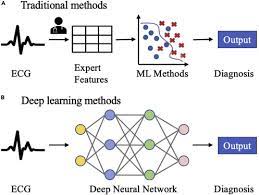
**Arrhythmia Detection Using Deep Learning With 2-D spectrum image Representation**

1. **INTRODUCTION** 
   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 convolution al 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

* 1. **Purpose**

In the past few decades, Deep Learning has proved to be a compelling tool  
because of its ability to handle large amounts of data. The interest to use  
hidden layers has surpassed traditional techniques, especially in pattern  
recognition. One of the most popular deep neural networks is Convolution  
al Neural Networks.



1. **LITERATURE SURVEY**
   1. **Existing problem**

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.

**Diagram

Description automatically generated**

* 1. **References**

|  |  |  |  |
| --- | --- | --- | --- |
| **Topic** | **Authors** | **Published**  **Year** | **Abstract** |
| Classiﬁcation of Arrhythmia by Using Deep Learning  with 2-D ECG Spectral Image Representation | Amin Ullah,Syed Muhammad Anwar,Muhammad Billal,Raja Mehmood | 2020 | In this study they have proposed a two-dimensional(2D) convolutional neural network (CNN) model for the classification of ECG signals into the eight classes. The eight classes are the 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 proposed CNN classifier was implemented in Python with the open-source library Tensor Flow which was developed by Google for deep learning. This research paper uses only a single-lead ECG signal. |
| Cardiac Arrhythmia Detection from 2D ECG Images by Using Deep Learning Technique | E. Izci, M. A. Ozdemir, M. Degirmenci and A. Akan | 2019 | In this research paper, they proposed a deep learning approach using convolutional neural network for classifying the five different types of arrythmia disease. They segmented the heartbeats out of the ECG radio signals and all the images are converted into grayscale images. They implemented novel preprocessing, extraction and selection. They achieved an accuracy of 97.52% |
| Cyberbullying Detection Using Machine Learning | Aaminah Ali, Adeel M.Syed | 2020 | In this study they used numeric data for training, So the text was first converted into the numerical form using a label encoder. After that, the dataset divided into 80% training set, and 20% test and then classification algorithms were applied. The algorithms were used in classification are SVM, naïve Bayes, Random Forest and then an ensemble approach. The ensemble approach was a hybrid model, In this approach, a soft voting criterion was used. Which predicts the class label utilizing the maximum sum of the predicted probabilities. |
| Social Media Cyberbullying Detection using Machine Learning | John Hani , Mohamed Nashaat , Mostafa Ahmed | 2019 | In this research paper, they have proposed an approach to detect cyberbullying using machine learning techniques. And they evaluated their model on two classifiers SVM and neural network and they used TFIDF and sentiment analysis algorithms for features extraction. The dataset has been taken from Kaggle. Thus, a larger cyberbullying data is needed to improve the performance. Hence, deep learning techniques will be suitable in the larger data as they are proven to outperform machine learning approaches over larger size data. |

