CLASSIFICATION OF ARRHYTHMIA USING DEEP LEARNING WITH 2-D ECG SPECTRAL IMAGE PRESENTATION

TEAM ID : PNT2022TMID31222

TEAM MEMBERS: 1. PRASANTH A

2. BHARATH S
3. NAVEENSRIDHAR K
4. RITHAN S
5. PRASANTH R

1. INTRODUCTION

1.1 Project Overview

In recent years, cardiovascular diseases (CVDs) are the major cause of death. Over 17.7 million people died from CVDs in the year 2017 all over the world. 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. In this project, we build an effective electrocardiogram (ECG) arrhythmia classification method using a convolutional neural network (CNN), in which we classify ECG into six categories, one being normal and the other five 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.

1.2 Purpose

Deep learning techniques surpassed traditional techniques especially in pattern recognition.It is mostly applied to analyze visual images. So these techniques can be used to predict the types of arrhythmia using ECG images.We can create an application to upload and display the results. It makes it easier for everyone.

2. LITERATURE SURVEY

S. NO	PAPER	AUTHOR	YEAR	DESCRIPTION
1	Classification of Arrhythmia in Heartbeat Detection Using Deep Learning	Wusat Ullah,Imran Siddique , Rana Muhammad Zulqarnain ,Mohamm ad Mahtab Alam , Irfan Ahmad, and Usman Ahmad Raza.	2021	Aims to apply deep learning techniques on the publicly available dataset to classify arrhythmia. The system combines three different types of information: RR intervals,signal morphology, and higher-level statistical data. It is concluded that fuzzy based technology is successful in the analysis of computerized ECG but needs more research
2	Arrhythmia Classification Techniques Using Deep Neural Network	Ali Haider Khan ,Muzammil Hussain ,and Muhammad Kamran Malik	2021	The automated screening of arrhythmia classification using ECG beats is developed for ages. The deep learning based automated Arrhythmia Classification techniques are developed with high accuracy. The primary concerns that affect the success of the Developed arrhythmia detection systems are (i) manual features selection, (ii) techniques used for features extraction, and (iii) algorithm used for classification and the most important is the use of imbalanced data for classification.

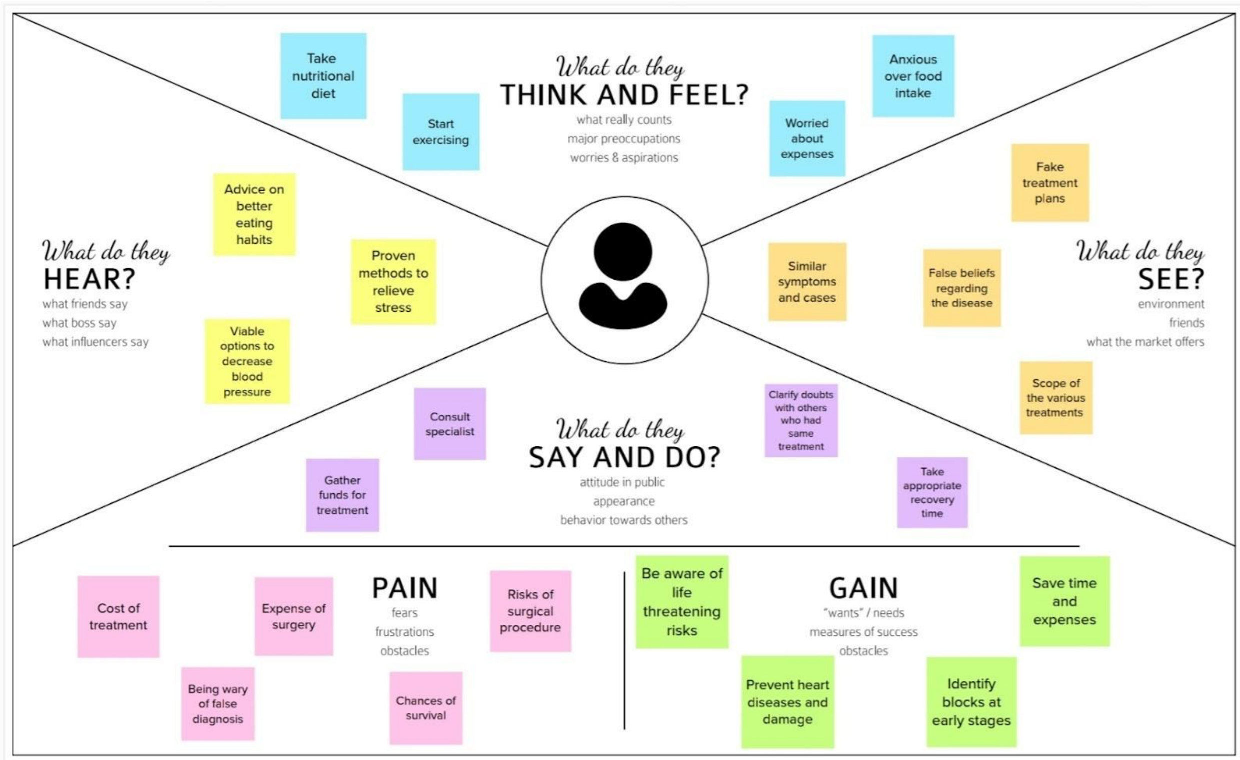
3	Classification of Arrhythmia by Using Deep Learning with 2- D ECG Spectral Image Representation	Amin Ullah, Syed Anwar, Muhammad Bilal, Raja Majid Mehmood	2020	Proposal of two-dimensional (2-D)convolutional neural network (CNN)model for the classification of ECGsignals into eight classes; namely,normal beat, premature ventricular contraction beat, paced beat, right bundle branch block beat, left bundle branch block beat, atrial premature contraction beat, ventricular flutter wave beat, and ventricular escape beat. The one-dimensional ECG time series signals are transformed into 2-Dspectrograms through a short-time Fouriertransform. The 2-D CNN model consisting of four convolutional layers and four pooling layers is designed for extracting robust features from the input spectrograms.

4	A deep convolutional neural network model to classify heartbeats	Rajendra Acharya, Shu Lih Oh, Yuki Hagiwara, Jen Hong Tan, Muhammad Adam	2017	The basis of arrhythmia diagnosis is the identification of normal versus abnormal individual heart beats, and their correct classification into different diagnoses,based on ECG morphology. Heartbeat scan be subdivided into five categories namely non-ectopic, supraventricular ventricular ectopic, ventricular ectopic, fusion, andun-known beats. It is challenging and time-consuming to distinguish these heartbeats on ECG as these signals are typically corrupted by noise. We developed a 9-layer deep convolutional neural network (CNN) to automatically identify 5 different categories of heartbeats in ECG signals. Our experiment was conducted in original and noise attenuated sets of ECG signals derived from a publicly available database.

