial Intelligence is a data hunger technology, it depends heavily on data, without data, it is impossible for a machine to learn. It is the most crucial aspect that makes algorithm training possible. In Convolutional Neural Networks, as it deals with images, we need training and testing data set. It is the actual data set used to train the model for performing various actions. In this activity lets focus of gathering the dataset

Download The Dataset

You can collect datasets from different open sources like kaggle.com, data.gov, UCI machine learning repository, etc.

The dataset used for this project is in this Link Please refer to the link to download the data set

The dataset contains six classes:

1. Left Bundle Branch Block
2. Normal
3. Premature Atrial Contraction
4. Premature Ventricular Contractions
5. Right Bundle Branch Block
6. Ventricular Fibrillation

https://drive.google.com/file/d/16SUrk6lMaakmVf4axGNDub3joHl-XdBT/view?usp=sharing

Image Preprocessing

Image Pre-processing includes the following main tasks

- Import ImageDataGenerator Library.
- Configure ImageDataGenerator Class.
- Applying ImageDataGenerator functionality to the trainset and test set.

Note: The ImageDataGenerator accepts the original data, randomly transforms it, and returns only the new, transformed data.

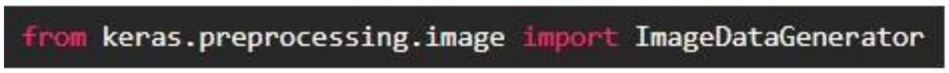
To know more about the data generator class click on this **link**

Import The ImageDataGenerator Library

Image data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset.

The Keras deep learning neural network library provides the capability to fit models using image data augmentation via the ImageDataGenerator class.

Let us import the ImageDataGenerator class from Keras



```
from keras.preprocessing.image import ImageDataGenerator
```

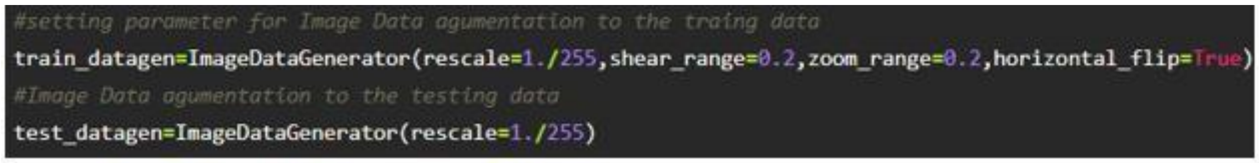
Configure ImageDataGenerator Class

There are five main types of data augmentation techniques for image data; specifically:

- Image shifts via the width_shift_range and height_shift_range arguments.
- Image flips via the horizontal_flip and vertical_flip arguments.

- Image rotates via the rotation_range argument • Image brightness via the brightness_range argument.
- Image zooms via the zoom_range argument.

An instance of the ImageDataGenerator class can be constructed for train and test.

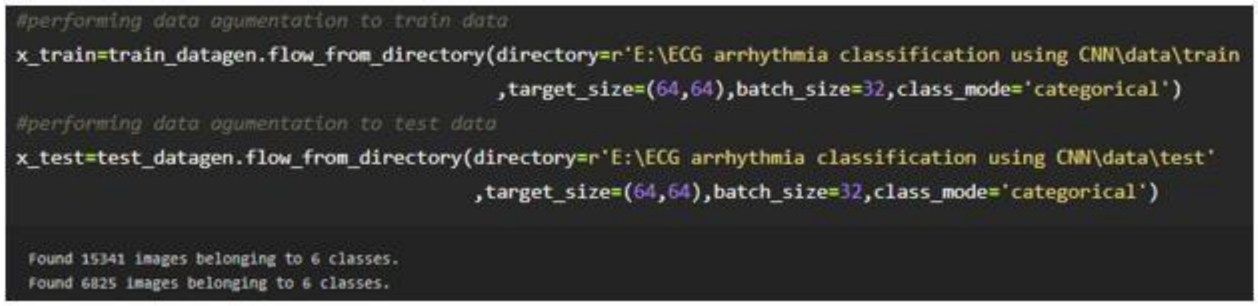


```
#setting parameter for Image Data agumentation to the traing data
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
#Image Data agumentation to the testing data
test_datagen=ImageDataGenerator(rescale=1./255)
```

Apply ImageDataGenerator Functionality To Trainset And Testset

Let us apply ImageDataGenerator functionality to Train set and Test set by using the following code

This function will return batches of images from the subdirectories Left Bundle Branch Block, Normal, Premature Atrial Contraction, Premature Ventricular Contractions, Right Bundle Branch Block and Ventricular Fibrillation, together with labels 0 to 5{'Left Bundle Branch Block': 0, 'Normal': 1, 'Premature Atrial Contraction': 2, 'Premature Ventricular Contractions': 3, 'Right Bundle Branch Block': 4, 'Ventricular Fibrillation': 5}



```
#performing data agumentation to train data
x_train=train_datagen.flow_from_directory(directory=r'E:\ECG arrhythmia classification using CNN\data\train
                                          ,target_size=(64,64),batch_size=32,class_mode='categorical')
#performing data agumentation to test data
x_test=test_datagen.flow_from_directory(directory=r'E:\ECG arrhythmia classification using CNN\data\test'
                                        ,target_size=(64,64),batch_size=32,class_mode='categorical')

 Found 15341 images belonging to 6 classes.
 Found 6825 images belonging to 6 classes.
```

We can see that for training there are 15341 images belonging to 6 classes and for testing there are 6825 images belonging to 6 classes.

Arguments:

- directory: Directory where the data is located. If labels is "inferred", it should contain subdirectories, each containing images for a class. Otherwise, the directory structure is ignored.
- batch_size: Size of the batches of data. Default: 32.
- target_size: Size to resize images to after they are read from disk.
- class_mode:
  - ‘int': means that the labels are encoded as integers (e.g. for sparse_categorical_crossentropy loss). ○ 'categorical' means that the labels are encoded as a categorical vector (e.g. for categorical_crossentropy loss).
  - 'binary' means that the labels (there can be only 2) are encoded as float32 scalars with values 0 or 1 (e.g. for binary_crossentropy).
  - None (no labels).

Model Building

We are ready with the augmented and pre-processed image data, Lets begin our model building, this activity includes the following steps

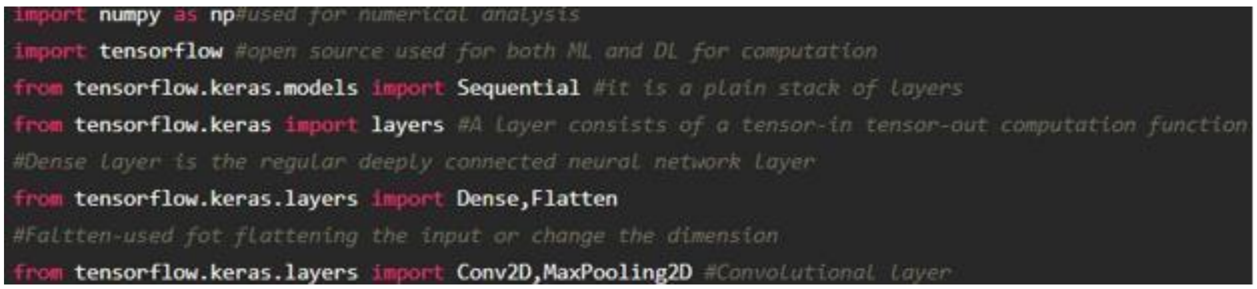
- Import the model building Libraries
- Initializing the model
- Adding CNN Layers
- Adding Hidden Layer
- Adding Output Layer

- Configure the Learning Process
- Training and testing the model
- Saving the model

To know more about model building please click here

Import The Libraries

This is a very crucial step in our deep learning model building process. We have to define how our model will look and that requires.



```
import numpy as np#used for numerical analysis
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A layer consists of a tensor-in tensor-out computation function
#Dense layer is the regular deeply connected neural network layer
from tensorflow.keras.layers import Dense,Flatten
#Faltten-used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D,MaxPooling2D #Convolutional layer
```

Initialize The Model

Keras has 2 ways to define a neural network:

- Sequential
- Function API

The Sequential class is used to define linear initializations of network layers which then, collectively, constitute a model. In our example below, we will use the Sequential constructor to create a model, which will then have layers added to it using the add () method.
Now, will initialize our model.