* 1. **Problem Statement Definition**

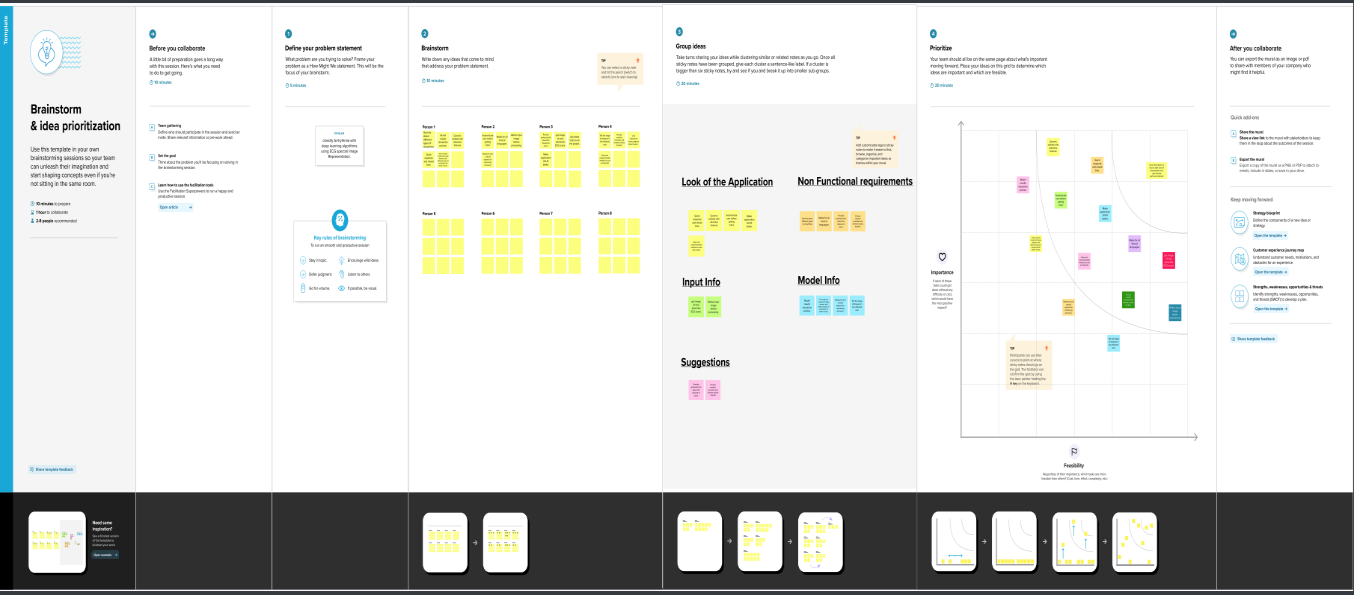
The presented problem wants us to identify and classify the types of Arrhythmias provided in the dataset using spectral images of the Electrocardiogram that are employed for prediction of cardiovascular diseases