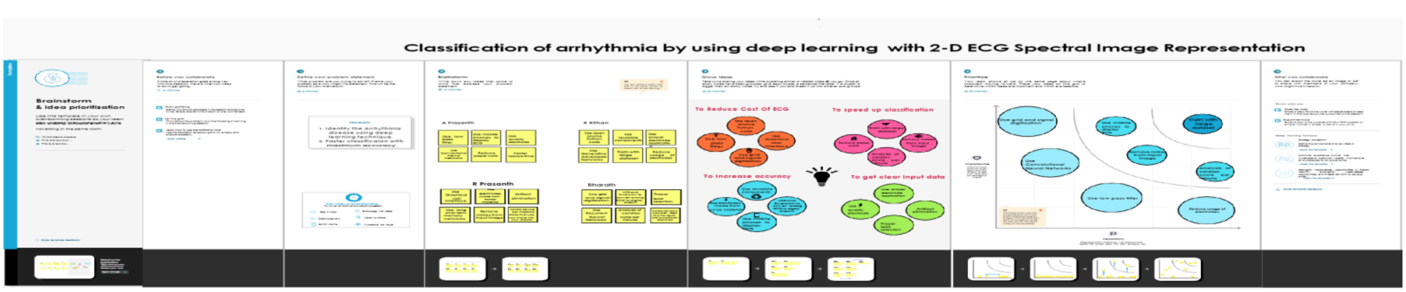
5	Cardiac arrhythmia detection using deep Learning	Ali Isina, Selen Ozdalili	2017	An electrocardiogram is an important diagnostic tool for the assessment of cardiac arrhythmias in clinical routine. A deep learning framework previously trained on a general image data set is transferred to carry out automatic ECGarrhythmia diagnostics by classifying patient ECG's into corresponding cardiac conditions. Transferred deep convolutional neural networks are used as feature extractor and the extracted features are fed into a simple backpropagation neural network to carry out the final classification.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



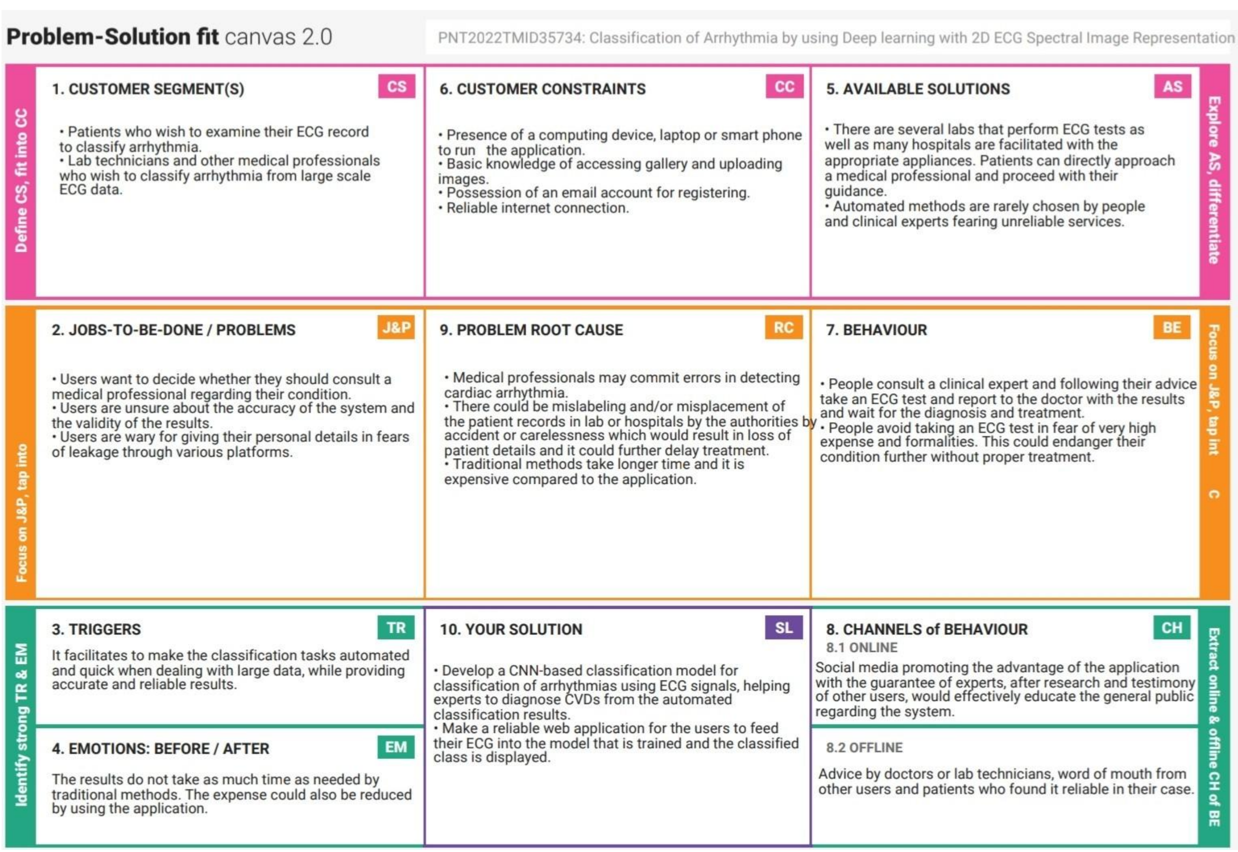
3.2 Ideation & Brainstorming



3.3 Proposed Solution

The automatic classification of arrhythmia using the ECG signal in a supervised way is proposed using a CNN-based model. 2. The type of arrhythmia present is identified by appropriate labeling on the ECG data utilized in the study.1 3. Expert cardiologists assigned these designations, which are then used for supervised training.The arrhythmia class label was applied to the associated spectrogram picture representation for each heartbeat segment.Comparative analysis of various CNN models like ResNet, Xception, VGG19 and a custom model will be performed before deploying the model with the best performance in the web application. It is extremely difficult to predict abnormal heart rates interactively. As a result, an automated system capable of identifying discrete abnormal heartbeats from a large amount of ECG data will promote safe and independent living among the public which will make them more self-reliable.

3.4 Problem Solution Fit



4.REQUIREMENT ANALYSIS

4.1 Functional Requirements

SNO	REQUIREMENTS	SUB REQUIREMENTS
1	User Input	Upload ECG image as JPEG/PNG
2	Process Image	The trained CNN model processes the input image to classify the Arrhythmia.
3	Generate Result	Display the classification result on the screen.

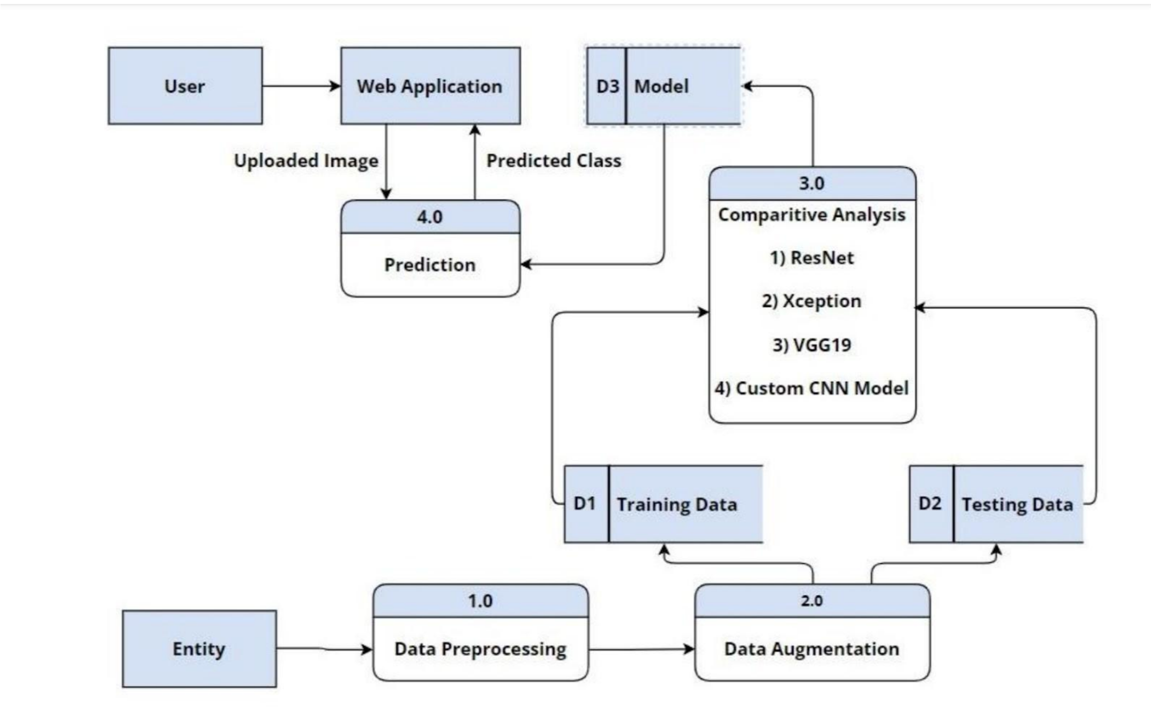
4.2 Non-Functional Requirements

SNO	NON-FUNCTIONAL REQUIREMENTS	DESCRIPTION
1	Usability	It is a user-friendly application which allows users to upload ECG images to classify Arrhythmia.
2	Security	Data is not used for any other purposes other than processing. Only users can view the results of the uploaded image.
3	Reliability	The application is defect free, deployed with a high accuracy CNN model which provides the correct prediction for the given input.
4	Performance	High accuracy models are used for classification thereby increasing the performance of the