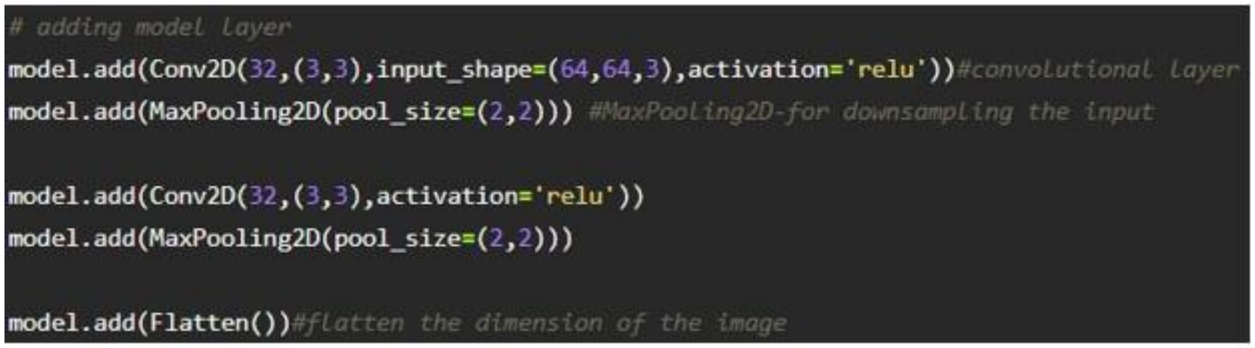
Adding CNN Layers

For information regarding CNN Layers refer to the **link**

We are adding a **convolution layer** with an activation function as “relu” and with a small filter size (3,3) and a number of filters as (32) followed by a max-pooling layer.

The **Max pool layer** is used to down sample the input.

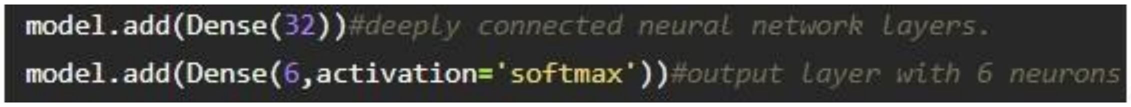
The **flatten layer** flattens the input.



```
# adding model layer
model.add(Conv2D(32,(3,3),input_shape=(64,64,3),activation='relu'))#convolutional layer
model.add(MaxPooling2D(pool_size=(2,2))) #MaxPooling2D-for downsampling the input

model.add(Conv2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())#flatten the dimension of the image
```

Adding Dense Layers

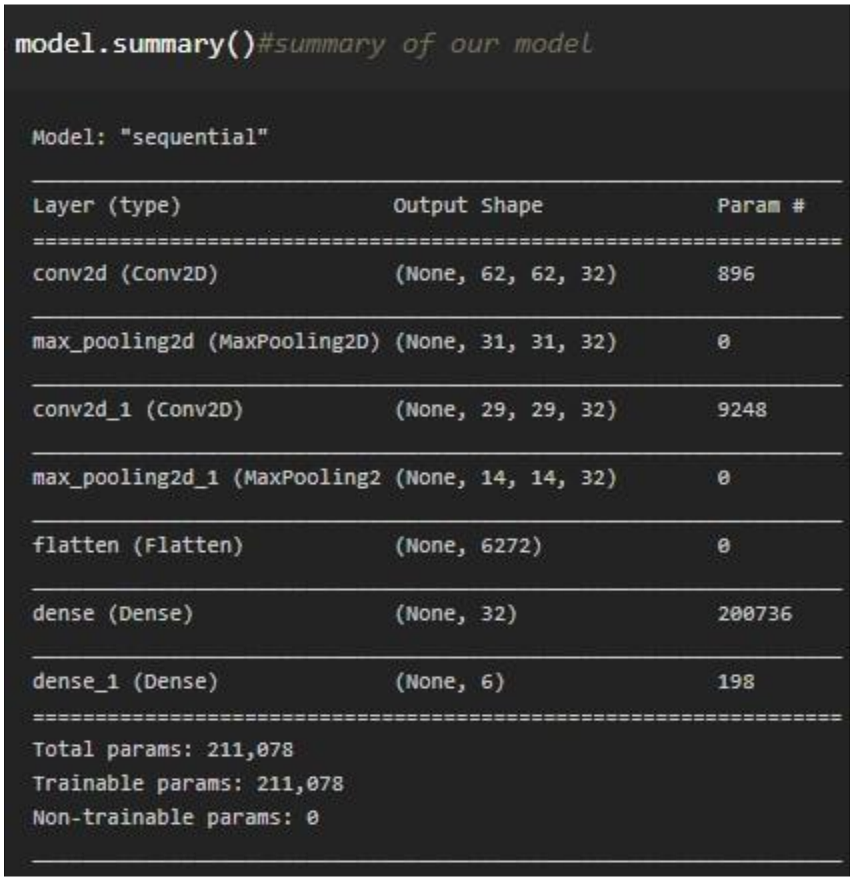
Dense layer is deeply connected neural network layer. It is most common and frequently used layer.



```
model.add(Dense(32))#deeply connected neural network layers.
model.add(Dense(6,activation='softmax'))#output layer with 6 neurons
```

We have 6 neurons in op layer as we have considered 6 classes

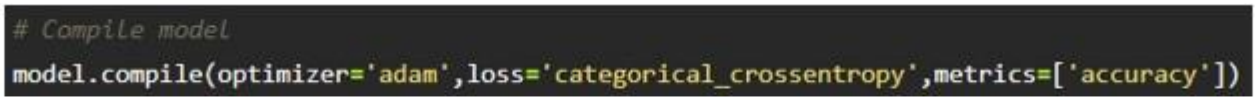
Understanding the model is very important phase to properly use it for training and prediction purposes. Keras provides a simple method, summary to get the full information about the model and its layers.



```
model.summary()#summary of our model

Model: "sequential"
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
conv2d (Conv2D)             (None, 62, 62, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 31, 31, 32)       0
_________________________________________________________________
conv2d_1 (Conv2D)           (None, 29, 29, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 32)       0
_________________________________________________________________
flatten (Flatten)           (None, 6272)              0
_________________________________________________________________
dense (Dense)               (None, 32)                200736
_________________________________________________________________
dense_1 (Dense)             (None, 6)                 198
=================================================================
Total params: 211,078
Trainable params: 211,078
Non-trainable params: 0
_________________________________________________________________
```