1. **IDEATION & PROPOSED SOLUTION**
   1. **Empathy Map Canvas**

**Graphical user interface, diagram

Description automatically generated with medium confidence**

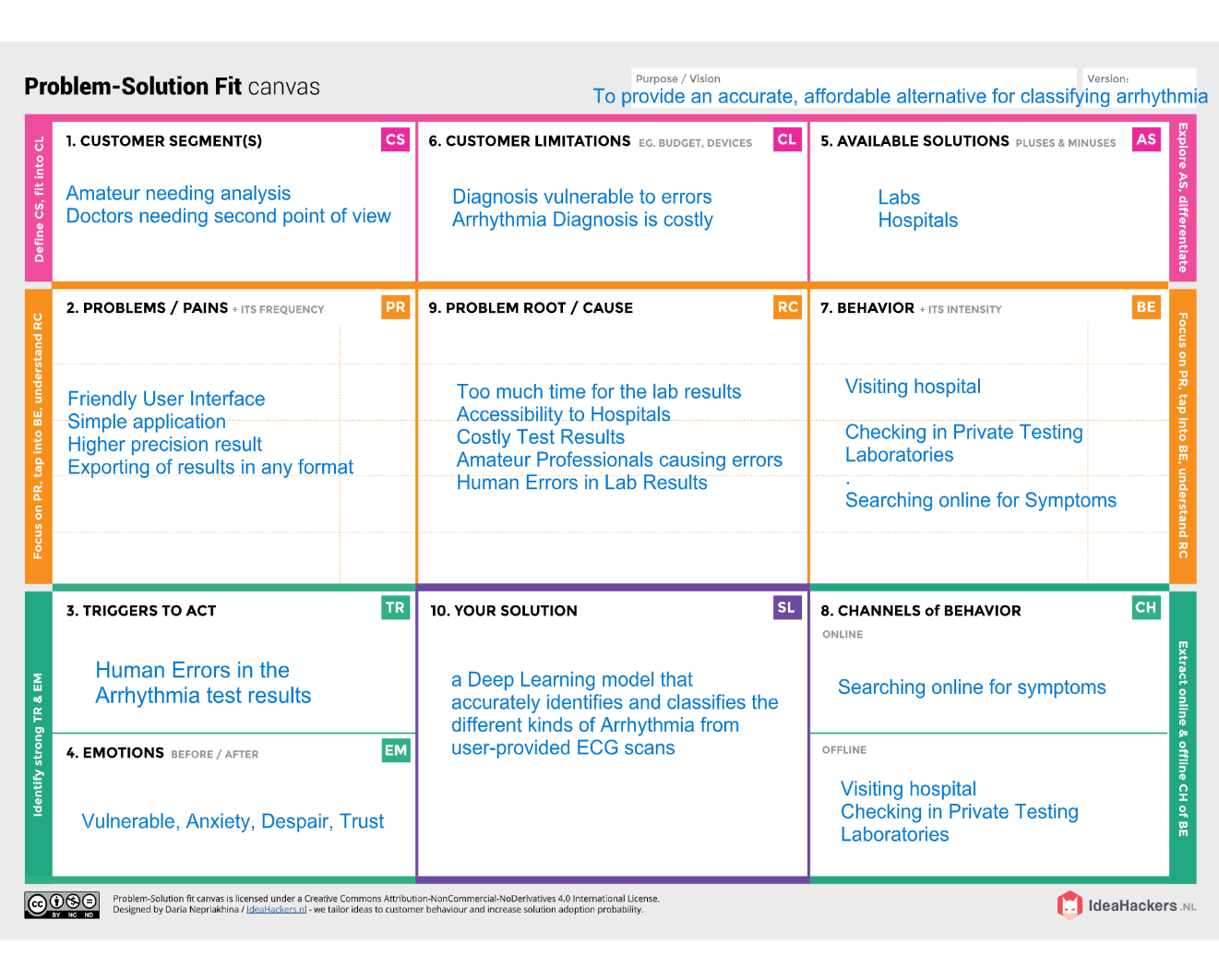
* 1. **Ideation & Brainstorming**

****

* 1. **Proposed Solution**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Parameter** | **Description** | |
| 1. | Problem Statement (Problem to be solved) | The presented problem wants us to identify and classify the types of Arrhythmias provided in the dataset using spectral images of the Electrocardiogram that are employed for prediction of cardiovascular diseases. | |
| 2. | Idea / Solution description | The given dataset for this problem initially undergoes various data pre-processing steps to identify various forms of noise in the dataset and denoise them to make the data suitable for training a deep learning model. We will employ 2-Dimensional  Convolutional Neural Network Model to carry out this classification. | |
| 3. | Novelty / Uniqueness | * Deploys the model to a mobile application by assigning all heavy pre-processing to the cloud through an API. * Usage of cloud-based ML-training services. * Proposed solution will be deployed in the cloud making it easily accessible over the internet to people across the globe. * Provide well detailed instructions or point the user to well-equipped hospitals to get good treatment. | |
| 4. | Social Impact / Customer Satisfaction | * Promotes Simplicity. * Promotes Self-Diagnosis. * Requires minimal effort and time. * Proposed solution abides by privacy laws and no private information of user is stored. * Delivers highly accurate results(classification of arrhythmia) in a short span of time. | |
| 5. | Business Model (Revenue Model) | • | Our business model primarily covers the expense we incur by deploying the service in cloud platforms. |
|  |  | • | Primary consumers of our proposed service are hospitals who seek immediate consultation or use our service as a reference. |
|  |  | • | Our service can be used by anybody who has access to internet services. |
|  |  | • | Most of the competing products do not offer a Revenue is generated from Corporate Editions which has a monthly subscription, whereas the Community Edition is free for individuals. |
|  |  | • | Users who would like to not travel to hospitals to get an ECG or get a selfdiagnosis can rent/buy an ECG Machine through our service which will be the secondary source of income. |
| 6. | Scalability of the Solution | • | Increasing the dataset used for model training will in turn increase the application's scalability. |
|  |  | • | Making the model more reliable will lead to hospitals using this application, this also eliminates human error. |
|  |  | • | More powerful Cloud Instances for Concurrent use of the application. |
|  |  | • | Periodically expanding the dataset and updating the model to increase scalability and reliability. |
|  |  | • | Cloud services guarantee high availability so there is very little probability that the service will face serious down times. |