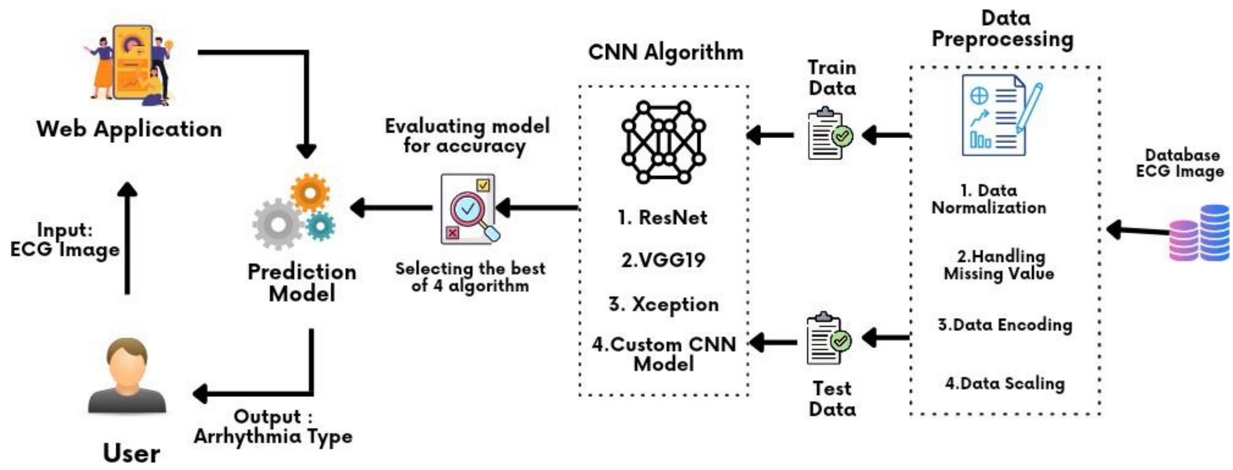
		application
5	Availability	The application can be accessed anytime from anywhere with an internet connection.
6	Scalability	The system must be scalable to process multiple images. Multiple users must be able to access the system simultaneously without traffic.

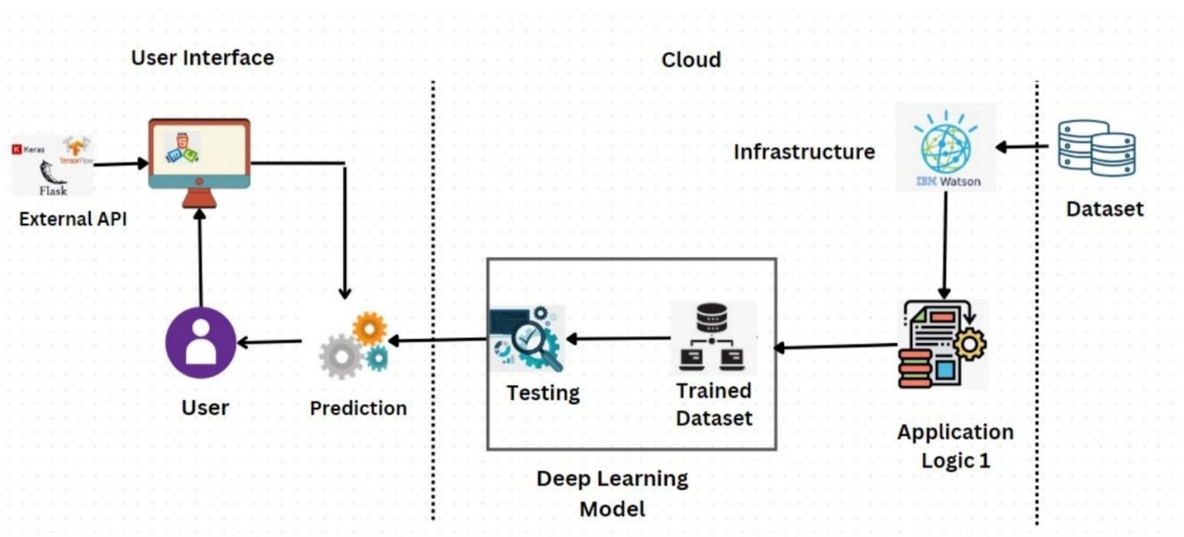
5.PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture





5.3 User Stories

User Type	Functional Requirement	User Story No	User Story	Acceptance Criteria	Priority	Release
Mobile / Web User	Image Upload	USN-1	As a user, I can upload an ECG image in the application.	I can upload an ECG image and click on ‘Upload’ to get the result.	High	Sprint 2
	Prediction Result	USN-2	As a user, I can view the predicted class in the application.	I can view the result of my uploaded image	High	Sprint 2
Customer care executive	Support	USN-1	As a customer care executive, I can provide support and solve the issues that users face with the application.	I can provide support and solve issues w.r.t the application	Medium	Sprint 3

Administrator	Application Maintenance	USN-1	As an administrator, I can upgrade or update the application whenever a bug is discovered, or a newer version is developed.	I can update or upgrade the application	High	Sprint 1
	Application Security	USN-2	As an administrator, I can implement security measures and make necessary changes.	I can make the application more secure.	High	Sprint 1

6.PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement	User Story NO	User Story	Story Points	Priority	Team Members
Sprint1	Dashboard	USN1	As a user, based on my requirement I can navigate through the	2	Low	Prasanth A

			dashboard.			
Sprint1	Pre-process the dataset	USN2	The image dataset is pre-processed.	4	Medium	Prasanth A Prasanth R
Sprint2	Upload images and display output page	USN1	As a user, I should be able to upload the image of ECG and get the output	6	High	Rithan R
Sprint2	Train the pre-trained model	USN2	The pre-trained models Inception, ResNet and AlexNet are trained on the preprocessed dataset	6	High	Rithan R Naveensridhar K
Sprint2	Build Python Code	USN3	Build the flask file 'app.py' which is a web framework written in python for server side scripting	8	High	Prasanth A
Sprint3	Train custom CNN model	USN1	Train the model with the image dataset. fit_generator functions are used to train a deep learning neural network	10	High	Prasanth A Rithan Bharath Naveensridhar Prasanth R

Sprint3	Test the Models	USN2	Test the model through Loaded necessary libraries, the model is evaluated for accurate results.	10	Medium	Prasanth A Rithan Bharath Naveensridhar Prasanth R
Sprint4	Register in IBM cloud	USN1	Register in IBM Cloud	10	Medium	Prasanth A Rithan Bharath Naveensridhar Prasanth R
Sprint4	Train the model on IBM	USN2	Train the model on IBM	10	High	Prasanth A Rithan Bharath Naveensridhar Prasanth R

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint start date	Sprint End date	Story Points completed	Sprint Release Date
Sprint 1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint 2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint 3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint 4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