Configure The Learning Process

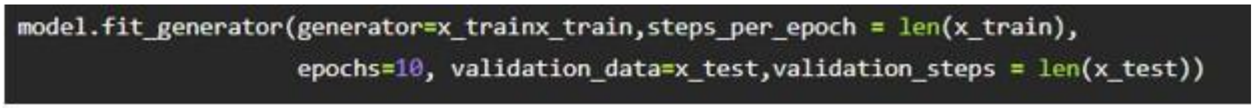
- The compilation is the final step in creating a model. Once the compilation is done, we can move on to the training phase. The loss function is used to find error or deviation in the learning process. Keras requires loss function during the model compilation process.
- Optimization is an important process that optimizes the input weights by comparing the prediction and the loss function. Here we are using adam optimizer
- Metrics is used to evaluate the performance of your model. It is similar to loss function, but not used in the training process



```
# Compile model
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

Train The Model

Now, let us train our model with our image dataset. fit_generator functions used to train a deep learning neural network



```
model.fit_generator(generator=x_trainx_train,steps_per_epoch = len(x_train),
                    epochs=10, validation_data=x_test,validation_steps = len(x_test))
```

steps_per_epoch: it specifies the total number of steps taken from the generator as soon as one epoch is finished and next epoch has started. We can calculate the value of steps_per_epoch as the total number of samples in your train dataset divided by the batch size.

Epochs: an integer and number of epochs we want to train our model for.
validation_data can be either:

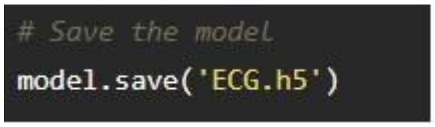
- an inputs and targets list
- a generator
- an inputs, targets, and sample_weights list which can be used to evaluate the loss and metrics for any model after any epoch has ended.

validation_steps: only if the **validation_data** is a generator then only this argument can be used. It specifies the total number of steps taken from the generator before it is stopped at every epoch and its value is calculated as the total number of validation data points in your test dataset divided by the validation batch size.
Note: it will take time for training your data based on epochs

Save The Model

The model is saved with .h5 extension as follows

An H5 file is a data file saved in the Hierarchical Data Format (HDF). It contains multidimensional arrays of scientific data.

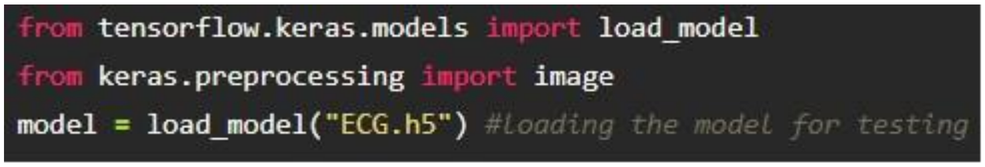


```python
# Save the model
model.save('ECG.h5')
```

Test The Model

Open the other jupyter file and write the code mentioned below

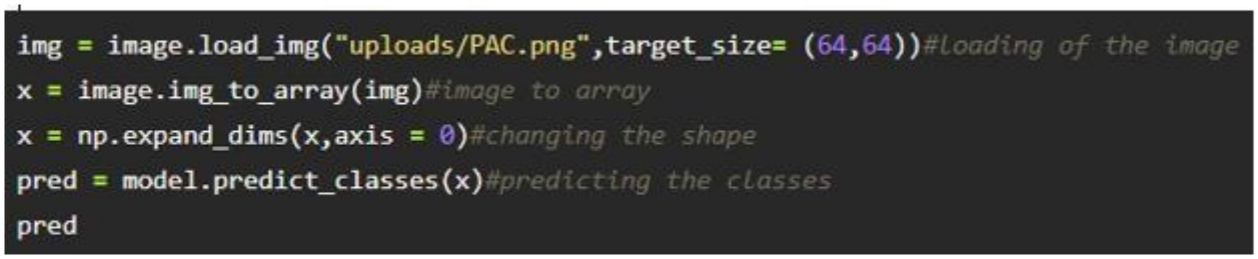
Load necessary libraries, Load the saved model using load_model



```python
from tensorflow.keras.models import load_model
from keras.preprocessing import image
model = load_model("ECG.h5") #loading the model for testing
```

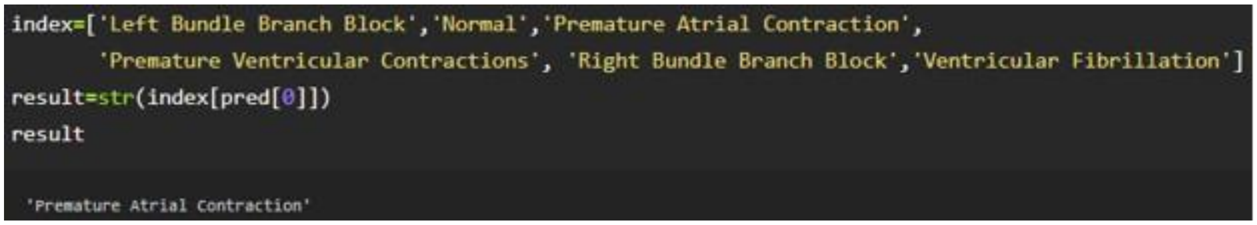
Taking an image as input and checking the results

Note the target size should for the image that is should be the same as the target size that you have used for training



```python
img = image.load_img("uploads/PAC.png",target_size= (64,64))#loading of the image
x = image.img_to_array(img)#image to array
x = np.expand_dims(x,axis = 0)#changing the shape
pred = model.predict_classes(x)#predicting the classes
pred
```

By using the model we are predicting the output for the given input images


```
index=['Left Bundle Branch Block','Normal','Premature Atrial Contraction',
       'Premature Ventricular Contractions', 'Right Bundle Branch Block','Ventricular Fibrillation']
result=str(index[pred[0]])
result