* 1. **Problem Solution fit**

****

1. **REQUIREMENT ANALYSIS**
   1. **Functional requirement**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story/Sub-Task)** |
| FR-I | User Registration | registration on a website utilising an email address for confirmation |
| FR-2 | User Confirmation | OTP verification on the webpage |
| FR-3 | User Data | uploading the ECG image used to diagnose arrhythmias |
| FR-4 | User Diagnosis Results (Single Input) | The website displays the results of the arrhythmia prediction and classification. |
| FR-5 | User Diagnosis Results (Multiple Input) | The Arrhythmia Prediction & Classification results are emailed to the registered email. |

* 1. **Non-Functional requirements**

Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **NFR No.** | **Non-Functional Requirement** | **Sub Requirement (Story/Sub-Task)** |
| NFR-I | Reliability | Probability of successful operations in each environment for a certain period of time |
| NFR-2 | Scalability | The website's ability to accommodate expansion |
| NFR-3 | Security | Authentication and Authorization of the website |
| NFR-4 | Usability | efficient and overall user satisfaction with the diagnosis service provided by the website |
| NFR-5 | Availability | All users should be able to access the service easily. |

1. **PROJECT DESIGN**
   1. **DataFlowDiagrams**

**Diagram

Description automatically generated**

**Diagram

Description automatically generated**

* 1. **Solution & Technical Architecture**

**Diagram

Description automatically generated**

* 1. **User Stories**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| User Type | Functional  Requirement  (Epic) | User  Story  Number | User  Story/Task | Acceptance  Criteria | Priority | Release |
| Customer (Web User) | Registration | USN-1 | As a user I can register an account on the website | I can access classification page | High | Sprint-1 |
|  | Login | USN-2 | As a user I can login into the website using credentials | I can access classification page | High | Sprint-1 |
|  | Arrhythmia Classification page | USN-3 | User uploads the data | Input for the model | High | Sprint-1 |
|  | Info page | USN-4 | Info page displays information about Arrhythmia | User can learn about Arrhythmia | Medium | Sprint-2 |

1. **PROJECT PLANNING & SCHEDULING**
   1. **Sprint Planning & Estimation**

|  |  |  |
| --- | --- | --- |
| **Title** | **Description** | **Date** |
| **Literature Survey and**  **Information Gathering** | Technical papers were referred to gather information | 10 September 2022 |
| **Prepare Empathy Map** | Problem statement was addressed by considering user pain and gain | 17 September 2022 |
| **Ideation** | Determine the importance and viability of the top three concepts. | 18 September 2022 |
| **Proposed Solution** | organizational innovation, scalability, and business  the solution's model, economic effect, and scalability | 1 October 2022 |
| **Problem Solution Fit** | Document-fitting solution | 1 October 2022 |
| **Solution Architecture** | Structure of the Solution | 1 October 2022 |
| **Customer Journey** | to comprehend user interactions and application experiences | 8 October 2022 |
| **Functional Requirement** | To comprehend user needs and create functional requirements | 9 October 2022 |
| **Data flow Diagrams** | Data flow diagram | 11 October 2022 |
| **Technology Architecture** | Technology Architecture diagram | 15 October 2022 |
| **Milestone & sprint delivery plan** | Activity what we done  &further plans | 21 October 2022 |
| **Project Development-**  **Delivery of sprint 1,2,3 &4** | Submission of code after testing using software | 24 October 2022 –  19 November 2022 |

1. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
   1. **Application Building**

We created a website using HTML, CSS which helps the user to classify the type of arrhythmia by inputting their ECG image into the website. Flask was also used to render our HTML files and also to integrate our model to the website which helps in classification of the disease.