7.CODING & SOLUTIONING

7.1 Download dataset

```
1 !unzip 'IBM Dataset.zip'
```

```
Streaming output truncated to the last 5000 lines.
  inflating: data/train/Premature Ventricular Contractions/fig_1532.png
  inflating: data/train/Premature Ventricular Contractions/fig_1533.png
  inflating: data/train/Premature Ventricular Contractions/fig_1534.png
  inflating: data/train/Premature Ventricular Contractions/fig_1535.png
  inflating: data/train/Premature Ventricular Contractions/fig_1536.png
  inflating: data/train/Premature Ventricular Contractions/fig_1537.png
  inflating: data/train/Premature Ventricular Contractions/fig_1538.png
  inflating: data/train/Premature Ventricular Contractions/fig_1539.png
  inflating: data/train/Premature Ventricular Contractions/fig_154.png
  inflating: data/train/Premature Ventricular Contractions/fig_1540.png
  inflating: data/train/Premature Ventricular Contractions/fig_1541.png
  inflating: data/train/Premature Ventricular Contractions/fig_1542.png
  inflating: data/train/Premature Ventricular Contractions/fig_1543.png
  inflating: data/train/Premature Ventricular Contractions/fig_1544.png
  inflating: data/train/Premature Ventricular Contractions/fig_1545.png
  inflating: data/train/Premature Ventricular Contractions/fig_1546.png
  inflating: data/train/Premature Ventricular Contractions/fig_1547.png
  inflating: data/train/Premature Ventricular Contractions/fig_1548.png
  inflating: data/train/Premature Ventricular Contractions/fig_1549.png
  inflating: data/train/Premature Ventricular Contractions/fig_155.png
  inflating: data/train/Premature Ventricular Contractions/fig_1550.png
  inflating: data/train/Premature Ventricular Contractions/fig_1551.png
  inflating: data/train/Premature Ventricular Contractions/fig_1552.png
  inflating: data/train/Premature Ventricular Contractions/fig_1553.png
```

7.2 Image Preprocessing

```
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True,)

test_datagen = ImageDataGenerator(rescale=1./255)
```

7.3 Import Libraries

```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.utils import resample
from sklearn.model_selection import train_test_split

import tensorflow as tf

import keras
from keras.preprocessing.image import ImageDataGenerator
from keras.applications import VGG19
from keras.models import Model, Sequential
from keras.layers import Input, GlobalAveragePooling2D, BatchNormalization, Dropout, Dense
from keras.layers import MaxPooling2D, Flatten, Conv2D, Softmax
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.models import load_model
from keras_preprocessing import image
```

7.4 Configure learning process

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

7.5 Adding layers

```python
model = Sequential()
model.add(Conv2D(32, (3,3), padding='same', activation='relu', input_shape=(128, 128, 3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(32, (3,3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(35))
model.add(Dense(6, activation='softmax'))

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

model.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_11 (Conv2D)          (None, 128, 128, 32)      896

 max_pooling2d_5 (MaxPooling  (None, 64, 64, 32)       0
 2D)

 conv2d_12 (Conv2D)          (None, 64, 64, 32)        9248

 max_pooling2d_6 (MaxPooling  (None, 32, 32, 32)       0
 2D)

 flatten_3 (Flatten)         (None, 32768)             0

 dropout_3 (Dropout)         (None, 32768)             0

 dense_5 (Dense)             (None, 35)                1146915

 dense_6 (Dense)             (None, 6)                 216

=================================================================
Total params: 1,157,275
Trainable params: 1,157,275
Non-trainable params: 0
_________________________________________________________________
```

7.6 Train Model

```
x_train = train_datagen.flow_from_directory(directory=r'/content/data/train',
                                            target_size=(128,128),
                                            batch_size=32,
                                            class_mode='categorical',
                                            classes=['Left Bundle Branch Block', 'Normal',
                                                     'Premature Atrial Contraction',
                                                     'Premature Ventricular Contractions',
                                                     'Right Bundle Branch Block', 'Ventricular Fibrillation'],
                                            )
x_test = test_datagen.flow_from_directory(directory=r'/content/data/test',
                                          target_size=(128,128),
                                          batch_size=32,
                                          class_mode='categorical',
                                          classes=['Left Bundle Branch Block', 'Normal',
                                                   'Premature Atrial Contraction',
                                                   'Premature Ventricular Contractions',
                                                   'Right Bundle Branch Block', 'Ventricular Fibrillation'],
                                          )
```

7.7 Save Model

```
model.save('ECG (CNN).h5')
```

```
model = load_model("ECG (CNN).h5")
img = image.load_img("/content/data/test/Premature Atrial Contraction/fig_107.png", target_size=(128, 128))

index = ['Left Bundle Branch Block', 'Normal','Premature Atrial Contraction',
         'Premature Ventricular Contractions','Right Bundle Branch Block', 'Ventricular Fibrillation']

```

7.8 Test Model

```
model = load_model("ECG (CNN).h5")
img = image.load_img("/content/data/test/Premature Atrial Contraction/fig_107.png", target_size=(128, 128))

index = ['Left Bundle Branch Block', 'Normal','Premature Atrial Contraction',
         'Premature Ventricular Contractions','Right Bundle Branch Block', 'Ventricular Fibrillation']

x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
pred = model.predict(x)
result = str(index[np.where(pred[0]==1)[0][0]])
print("Predicted Class: ", result)
```

```
1/1 [==============================] - 0s 211ms/step
Predicted Class:  Premature Atrial Contraction
```

7.8 flask python code

```
import os
import numpy as np
from flask import Flask,request,render_template
from keras.models import load_model
import tensorflow as tf
from PIL Image

app=Flask(__name__)
model=load_model('ECG.h5')
```

```
@app.route("/")
def about():
    return render_template("home.html")

@app.route("/home")
def home():
    return render_template("home.html")

@app.route("/info")
def info():
    return render_template("info.html")

@app.route("/guide")
def guide():
    return render_template("guide.html")

@app.route("/predict")
def test():
    return render_template("predict.html")

@app.route("/predict",methods=["GET","POST"])
def upload():
    if request.method=='POST':
        f=request.files['file']
        basepath=os.path.dirname('__file__')
        filepath=os.path.join(basepath,"uploads",f.filename)
        f.save(filepath)

        img=tf.keras.utils.load_img(filepath,target_size=(128,128))
        x=tf.keras.utils.img_to_array(img)
        x=np.expand_dims(x,axis=0)
```

```
        pred=model.predict(x)
        y_pred = np.argmax(pred)
        print("prediction",y_pred)

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

        return result
    return None

if __name__=="__main__":
    app.run(debug=False)
```

7.9 User Interface

7.9.1 Home.html

```
<!DOCTYPE html>
<html>

<head>

<title>Home</title>

<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="../static/css/index.css">
<link href="{{ url_for('static', filename='css/index.css') }}" rel="stylesheet">

</head>
```

```
<body>

  <div class="navbar">
    <a href="/predict" >PREDICT</a>
    <a href="/info">INFO</a>
  </div>

  <div>
    <center><h2 class="header">ARRHYTHMIA PREDICTION</h2></center>
    <br>
    <center>
      <b class="pd">
        <font color = "#ffec78" size="13" font-family = "Helvetica">
          ECG arrhythmia classification using CNN
        </font>
      </b>
    </center>
  </div> <br>

  <center>
    <p>
      <font color = "white">
        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.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.
            </font>
        </p>
    </center>

</body>
</html>
```