 'Premature Atrial Contraction'
```

The predicted class index name will be printed here.

Application Building

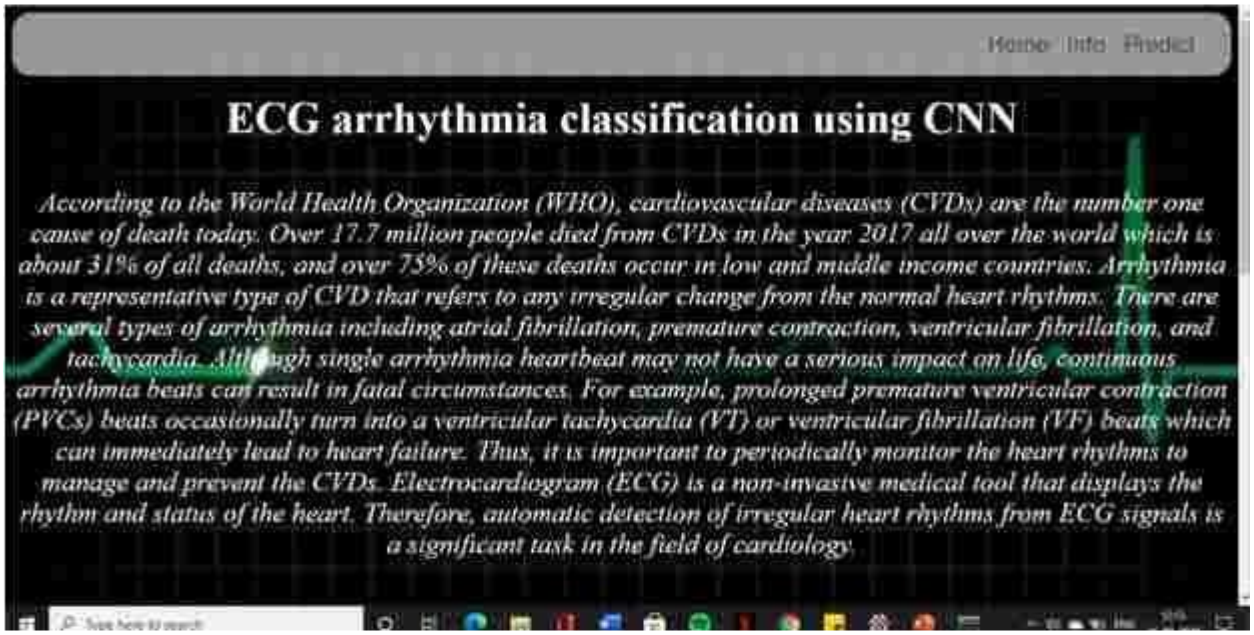
In this section, we will be building a web application that is integrated into the model we built. A UI is provided for the uses where he has uploaded an image. The uploaded image is given to the saved model and prediction is showcased on the UI. This section has the following tasks

- Building HTML Pages
- Building server-side script

Create HTML Files

- We use HTML to create the front end part of the web page.
- Here, we created 4 html pages- about.html, base.html, index6.html, info.html.
- about.html displays the home page.
- Info.html displays all important details to be known about ECG.
- base.html and index6.html accept input from the user and predicts the values.

Your Home page looks like

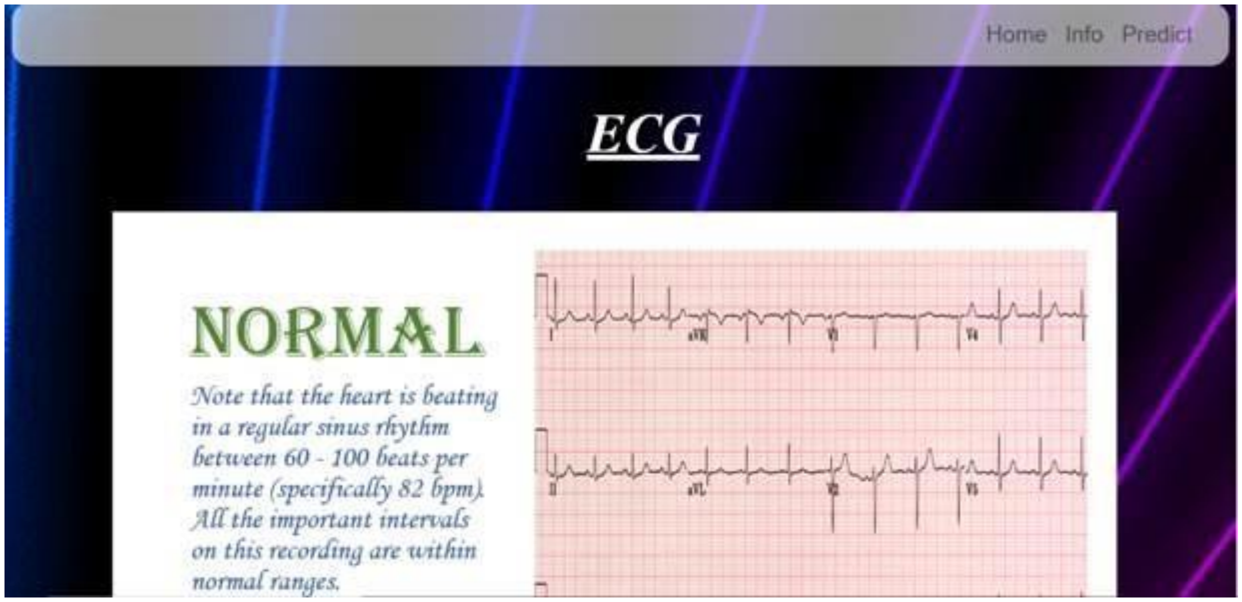


ECG arrhythmia classification using CNN

According to the World Health Organization (WHO), cardiovascular diseases (CVDs) are the number one cause of death today. Over 17.7 million people died from CVDs in the year 2017 all over the world which is about 31% of all deaths, and over 75% of these deaths occur in low and middle income countries. Arrhythmia is a representative type of CVD that refers to any irregular change from the normal heart rhythms. There are several types of arrhythmia including atrial fibrillation, premature contraction, ventricular fibrillation, and tachycardia. Although single arrhythmia heartbeat may not have a serious impact on life, continuous arrhythmia beats can result in fatal circumstances. For example, prolonged premature ventricular contraction (PVCs) beats occasionally turn into a ventricular tachycardia (VT) or ventricular fibrillation (VF) beats which can immediately lead to heart failure. Thus, it is important to periodically monitor the heart rhythms to manage and prevent the CVDs. Electrocardiogram (ECG) is a non-invasive medical tool that displays the rhythm and status of the heart. Therefore, automatic detection of irregular heart rhythms from ECG signals is a significant task in the field of cardiology.

When the "Info" button is clicked, localhost redirects to "info.html"

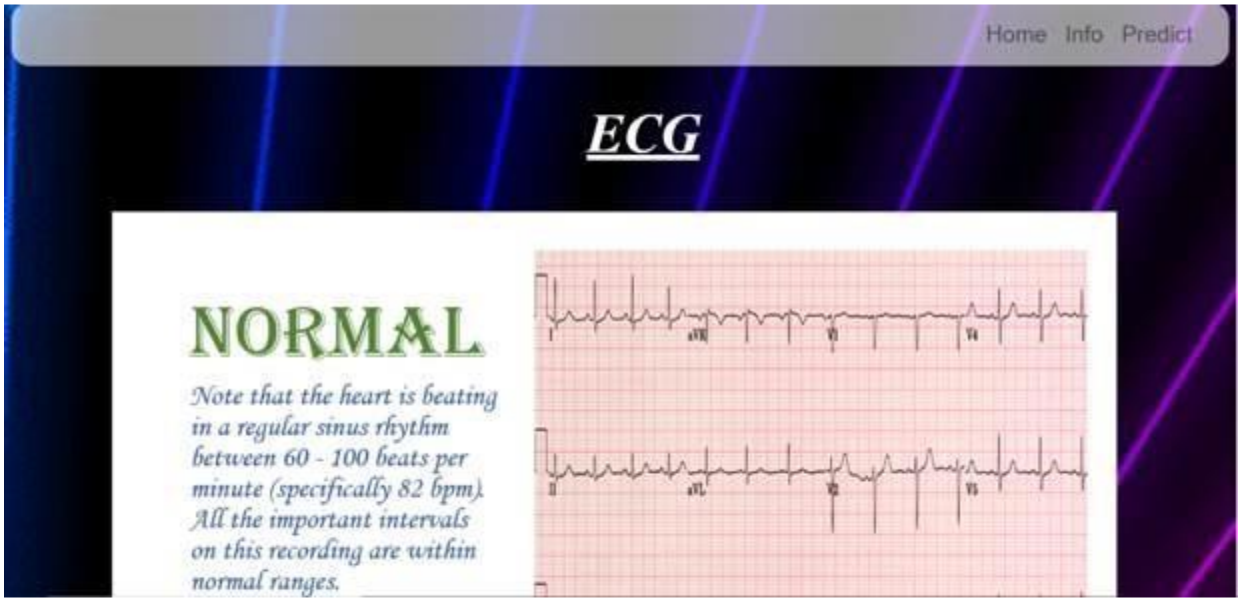
Your info.html Page looks like:



ECG

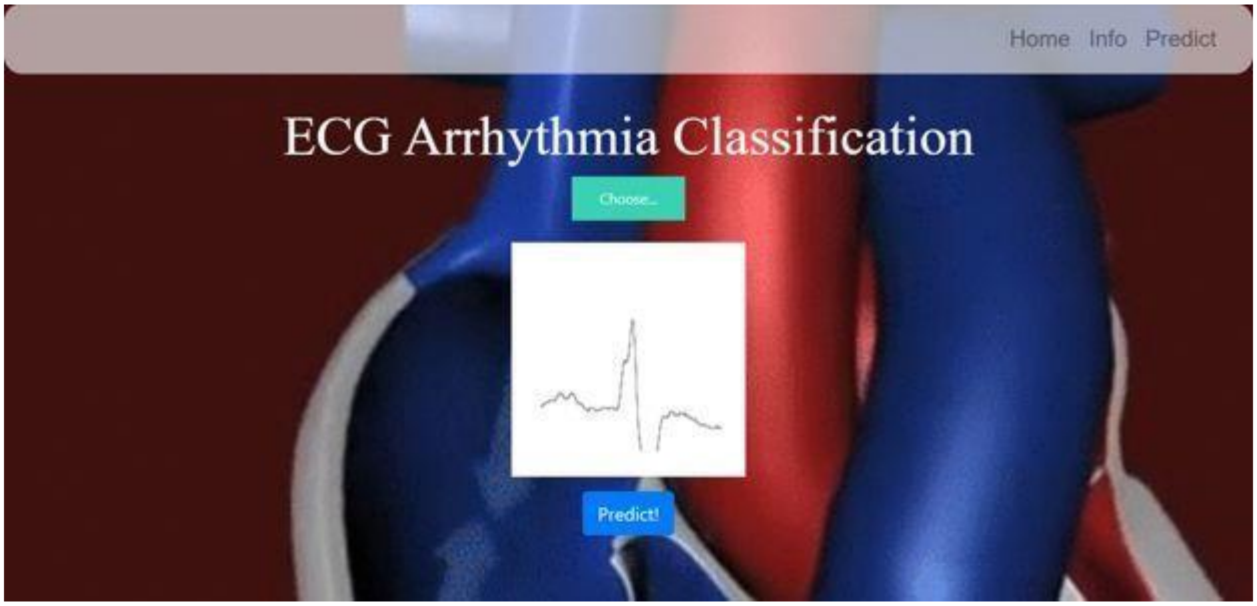
NORMAL

Note that the heart is beating in a regular sinus rhythm between 60 - 100 beats per minute (specifically 82 bpm). All the important intervals on this recording are within normal ranges.



Note that the heart is beating in a regular sinus rhythm between 60 - 100 beats per minute (specifically 82 bpm). All the important intervals on this recording are within normal ranges.

Your prediction page will look like :

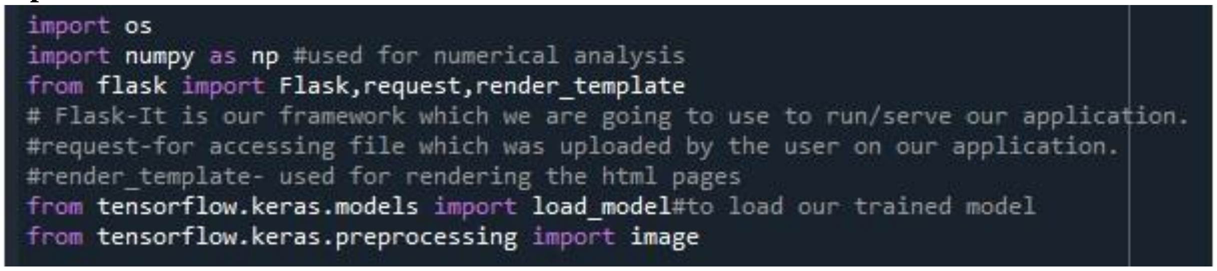


Upload the image and click on the Predict button to view the result on the "base.html" page on the localhost.

Build Python Code

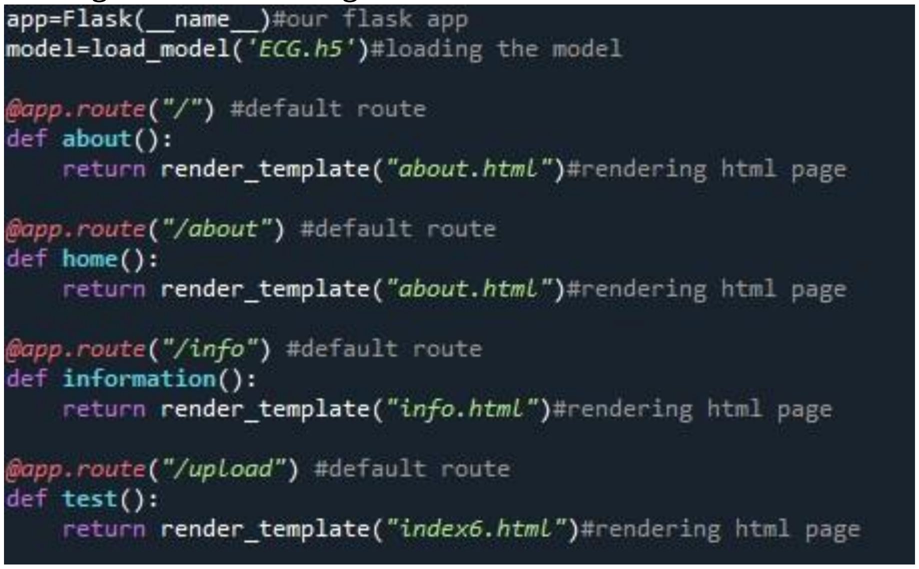
- Let us build the flask file 'app.py' which is a web framework written in python for serverside scripting. Let's see step by step procedure for building the backend application.
- The app starts running when the "__name__" constructor is called in main.
- render_template is used to return HTML file.
- "GET" method is used to take input from the user.
- "POST" method is used to display the output to the user.

Import the libraries