* 1. **Model Building**

Model Building is one of the most important parts of the application which helps to classify the arrhythmia disease. We imported various packages and libraries such as TensorFlow, Pandas, NumPy, etc. for data manipulation and pre-processing. Dataset was taken from Kaggle and we trained and tested using ImageDataGenerator.

We used the Sequential CNN (Convolution Neural Network) model as our classifying model which had 2 hidden layers. Model was made to run a total of 10 epochs for higher accuracy and we attained an accuracy of 92% which is a very high accuracy for disease prediction

1. **RESULTS**
   1. Performance Metrics

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Parameter** | **Values** | **Screenshot** |
|  | Metrics | **Regression Model:** MAE - , MSE - , RMSE - , R2 score -  **Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report - |  |

1. **ADVANTAGES & DISADVANTAGES**

**ADVANTAGES:**

* Very fast and time-efficient
* Easy to access and communicate
* Cost-efficient

**DISADVANTAGES:**

* Low chance of error in classification
* No emotion in conveying the result
* Requires good internet access for accessing the website which may not be available in an urgent situation

1. **CONCLUSION**

Application was built using various web technologies such as HTML, CSS, Flask, etc and model was also build using Machine Learning Techniques such as Data Preprocessing and model building using TensorFlow and other packages to classify arrhythmia accurately and time-efficiently.

1. **FUTURE SCOPE**

This application can help other classification of diseases much more easier as it is nearly equivalent to this application and can further be researched to improve accuracy and other metrics.

1. **APPENDIX**

Source Code

Model code

from keras.preprocessing.image import ImageDataGenerator

import numpy as np

import tensorflow as tf

import PIL

from PIL import Image

from tensorflow.keras import datasets, layers, models

from tensorflow.keras.models import Sequential

from tensorflow.keras import layers

from tensorflow.keras.layers import Dense,Flatten

from tensorflow.keras.layers import Conv2D,MaxPooling2D

train\_datagen= ImageDataGenerator(rescale=1./255, shear\_range=0.2, zoom\_range=0.2, horizontal\_flip= True)

test\_datagen= ImageDataGenerator(rescale=1./255)

import os, types

import pandas as pd

from botocore.client import Config

import ibm\_boto3

def \_\_iter\_\_(self): return 0

# @hidden\_cell

# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.

# You might want to remove those credentials before you share the notebook.

cos\_client = ibm\_boto3.client(service\_name='s3',

    ibm\_api\_key\_id='mN-ebj9Okze1dUnk6NRehD1nG6pbj\_9LEkFJyxCLIMg8',

    ibm\_auth\_endpoint="https://iam.cloud.ibm.com/oidc/token",

    config=Config(signature\_version='oauth'),

    endpoint\_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'arrhythmiaclassification-donotdelete-pr-4avxy8o7d2xjtk'

object\_key = 'Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation.zip'

streaming\_body\_3 = cos\_client.get\_object(Bucket=bucket, Key=object\_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.

# Please read the documentation of ibm\_boto3 and pandas to learn more about the possibilities to load the data.

# ibm\_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/

# pandas documentation: http://pandas.pydata.org/

from io import BytesIO

import zipfile

unzip = zipfile.ZipFile(BytesIO(streaming\_body\_3.read()),'r')

file\_paths= unzip.namelist()

for path in file\_paths:

    unzip.extract(path)

import os

filenames=os.listdir('/home/wsuser/work/Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation/Dataset')

filenames

x\_train=train\_datagen.flow\_from\_directory(directory=r'/home/wsuser/work/Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation/Dataset/train', target\_size=(64,64), batch\_size=32, class\_mode='categorical')