7.9.2 Info.html

```
<!DOCTYPE html>
<html>
<head>
<style>
img{
width:20%;
height:10%;
padding:20px;
margin-top:5px;
}
</style>
<title>
    Info
</title>

<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="../static/css/index.css">
<link href="{{ url_for('static', filename='css/index.css') }}" rel="stylesheet">

</head>

<body>
```

```
<div class="navbar">
    <a href="/predict" >PREDICT</a>
    <a href="/guide">ECG</a>
    <a href="/home">HOME</a><br>
</div>

<div class="container" >
     <p>
         <h2>
             Types of Arrhythmia
        </h2>

    <b>
        <font color = "#bb8206">
            Supraventricular arrhythmias :
        </font>
    </b>
        Arrhythmias that begin in the atria (the heart's upper chambers). "Supra" means
above.
        "Ventricular" refers to the lower chambers of the heart or ventricles. <br>
    <b>
        <font color = "#bb8206">
            Ventricular arrhythmias :
        </font>
    </b>
        Arrhythmias that begin in the ventricles (the heart's lower chambers).<br>
    <b>
        <font color = "#bb8206">
            Bradyarrhythmias :
        </font>
    </b>
        Slow heart rhythms that may be caused by disease in the heart's conduction
```

system,
such as the sinoatrial (SA) node, atrioventricular (AV) node or HIS-Purkinje network.
</p>

<h2>
Symptoms of Arrhythmia
</h2>

<ol style="display:inline-table;">

A feeling of skipped heartbeat or that your heart is “running away,” fluttering or doing "flip-flops.

Pounding in your chest.

Dizziness or feeling lightheaded.

Shortness of breath.

Chest discomfort.

Weakness or fatigue (feeling very tired).

Weakening of the heart muscle or low ejection fraction.

<h2>

Causes of Arrhythmia

</h2>

<ol style="display:inline-table;">

Coronary artery disease.

Irritable tissue in the heart (due to genetic or acquired causes).

High blood pressure.

Changes in the heart muscle (cardiomyopathy).

Valve disorders.

Electrolyte imbalances in your blood, such as sodium or potassium imbalances.

Injury from a heart attack.

The healing process after heart surgery.


```
            <li>
                Other medical conditions.
            </li>
        </ol>

    </div>
</body>

</html>
```

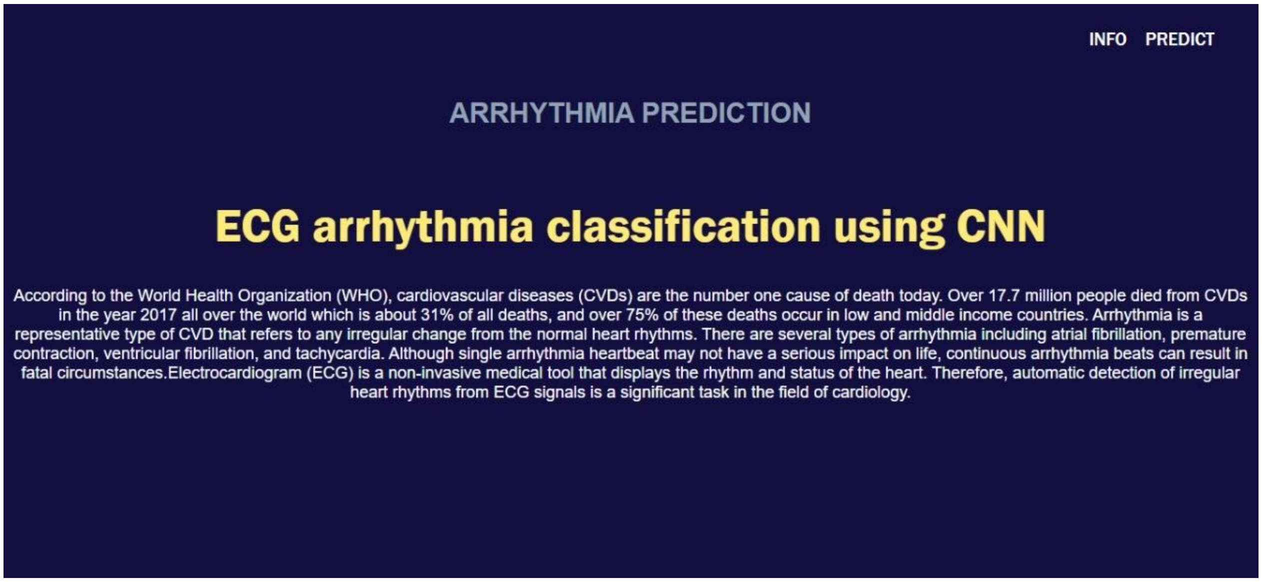
7.9.3 Predict_base.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">

        <title>Predict</title>

        <meta name="viewport" content="width=device-width, initial-scale=1">
        <link rel="stylesheet" href="../static/css/index.css">

        <link href="{{ url_for('static', filename='css/index.css') }}" rel="stylesheet">


    </head>

    <body>
        <div class="navbar">
            <a href="/guide">ECG</a>
```

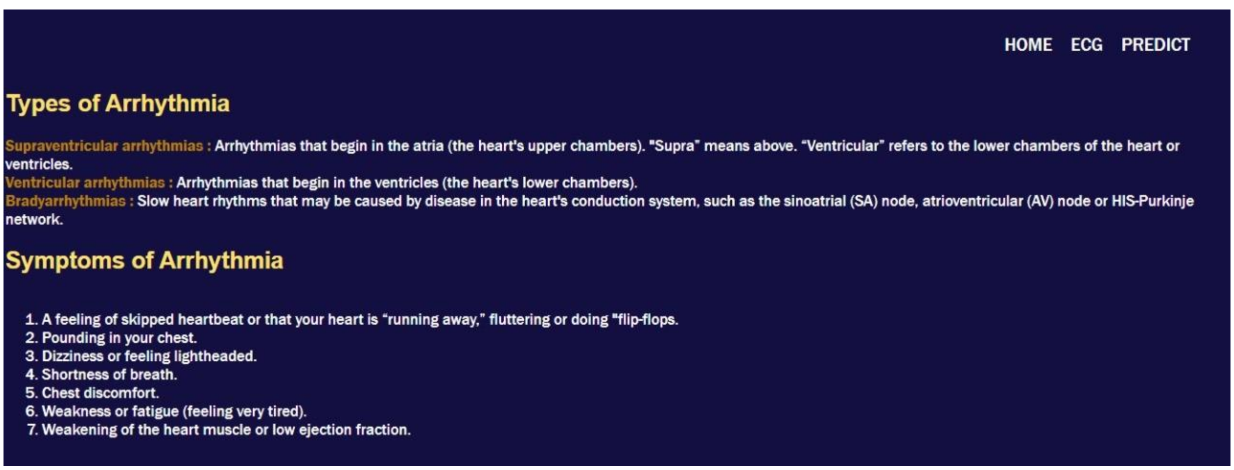
```
            <a href="/info">INFO</a>
            <a href="/home">HOME</a><br>
        </div>

        <div class="container">
           <center>
               <div id="content" style="margin-top:2em">
                    {% block content %}{% endblock %}
                </div>
            </center>
        </div>