```
import os
import numpy as np #used for numerical analysis
from flask import Flask,request,render_template
# Flask-It is our framework which we are going to use to run/serve our application.
#request-for accessing file which was uploaded by the user on our application.
#render_template- used for rendering the html pages
from tensorflow.keras.models import load_model#to load our trained model
from tensorflow.keras.preprocessing import image
```

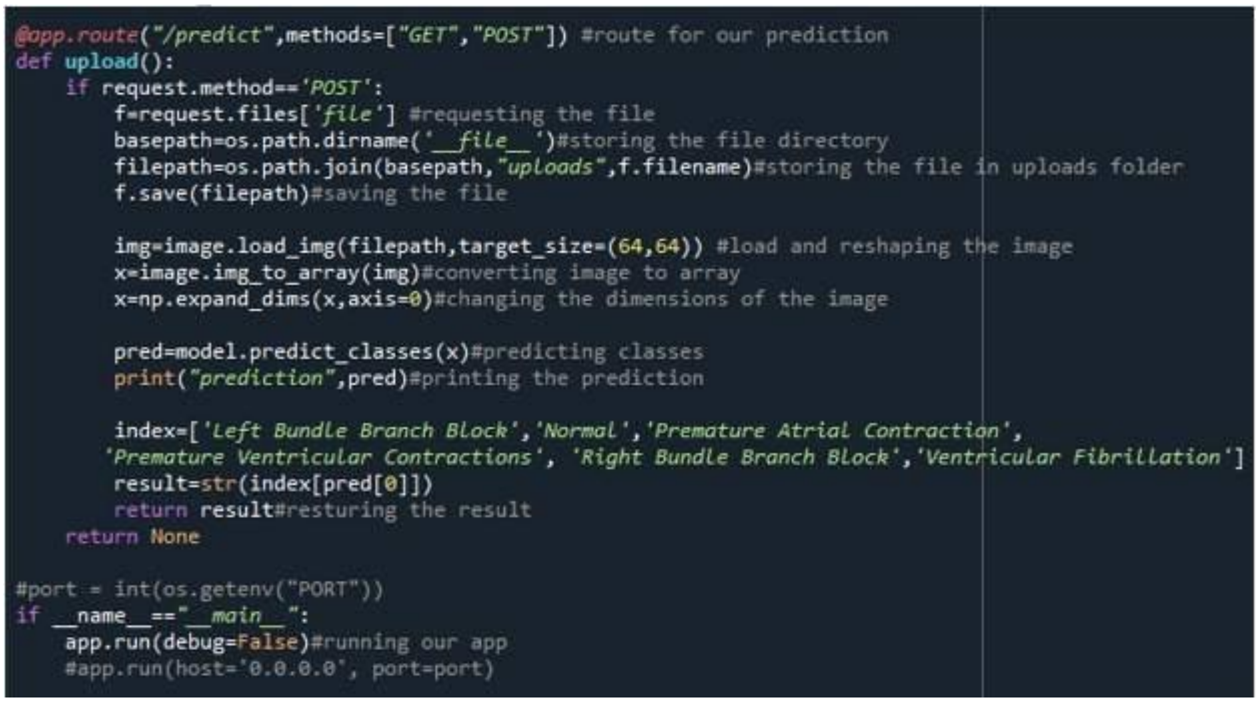
Routing to the HTML Page