x\_test=train\_datagen.flow\_from\_directory(directory=r'/home/wsuser/work/Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation/Dataset/test', target\_size=(64,64), batch\_size=32, class\_mode='categorical')

model=models.Sequential()

model.add(layers.Conv2D(32,(3,3), input\_shape=(64, 64, 3), activation='relu'))

model.add(layers.MaxPooling2D((2,2)))

model.add(layers.Conv2D(32,(3,3), activation='relu'))

model.add(layers.MaxPooling2D((2,2)))

model.add(layers.Flatten())

model.add(Dense(32))

model.add(Dense(6, activation='softmax'))

model.summary()

model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

model.fit\_generator(generator=x\_train, steps\_per\_epoch=len(x\_train), epochs=10, validation\_data=x\_test, validation\_steps=len(x\_test))

model.save('ECG.h5')

!tar -zcvf classification-model.tgz ECG.h5

from ibm\_watson\_machine\_learning import APIClient

wml\_credentials={

    "url":"https://us-south.ml.cloud.ibm.com",

    "apikey":"ur2JUzcKDS7x3V4Y0m4sVewkTi-qifnbCQpI1t-Yf4-D"}

client=APIClient(wml\_credentials)

def guid\_from\_space\_name(client, space\_name):

    space=client.spaces.get\_details()

    #print(space)

    return(next(item for item in space ['resources'] if item['entity']["name"]==space\_name)['metadata']['id'])

space\_uid= guid\_from\_space\_name(client,'model\_deployment')

print("Space UID -" + space\_uid)

client.set.default\_space(space\_uid)

client.software\_specifications.list(limit=100)

software\_spec\_uid = client.software\_specifications.get\_uid\_by\_name("tensorflow\_rt22.1-py3.9")

software\_spec\_uid

model\_details= client.repository.store\_model(model='classification-model.tgz',meta\_props={

    client.repository.ModelMetaNames.NAME:"CNN" ,

    client.repository.ModelMetaNames.TYPE:"tensorflow\_2.7" ,

    client.repository.ModelMetaNames.SOFTWARE\_SPEC\_UID:software\_spec\_uid})

model\_id = client.repository.get\_model\_uid(model\_details)

client.repository.download(model\_id,'my\_model.tar.gz' )

from tensorflow.keras.models import load\_model

from keras.preprocessing import image

import tensorflow as tf

from tensorflow import keras

model = load\_model("ECG.h5")

x= image.load\_img('/home/wsuser/work/Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation/Dataset/train/Ventricular Fibrillation/VFEfig\_182.png',target\_size=(64,64))

x= np.expand\_dims(x, axis=0)

pred= model.predict(x)

pred

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

result=str(index[pred[0].tolist().index(1.)])

result

App.py

import os

import numpy as np

from flask import Flask, request, render\_template, url\_for

from tensorflow.keras.models import load\_model

from tensorflow.keras.preprocessing import image

app=Flask(\_\_name\_\_)#our flask app

model=load\_model('ECG.h5')

@app.route("/") #default routC:\Users\hariharan\Documents\Projects\Untitled Foldere

def about():

    return render\_template( "about.html")

@app.route("/about")

def home():

    return render\_template("about.html")

@app.route("/info") #default route

def information():

    return render\_template("info.html")

@app.route("/upload")

def test():

    return render\_template("index6.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=image.load\_img(filepath,target\_size=(64,64))

        x=image.img\_to\_array(img)

        x=np.expand\_dims(x,axis=0)

        pred=(model.predict(x) > 0.5).astype("int32")

        print("Prediction",pred)

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

        result=str(index[pred[0].tolist().index(1.)])

        return render\_template("base.html", name = result)

    return None

if \_\_name\_\_=="\_\_main\_\_":

    app.run(debug=True)

    app.run(host='127.0.0.1', port=5000)

GitHub & Project Demo Link

Github: https://github.com/IBM-EPBL/ECG-Project