    </body>

    <footer>
        <script src="{{ url_for('static', filename='js/main.js') }}"
type="text/javascript"></script>
    </footer>
</html>
```

7.10 Run Application

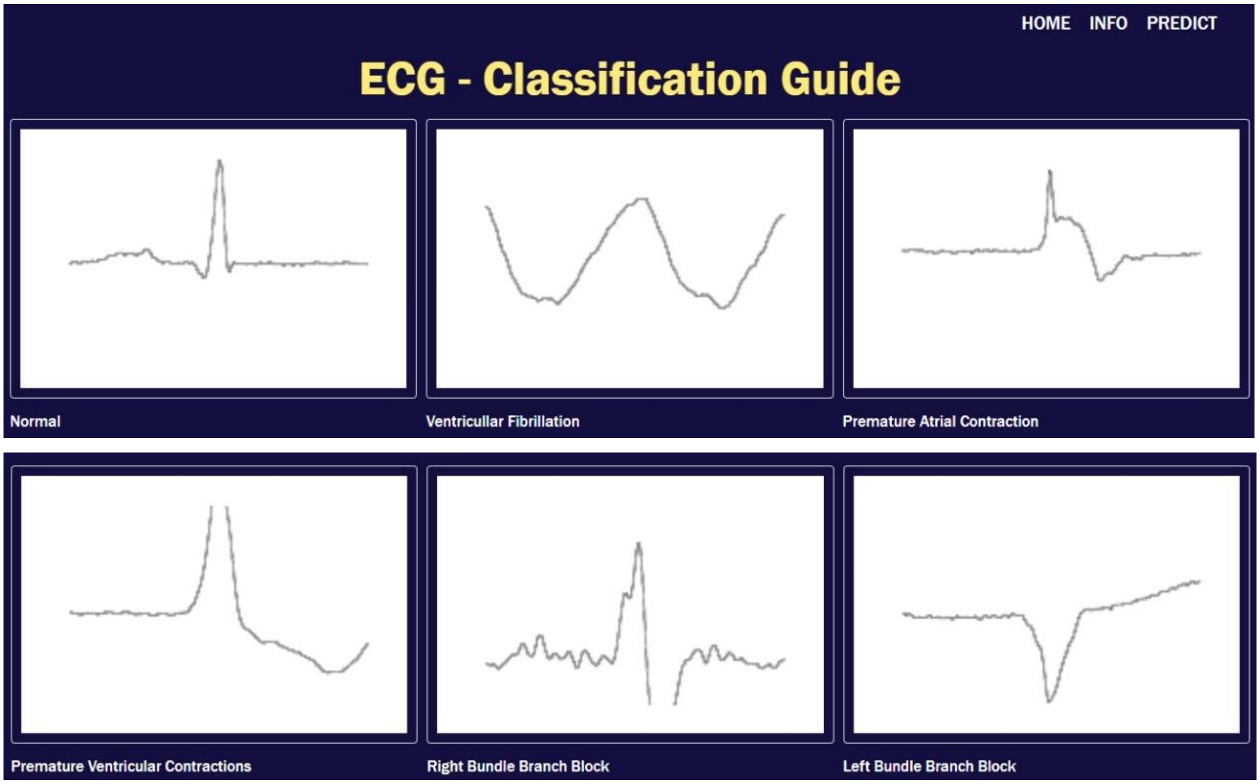
Home.html



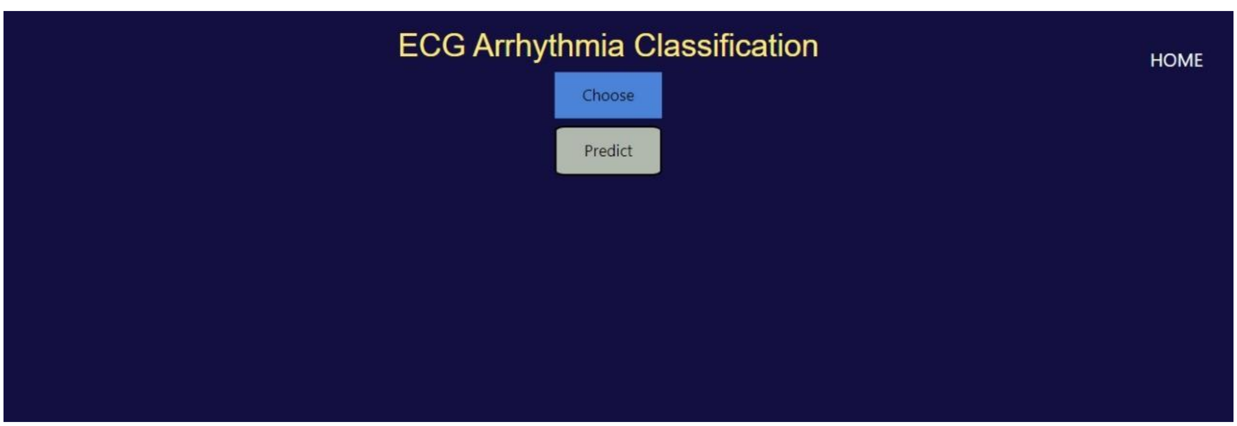
Info.html



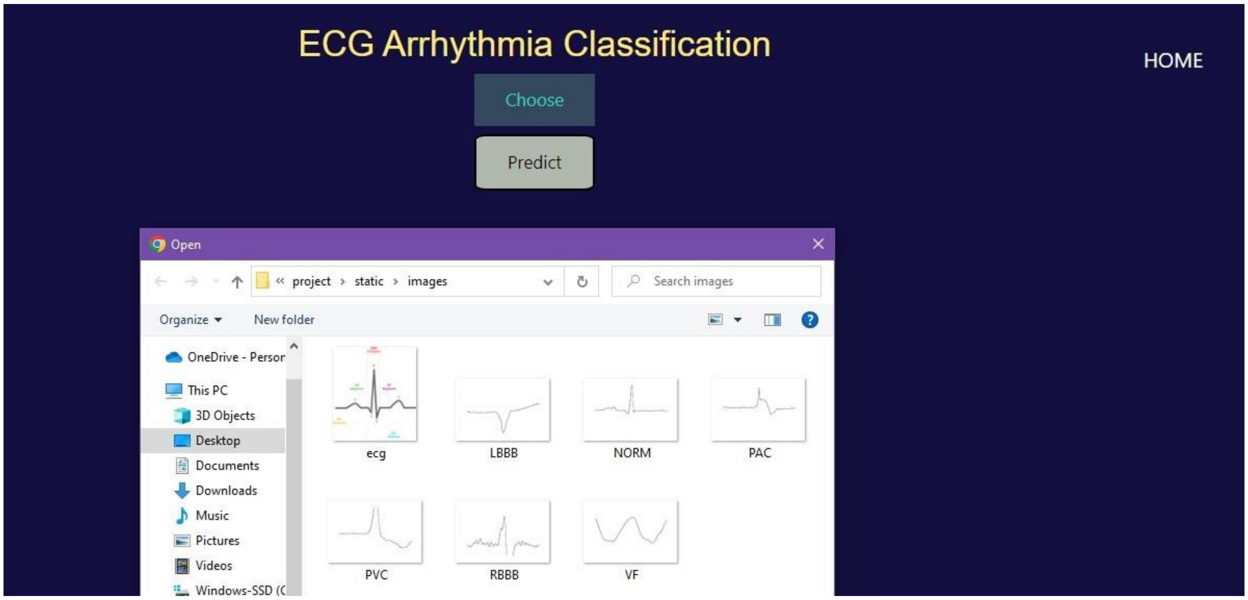
Guide.html



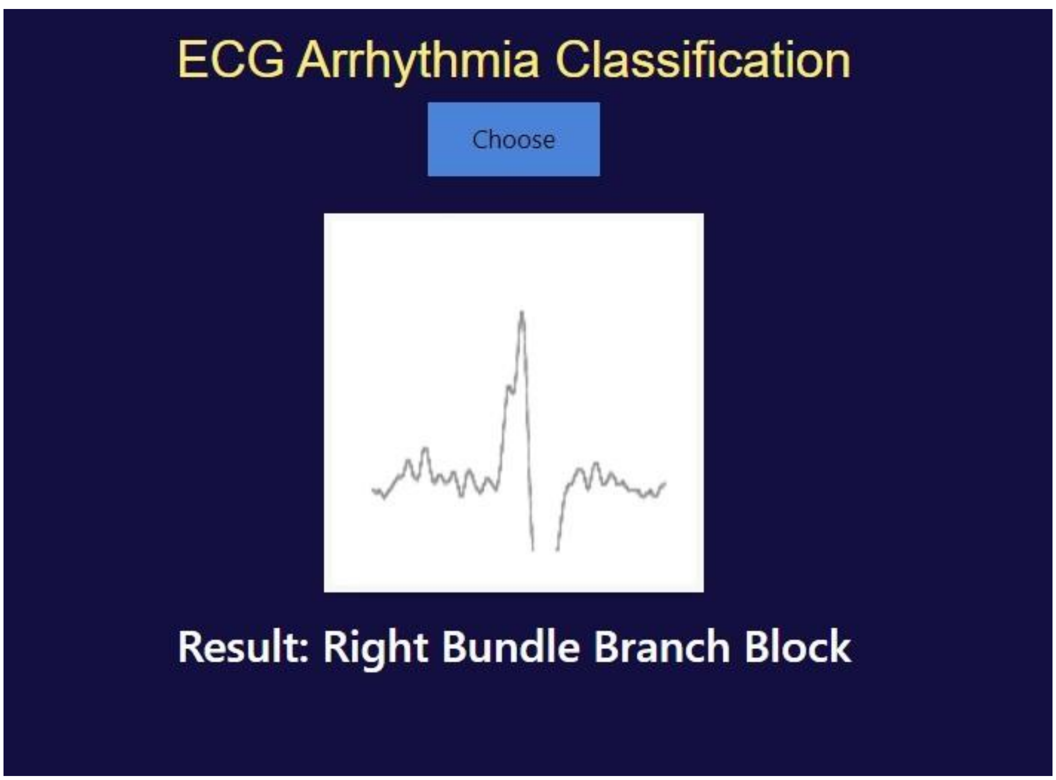
Predict.html



Upload file



Predicting output and displaying



7.11 Train Model on IBM Watson

Compress the file

In [17]:

```
model.save('ECG.h5')
```

In [18]:

```
#compress the file
!tar -zcvf arrhythmia-classification-model.tgz ECG.h5
```

```
ECG.h5
```

Deploying model in IBM Watson

In [19]:

```
!pip install ibm_watson_machine_learning
```

```
Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.26.7)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.
3.4)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (21.3)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.11.
0)
```

In [20]:

```
from ibm_watson_machine_learning import APIClient
```

In [21]:

```
wml_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey": "ChIkpkbcs--Hrju5LZSVzaEK-4HxxXYXSUez9G6VNuP6"
}
```

In [22]:

```
client = APIClient(wml_credentials)
```

In [23]:

```
client
```

Out[23]:

In [24]:

```
client.spaces.list()
```

```
Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50
------------------------------------  ------------------------  ------------------------
ID                                    NAME                      CREATED
11f2d9ff-4100-4f35-a236-3cc9706e0a79  arrhythmia_classification  2022-11-19T20:37:17.381Z
------------------------------------  ------------------------  ------------------------
```

```
In [25]: space_id = "11f2d9ff-4100-4f35-a236-3cc9706e0a79"

In [26]: client.set.default_space(space_id)

Out[26]: 'SUCCESS'

In [27]: client.software_specifications.list()

-----------------------------  ------------------------------------  ----
NAME                           ASSET_ID                              TYPE
default_py3.6                  0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base

In [28]: software_space_uid = client.software_specifications.get_id_by_name("tensorflow_rt22.1-py3.9")

In [29]: software_space_uid

Out[29]: 'acd9c798-6974-5d2f-a657-ce06e986df4d'
```

8.TESTING

Test case ID	Feature Type	Test Scenario	Steps to execute	Expected Result	Actual Result	Status
TC_001	Functional	Verify user is able to access the landing page	1.Enter URL and click go 2.Click upload button 3.Choose a image from local directory or paste or drop 4.Click predict to view result	Predicted Result Should Display	Working as expected	Pass

TC_ 002	UI	Verify the UI elements	1.Sliding banner 2.buttons	Application should show upload and predict button	Worki ng as expect ed	Pass
TC_ 003	Function al	Verify whether the link is legitimate or not	1.Enter URL and click go 2.Type or copy paste URL 3.Check the website is legitimate or not 4.Observe the results	User should observe whether the website is legitimate or not	Worki ng as expect ed	Pass
TC_ 004	Function al	Verify user is able to access the legitimate website or not	1.Enter URL and click go 2.Type or copy paste URL 3.Check the website is legitimate or not 4.Continue if the website is legitimate or be cautious if it is not legitimate	Application should show that safe webpage or unsafe	Worki ng as expect ed	Pass
TC_ 005	Function al	Testing website with multiple url	1.Enter URL and click go 2.Type or copy paste URL 3.Check the website is legitimate or not 4.Continue if the website is Secure or be cautious if it is not Secure	User can able to identify the websites whether it is secure or not	Worki ng as expect ed	Pass

9.RESULTS

9.1 Performance Metrics

SNO	MODEL	VALIDATION ACCURACY
1.	Xception	82.62%
2.	ResNet50	66.6%
3.	VGG19	87.5%
4.	CNN	87.9%

10.ADVANTAGES & DISADVANTAGES

10.1 Advantages

- The proposed model predicts Arrhythmia in images with a high accuracy rate of nearly 96%.
- The early detection of Arrhythmia gives better understanding of disease causes, initiates therapeutic interventions and enables developing appropriate treatments.

10.2 Disadvantages

- Not useful for identifying the different stages of Arrhythmia disease.
- Not useful in monitoring motor symptoms

11.CONCLUSION

Cardiovascular disease is a major health problem in today's world. The early diagnosis of cardiac arrhythmia highly relies on the ECG. Unfortunately, the expert level of medical resources is rare, visually identifying the ECG signal is challenging and time-consuming. The advantages of the proposed CNN network have been put to evidence. It is endowed with an ability to effectively process the non-filtered dataset with its potential anti-noise features. Besides that, ten-fold cross-validation is implemented in this work to further demonstrate the robustness of the network.

12.FUTURE SCOPE

For future work, it would be interesting to explore the use of optimization techniques to find a feasible design and solution. The limitation of our study is that we have yet to apply any optimization techniques to optimize the model parameters and we believe that with the implementation of the optimization, it will be able to further elevate the performance of the proposed solution to the next level.

13.APPENDIX

Source code

->CNN.ipynb file

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.utils import resample
from sklearn.model_selection import train_test_split

import tensorflow as tf

import keras
from keras.preprocessing.image import ImageDataGenerator
from keras.applications import VGG19
from keras.models import Model, Sequential
from keras.layers import Input, GlobalAveragePooling2D, BatchNormalization,
Dropout, Dense
from keras.layers import MaxPooling2D, Flatten, Conv2D, Softmax
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.models import load_model
```

```
from keras_preprocessing import image

!unzip 'IBM Dataset.zip'

train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True,)

test_datagen = ImageDataGenerator(rescale=1./255)

x_train = train_datagen.flow_from_directory(directory=r'/content/data/train',
                                            target_size=(128,128),
                                            batch_size=32,
                                            class_mode='categorical',
                                            classes=['Left Bundle Branch Block', 'Normal',
                                                     'Premature Atrial Contraction',
                                                     'Premature Ventricular Contractions',
                                                     'Right Bundle Branch Block', 'Ventricular Fibrillation'],
                                            )

x_test = test_datagen.flow_from_directory(directory=r'/content/data/test',
                                          target_size=(128,128),
                                          batch_size=32,
                                          class_mode='categorical',
                                          classes=['Left Bundle Branch Block', 'Normal',
                                                   'Premature Atrial Contraction',
                                                   'Premature Ventricular Contractions',
                                                   'Right Bundle Branch Block', 'Ventricular Fibrillation'],

tf.config.optimizer.set_jit(True)

model = Sequential()
```

```
model.add(Conv2D(32, (3,3), padding='same', activation='relu', input_shape=(128, 128,
3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(32, (3,3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(35))
model.add(Dense(6, activation='softmax'))

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

model.summary()
model_history = model.fit(x_train,
                  steps_per_epoch=len(x_train),
                  epochs=25,
                  validation_data=x_test,
                  validation_steps=len(x_test),)

model.save('ECG (CNN).h5')

model = load_model("ECG (CNN).h5")
img = image.load_img("/content/data/test/Premature Atrial Contraction/fig_107.png",
target_size=(128, 128))

index = ['Left Bundle Branch Block', 'Normal','Premature Atrial Contraction',
     'Premature Ventricular Contractions','Right Bundle Branch Block', 'Ventricular
Fibrillation']

x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
pred = model.predict(x)
result = str(index[np.where(pred[0]==1)[0][0]])
```

```
print("Predicted Class: ", result)
```