```
app=Flask(__name__)#our flask app
model=load_model('ECG.h5')#loading the model

@app.route("/") #default route
def about():
    return render_template("about.html")#rendering html page

@app.route("/about") #default route
def home():
    return render_template("about.html")#rendering html page

@app.route("/info") #default route
def information():
    return render_template("info.html")#rendering html page

@app.route("/upload") #default route
def test():
    return render_template("index6.html")#rendering html page
```

Showcasing prediction on UI When the image is uploaded, it predicts the category of uploaded the image is either 'Left Bundle

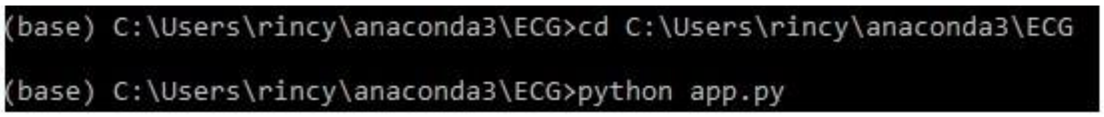
Branch Block', 'Normal', 'Premature Atrial Contraction', 'Premature Ventricular Contractions', 'Right Bundle Branch Block', 'Ventricular Fibrillation'. If the image predicts value as 0, then it is displayed as “Left Bundle Branch”. Similarly, if the predicted value is 1, it displays “Normal” as output and so on.



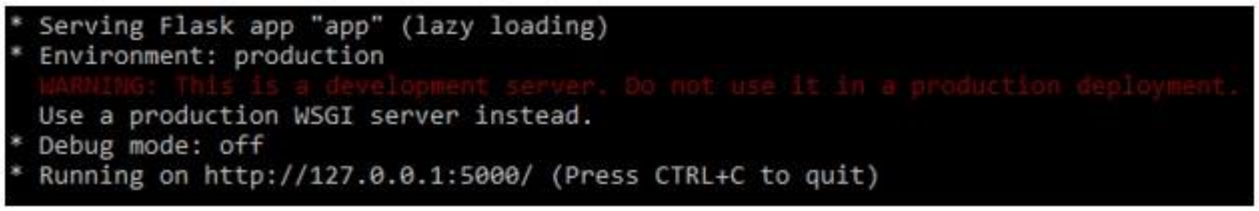
```
@app.route("/predict",methods=["GET","POST"]) #route for our prediction
def upload():
    if request.method=='POST':
        f=request.files['file'] #requesting the file
        basepath=os.path.dirname('__file__')#storing the file directory
        filepath=os.path.join(basepath,"uploads",f.filename)#storing the file in uploads folder
        f.save(filepath)#saving the file

        img=image.load_img(filepath,target_size=(64,64)) #load and reshaping the image
        x=image.img_to_array(img)#converting image to array
        x=np.expand_dims(x,axis=0)#changing the dimensions of the image

        pred=model.predict_classes(x)#predicting classes
        print("prediction",pred)#printing the prediction

        index=['Left Bundle Branch Block','Normal','Premature Atrial Contraction',
       'Premature Ventricular Contractions', 'Right Bundle Branch Block','Ventricular Fibrillation']
        result=str(index[pred[0]])
        return result#resturing the result
    return None

#port = int(os.getenv("PORT"))
if __name__=="__main__":
    app.run(debug=False)#running our app
    #app.run(host='0.0.0.0', port=port)
```

Run The APP

- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type “python app.py” command
- Navigate to the localhost where you can view your web page
- Then it will run on localhost:5000
- Navigate to the localhost (http://127.0.0.1:5000/)where you can view your web page.



```
(base) C:\Users\rincy\anaconda3\ECG>cd C:\Users\rincy\anaconda3\ECG

(base) C:\Users\rincy\anaconda3\ECG>python app.py
```



```
* 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 http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Upload an image and see the predicted result

Train The Model On IBM

In this milestone, you will learn how to build Deep Learning Model Using the IBM cloud.

Register For IBM Cloud

To complete this project you must have an IBM account

IBM Account:

- Please click **here** to register for IBM
- Please click **here** to log in to IBM Account.

Watch the below video to register and login into your IBM account

Train The Model on IBM Watson:

Please watch the below video to train the model on IBM and integrate it with the flask Application.