-> flask python code

```
import os
import numpy as np
from flask import Flask,request,render_template
from keras.models import load_model
import tensorflow as tf
from PIL import Image

app=Flask(__name__)
model=load_model('ECG.h5')

@app.route("/")
def about():
    return render_template("home.html")

@app.route("/home")
def home():
    return render_template("home.html")

@app.route("/info")
def info():
    return render_template("info.html")

@app.route("/guide")
def guide():
    return render_template("guide.html")

@app.route("/predict")
def test():
```

```
    return render_template("predict.html")

@app.route("/predict",methods=["GET","POST"])
def upload():
    if request.method=='POST':
        f=request.files['file']
        basepath=os.path.dirname('__file__')
        filepath=os.path.join(basepath,"uploads",f.filename)
        f.save(filepath)

        img=tf.keras.utils.load_img(filepath,target_size=(128,128))
        x=tf.keras.utils.img_to_array(img)
        x=np.expand_dims(x,axis=0)

        pred=model.predict(x)
        y_pred = np.argmax(pred)
        print("prediction",y_pred)

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

        return result
    return None

if __name__=="__main__":
    app.run(debug=False)
```

->User Interface:

Home.html

```
<!DOCTYPE html>
<html>

<head>

<title>Home</title>

<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="../static/css/index.css">
<link href="{{ url_for('static', filename='css/index.css') }}" rel="stylesheet">

</head>
<body>

   <div class="navbar">
      <a href="/predict" >PREDICT</a>
      <a href="/info">INFO</a>
   </div>

   <div>
      <center><h2 class="header">ARRHYTHMIA PREDICTION</h2></center>
      <br>
      <center>
         <b class="pd">
            <font color = "#ffec78" size="13" font-family = "Helvetica">
               ECG arrhythmia classification using CNN
            </font>
         </b>
```

```
        </center>
    </div> <br>

    <center>
        <p>
            <font color = "white">
                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.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.
            </font>
        </p>
    </center>

</body>
</html>
```

Info.html

```
<!DOCTYPE html>
<html>
<head>
<style>
img{
```

```
width:20%;
height:10%;
padding:20px;
margin-top:5px;
}
</style>
<title>
    Info
</title>

<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="../static/css/index.css">
<link href="{{ url_for('static', filename='css/index.css') }}" rel="stylesheet">

</head>

<body>
    <div class="navbar">
        <a href="/predict" >PREDICT</a>
        <a href="/guide">ECG</a>
        <a href="/home">HOME</a><br>
    </div>

    <div class="container" >
         <p>
             <h2>
                 Types of Arrhythmia
             </h2>

        <b>
            <font color = "#bb8206">
                Supraventricular arrhythmias :
```


Arrhythmias that begin in the atria (the heart's upper chambers). "Supra" means above.
“Ventricular” refers to the lower chambers of the heart or ventricles.

Ventricular arrhythmias :

Arrhythmias that begin in the ventricles (the heart's lower chambers).

Bradyarrhythmias :

Slow heart rhythms that may be caused by disease in the heart's conduction system,
such as the sinoatrial (SA) node, atrioventricular (AV) node or HIS-Purkinje network.
</p>

<h2>
Symptoms of Arrhythmia
</h2>

<ol style="display:inline-table;">

A feeling of skipped heartbeat or that your heart is “running away,” fluttering or doing "flip-flops.

Pounding in your chest.

Dizziness or feeling lightheaded.

Shortness of breath.

Chest discomfort.

Weakness or fatigue (feeling very tired).

Weakening of the heart muscle or low ejection fraction.

<h2>
Causes of Arrhythmia
</h2>

<ol style="display:inline-table;">

Coronary artery disease.

Irritable tissue in the heart (due to genetic or acquired causes).


```
            High blood pressure.
          </li>
          <li>
            Changes in the heart muscle (cardiomyopathy).
          </li>
          <li>
            Valve disorders.
          </li>
          <li>
            Electrolyte imbalances in your blood, such as sodium or potassium imbalances.
          </li>
          <li>
            Injury from a heart attack.
          </li>
          <li>
            The healing process after heart surgery.
          </li>
          <li>
            Other medical conditions.
          </li>
        </ol>

    </div>
</body>

</html>
```

Predict_base.html

```
<!DOCTYPE html>
<html>
```

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Predict</title>

    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="../static/css/index.css">

    <link href="{{ url_for('static', filename='css/index.css') }}" rel="stylesheet">



</head>

<body>
    <div class="navbar">
        <a href="/guide">ECG</a>
        <a href="/info">INFO</a>
        <a href="/home">HOME</a><br>
    </div>

    <div class="container">
      <center>
          <div id="content" style="margin-top:2em">
              {% block content %}{% endblock %}
          </div>
       </center>
    </div>

</body>

<footer>
```

```
        <script src="{{ url_for('static', filename='js/main.js') }}"
type="text/javascript"></script>
    </footer>
</html>
```

<u>Index.css</u>

```
.header {
    padding: 30px;
    text-align: center;
    font-family: Arial, Helvetica, sans-serif;
    color: #93a3b6;
    font-size: 30px;
  }


body{
    color: white;
    background: #130f40;
}

h2{
    font-family: Arial, Helvetica, sans-serif;
    color: #ffe169;
}

p{
    color:white;
    font-family:Arial, Helvetica, sans-serif;
    font-size: 18px;
}
```

```css
.navbar{
    margin : 0px;
    padding : 20px;
    padding-bottom: 20px;
    opacity : 5;
}

a:hover{
    background-color: #ffe169;
    border-radius:10px;
    padding-left:20px;
    color:black;
}

a{
    color:white;
    font-size: 20px;
    text-align: center;
    text-decoration: none;
    float:right;
    padding-right:20px;
}

.img-preview {
    width: 256px;
    height: 256px;
    position: relative;
    border: 5px solid #F8F8F8;
    box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
    margin-top: 1em;
    margin-bottom: 1em;
}
```

```
.img-preview>div {
   width: 100%;
   height: 100%;
   background-size: 256px 256px;
   background-repeat: no-repeat;
   background-position: center;
}

input[type="file"] {
   display: none;
}

.upload-label{
   display: inline-block;
   padding: 12px 30px;
   background: #39D2B4;
   color: #fff;
   font-size: 1em;
   transition: all .4s;
   cursor: pointer;
}

.upload-label:hover{
   background: #34495E;
   color: #39D2B4;
}

.loader {
   border: 8px solid #f3f3f3;
   border-top: 8px solid #ffec78;
   border-radius: 70%;
```

```
    width: 25px;
    height: 25px;
    animation: spin 1s linear infinite;
}

@keyframes spin {
    0% { transform: rotate(0deg); }
    100% { transform: rotate(360deg); }
}
```

GitHub & Project Demo Link

Github Link: https://github.com/IBM-EPBL/IBM-Project-4249-1658725786

Project Demo Link:

https://drive.google.com/file/d/12aqxRVgUeILN8uxQLqpttncQ-VMGL_zi/view?usp=share_link