SOURCE CODE

```
            import numpy as np
            import pandas  as
            pd  from pathlib
            import
            Path import os.path

        import tensorflow as tf
```

```
              dir       =       Path('../input/ecg-image-

              data/ECG_Image_data/train') filepaths =

              list(dir.glob(r'**/*.png'))

              labels           =           list(map(lambda
              os.path.split(os.path.split(x)[0])[1], file paths))

        filepaths     =  pd.Series(filepaths,
              name='Filepath').astype(str)  labels      =
              pd.Series(labels, name='Label')

        dataframe = pd.concat([filepaths , labels] ,
              axis=1) dataframe

              dataframe['Label '].value_counts(
              ) dataframe['Label
'].unique()
samples = [] for category in ['N', 'M', 'Q', 'S', 'V']:
category_slice = dataframe.query("Label == @category")
samples.append(category_slice.sample(2223, random_state=1))

  dataframe_train = pd.concat(samples, axis=0).sample(frac=1.0,
random_st ate=1).reset_index(drop=True)
dataframe_train['Label'].value_counts() dir =
Path('../input/ecgimage-data/ECG_Image_data/train')

  filepaths =
list(dir.glob(r'F/*.png'))
labels = list(map(lambda x:
os.path.split(os.path.split(x)[0])[1], file paths))


filepaths  =  pd.Series(filepaths, name='Filepath').astype(str)
labels = pd.Series(labels, name='Label')


F = pd.concat([filepaths , labels] , axis=1) F %%time dir
= Path('../input/ecg-imagedata/ECG_Image_data/test')

  filepaths =
list(dir.glob(r'**/*.png'))
```

```
labels = list(map(lambda x:
os.path.split(os.path.split(x)[0])[1], file paths))


filepaths  =  pd.Series(filepaths, name='Filepath').astype(str)
labels = pd.Series(labels, name='Label')

  dataframe_test = pd.concat([filepaths , labels] ,
axis=1) dataframe_test
```

Preprocessing of code

Model summary

Html coding

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Aroma Shop - Home</title>

<link rel="icon" href="static/img/Fevicon.png" type="image/png">

<link rel="stylesheet" href="static/vendors/bootstrap/bootstrap.min.css">

<link rel="stylesheet" href="static/vendors/fontawesome/css/all.min.css">

<link rel="stylesheet" href="static/vendors/themify-icons/themify-icons.css">

<link rel="stylesheet" href="static/vendors/nice-select/nice-select.css">

<link rel="stylesheet" href="static/vendors/owl-carousel/owl.theme.default.min.css"> <link
rel="stylesheet" href="static/vendors/owl-carousel/owl.carousel.min.css">


<link rel="stylesheet" href="static/css/style.css">

</head>

<body>
```

```
  <!--================ Start Header Menu Area =================--> <header
            class="header_area">
<div class="main_menu">
<nav class="navbar navbar-expand-lg navbar-light">
<div class="container">
            <a class="navbar-brand logo_h" href="index.html"><img src="static/img/logo.png"
  alt=""></a>
            <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent"
ariaexpanded="false" aria-label="Toggle  navigation">
<span class="icon-bar"></span>
<span class="icon-bar"></span>  <span
class="icon-bar"></span>
</button>
<div class="collapse navbar-collapse offset" id="navbarSupportedContent">
<ul class="nav navbar-nav menu_nav ml-auto mr-auto">
                      <li class="nav-item active"><a class="nav-link" href="index.html">Home</a></li>
<li class="nav-item submenu dropdown">
<a href="#" class="nav-link dropdown-toggle" data-toggle="dropdown"
  role="button" aria-haspopup="true" ariaexpanded="false">Shop</a>
<ul class="dropdown-menu">
<li class="nav-item"><a class="nav-link" href="category.html">Shop
Category</a></li>
                        <li class="nav-item"><a class="nav-link" href="single-product.html">Product
Details</a></li>
<li class="nav-item"><a class="nav-link" href="checkout.html">Product
Checkout</a></li>
<li class="nav-item"><a class="nav-link"
href="confirmation.html">Confirmation</a></li>
                    <li class="nav-item"><a class="nav-link" href="cart.html">Shopping Cart</a></li>
</ul>
```

```
</li>

<li class="nav-item submenu dropdown">

<a href="#" class="nav-link dropdown-toggle" data-toggle="dropdown"  role="button"

  aria-haspopup="true" aria-expanded="false">Blog</a>

<ul class="dropdown-menu">

<li class="nav-item"><a class="nav-link" href="blog.html">Blog</a></li> <li class="navitem"><a
class="nav-link" href="single-blog.html">Blog

Details</a></li>

</ul>

 </li>

 <li class="nav-item submenu  dropdown">

<a href="#" class="nav-link dropdown-toggle" data-toggle="dropdown"

  role="button" aria-haspopup="true" ariaexpanded="false">Pages</a>

<ul class="dropdown-menu">

<li class="nav-item"><a class="nav-link" href="sign.html">Login</a></li>

<li class="nav-item"><a class="nav-link" href="/register">Register</a></li>

                              <li class="nav-item"><a class="nav-link" href="tracking-
order.html">Tracking</a></li>

</ul>

</li>

    <li class="nav-item"><a class="nav-link" href="contact.html">Contact</a></li> </ul>


<ul class="nav-shop">

<li class="nav-item"><button><i class="ti-search"></i></button></li>

                 <li class="nav-item"><button><i class="ti-shopping-cart"></i><span class="nav- shop
  circle">3</span></button> </li>

    <li class="nav-item"><a class="button button-header" href="#">Buy Now</a></li> </ul>

</div>

</div>

</nav>
```

```
</div>

</header>

<!--================ End Header Menu Area =================-->


<main class="site-main">


<!--================ Hero banner start =================-->

<section class="hero-banner">

<div class="container">

<div class="row no-gutters align-items-center pt-60px">

<div class="col-5 d-none d-sm-block">

<div class="hero-banner_img">

  <img class="img-fluid" src="static/img/home/hero-banner.png" alt=""> </div>

</div>

<div class="col-sm-7 col-lg-6 offset-lg-1 pl-4 pl-md-5 pl-lg-0">

<div class="hero-banner_content">

<h4>Shop is fun</h4>

<h1>Browse Our Premium Product</h1>

           <p>Us which over of signs divide dominion deep fill bring they're meat beho upon
 own earth without morning over third. Their male dry. They are great appear whose land fly
 grass.</p>
  <a class="button button-hero" href="#">Browse Now</a> </div>

</div>

</div>

</div>

</section>

<!--================ Hero banner start =================-->


<!--================ Hero Carousel start =================-->

<section class="section-margin mt-0">

<div class="owl-carousel owl-theme hero-carousel">
```

```
<div class="hero-carousel_slide">
<img src="img/home/hero-slide1.png" alt="" class="img-fluid">
<a href="#" class="hero-carousel_slideOverlay">
<h3>Wireless Headphone</h3>
<p>Accessories Item</p>
</a>
</div>
<div class="hero-carousel_slide">
<img src="static/img/home/hero-slide2.png" alt="" class="img-fluid">
<a href="#" class="hero-carousel_slideOverlay">
<h3>Wireless Headphone</h3>
<p>Accessories Item</p>
</a>
</div>
<div class="hero-carousel_slide">
<img src="static/img/home/hero-slide3.png" alt="" class="img-fluid">
<a href="#" class="hero-carousel_slideOverlay">
<h3>Wireless Headphone</h3>
<p>Accessories Item</p>
</a>
</div>
</div>
</section>
<!--================ Hero Carousel end =================-->

<!-- ================ trending product section start ================= -->
<section class="section-margin calc-60px">
<div class="container">
```

```
<div class="section-intro pb-60px">  <p>Popular
Item in the market</p>
<h2>Trending <span class="section-intro_style">Product</span></h2>
<div class="row">
<div class="col-md-6 col-lg-4 col-xl-3">
<div class="card text-center card-product">
<div class="card-product_img">
<img class="card-img" src="static/img/product/product1.png" alt="">
<ul class="card-product_imgOverlay">
<li><button><i class="ti-search"></i></button></li>
<li><button><i class="ti-shopping-cart"></i></button></li>
  <li><button><i class="ti-heart"></i></button></li> </ul>
</div>
<div class="card-body">
<p>Accessories</p>
<h4 class="card-product_title"><a href="single-product.html">Quartz Belt
Watch</a></h4>
<p class="card-product_price">$150.00</p>
</div>
</div>
</div>
<div class="col-md-6 col-lg-4 col-xl-3">
<div class="card text-center card-product">
<div class="card-product_img">
<img class="card-img" src="static/img/product/product2.png" alt="">
<ul class="card-product_imgOverlay">
<li><button><i class="ti-search"></i></button></li>
<li><button><i class="ti-shopping-cart"></i></button></li>
  <li><button><i class="ti-heart"></i></button></li> </ul>
```

```
</div>
<div class="card-body">
<p>Beauty</p>
<h4 class="card-product_title"><a href="single-product.html">Women
Freshwash</a></h4>
<p class="card-product_price">$150.00</p>
</div>
</div>
</div>
<div class="col-md-6 col-lg-4 col-xl-3">
<div class="card text-center card-product">
<div class="card-product_img">
<img class="card-img" src="static/img/product/product3.png" alt="">
<ul class="card-product_imgOverlay">
<li><button><i class="ti-search"></i></button></li>
<li><button><i class="ti-shopping-cart"></i></button></li>
    <li><button><i class="ti-heart"></i></button></li> </ul>
</div>
  <div class="card-body"> <p>Decor</p>
Light</a></h4>        _
<p class="card-product_price">$150.00</p>
</div>
</div>
</div>
<div class="col-md-6 col-lg-4 col-xl-3">
<div class="card text-center card-product">
<div class="card-product_img">
<img class="card-img" src="static/img/product/product4.png" alt="">
<ul class="card-product_imgOverlay">
```

```
<li><button><i class="ti-search"></i></button></li>
<li><button><i class="ti-shopping-cart"></i></button></li>
    <li><button><i class="ti-heart"></i></button></li> </ul>
</div>
<div class="card-body">
<p>Decor</p>
<h4 class="card-product_title"><a href="single-product.html">Room Flash
Light</a></h4>
<p class="card-product_price">$150.00</p>
</div>
</div>
</div>
<div class="col-md-6 col-lg-4 col-xl-3">
<div class="card text-center card-product">
<div class="card-product_img">
<img class="card-img" src="static/img/product/product5.png" alt="">
<ul class="card-product_imgOverlay">
<li><button><i class="ti-search"></i></button></li>
<li><button><i class="ti-shopping-cart"></i></button></li>
    <li><button><i class="ti-heart"></i></button></li> </ul>
</div>
<div class="card-body">
<p>Accessories</p>
<h4 class="card-product_title"><a href="single-product.html">Man Office  Bag</a></h4>
<p class="card-product_price">$150.00</p>
</div>
</div>
</div>
<div class="col-md-6 col-lg-4 col-xl-3">
```

```
<div class="card text-center card-product">

<div class="card-product_img">

<img class="card-img" src="static/img/product/product6.png" alt="">

<ul class="card-product_imgOverlay">

<li><button><i class="ti-search"></i></button></li>

<li><button><i class="ti-shopping-cart"></i></button></li>

<li><button><i class="ti-heart"></i></button></li>

<!-- ================ offer section start ================= -->

      <section class="offer" id="parallax-1" data-anchor-target="#parallax-1" data-300-
  top="background-position: 20px 30px" data-top-bottom="background-position: 0 20px"> <div
  class="container">

<div class="row">

<div class="col-xl-5">

<div class="offer_content text-center">

<h3>Up To 50% Off</h3>

<h4>Winter Sale</h4>

<p>Him she'd let them sixth saw light</p>

    <a class="button button--active mt-3 mt-xl-4" href="#">Shop Now</a> </div>

</div>

</div>

</div>

</section>

<!-- ================ offer section end ================= -->


<!-- ================ Best Selling item carousel ================= -->

<section class="section-margin calc-60px">

<div class="container">

<div class="section-intro pb-60px">

<p>Popular Item in the market</p>

    <h2>Best <span class="section-intro_style">Sellers</span></h2> </div>
```

```
<div class="owl-carousel owl-theme" id="bestSellerCarousel">
<div class="card text-center card-product">
<div class="card-product_img">
<img class="img-fluid" src="static/img/product/product1.png" alt="">
<ul class="card-product_imgOverlay">
<li><button><i class="ti-search"></i></button></li>
<li><button><i class="ti-shopping-cart"></i></button></li>
  <li><button><i class="ti-heart"></i></button></li> </ul>
</div>
<div class="card-body">
<p>Accessories</p>
<h4 class="card-product_title"><a href="single-product.html">Quartz Belt
Watch</a></h4>
  <p class="card-product_price">$150.00</p> </div>
</div>

<div class="card text-center card-product">
<div class="card-product_img">
<img class="img-fluid" src="static/img/product/product2.png" alt="">
<ul class="card-product_imgOverlay">
<li><button><i class="ti-search"></i></button></li>
<li><button><i class="ti-shopping-cart"></i></button></li>



           <li><button><i class="ti-heart"></i></button></li>
<li><button><i class="ti-shopping-cart"></i></button></li>
  <li><button><i class="ti-heart"></i></button></li> </ul>
</div>
<div class="card-body">
```

```
<p>Decor</p>
            <h4 class="card-product_title"><a href="single-product.html">Room Flash
 Light</a></h4>
   <p class="card-product_price">$150.00</p> </div>
</div>
</div>
</div>
</section>
              <!-- ================ Best Selling item carousel end ================= -->
```

Conclusions

This project is designed the In using the MIT-BIH arrhythmia database, have proposed a system for the automatic processing of the ECG for the classification of arrhythmia images. The database of MIT-BIH is processed visually and a waveform detection method is proposed for detecting the QRS waveform. A CNN model was built to train and classify the ECG images. Experimental results show that according to the ANSI/AAMI EC57 evaluation criteria, The accuracy rate of a ventricular ectopic beat can reach 95.9% and the sensitivity evaluation is 93.0%. For the supraventricular ectopic beat class, the accuracy rate is 93.2% and the sensitivity evaluation is 81.3%

ADVANTAGES:

Your healthcare provider will be able to explain your results to you such as,

- Heart arrhythmias, such as premature ventricular complexes or atrial fibrillation
- Whether you have conduction abnormalities, which result from issues regarding how the electrical impulse spreads across the heart
  (such as with a bundle branch block)
- Signs of an ongoing or a prior myocardial infarction (heart attack)
- Whether you have signs of severe coronary artery disease (CAD), such as stable angina or unstable angina
- If your heart muscle has become abnormally thickened, as in hypertrophic cardiomyopathy
- Signs of congenital electrical abnormalities, such as Brugada syndrome

Future Scope:

In future work, we designed real-time implementation of processor design of

arrhythmia classification.

GITHUB LINK: **https://github.com/IBM-EPBL/IBM-Project-6597-1658832916**