____________PROJECT REPORT____________

Classification of Arrhythmia Using Deep Learning with 2-D ECG Spectral Image Representation

Introduction:

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 intoseven 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.

Purpose

In the past few decades, Deep Learning has proved to be a compelling toolbecause 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.

In deep learning, a convolution al neural network (CNN/ConvNet) is a classof deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics convolution is a mathematical operation on two functions that produces a third function thatexpresses how the shape of one is modified by the other.

Literature Survey

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 worldwhich 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.

References

"Analysis of ECG Signal and Classification of Arrhythmia," by H. S. Bhanu, S. Tejaswini, M. S. Sahana, K. Bhargavi, K. S. Praveena and S. S. Jayanna.

"An algorithm for detection of arrhythmia" by Mujeeb Rahman, Mohamed Nasor.

"Matlab Based GUI for ECG Arrhythmia Detection Using Pan-Tompkin Algorithm,“ by B. Jindal, Saudagar, Ekta and R. Devi.

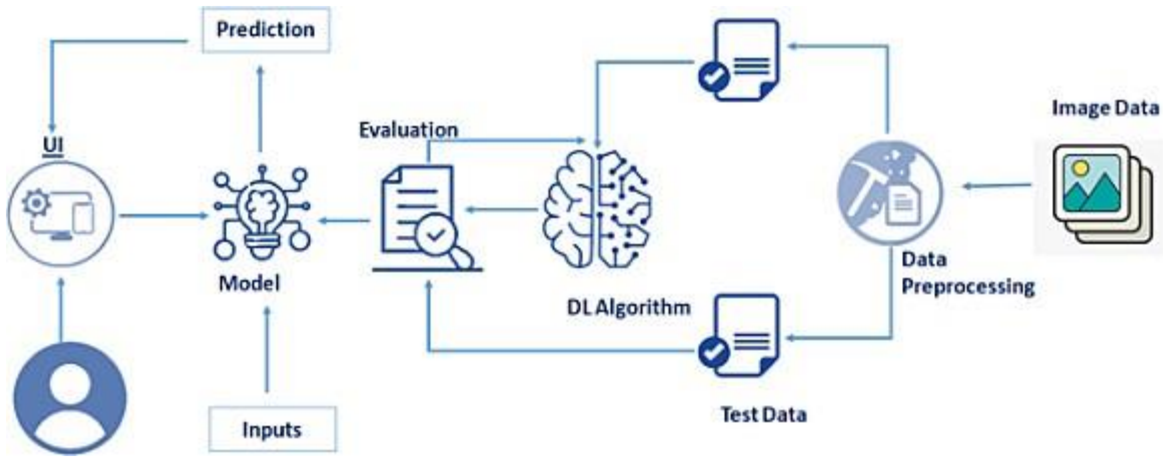
"An efficient neural network-based method for patient-specific information involved arrhythmia detection,” by Yunqing Liu, Chengjin Qin, Jinlei Liu, Yanrui Jin, Zhiyuan Li, Chengliang Liu.

"ECG-based heartbeat classification for arrhythmia detection: A survey,“ by Eduardo José da S. Luz, William Robson Schwartz, Guillermo Cámara-Chávez, David Menotti.

Problem Statement Definition

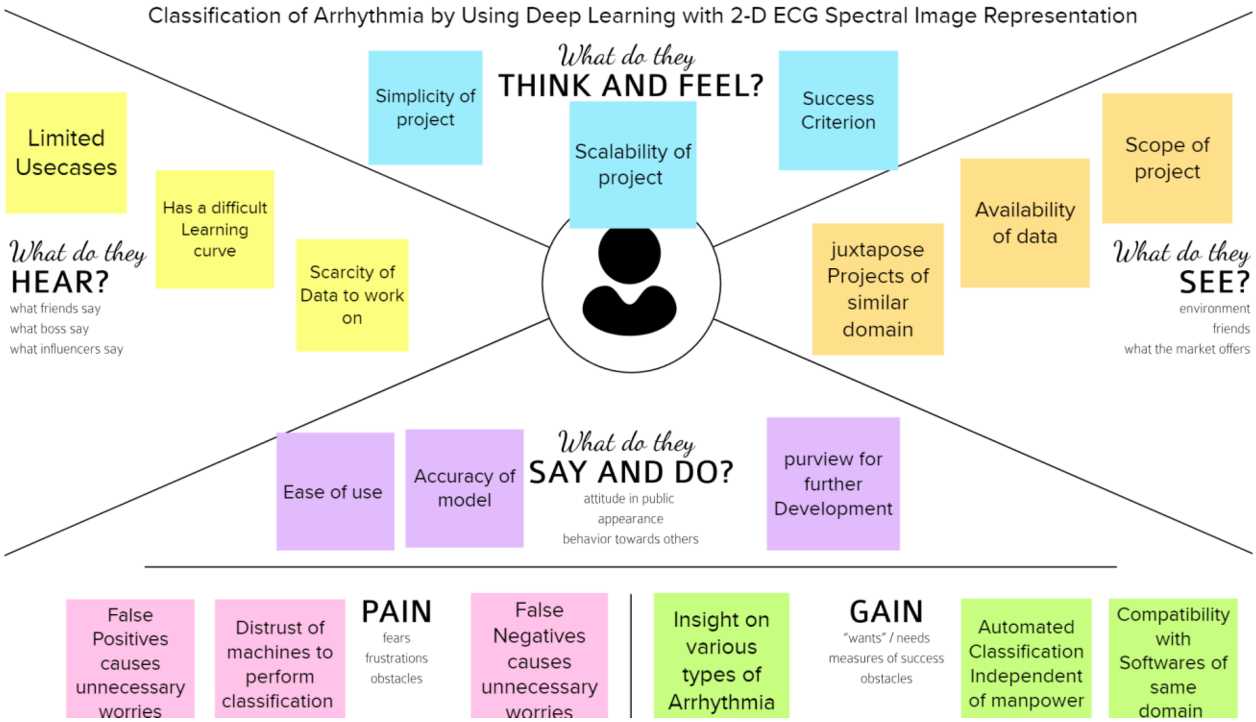
The Problem is to design and create an application that can automate the process of classifying arrhythmia which is usually done by doctors or other medical practitioners manually.

Shown below, is a flowchart that depicts what the application does:

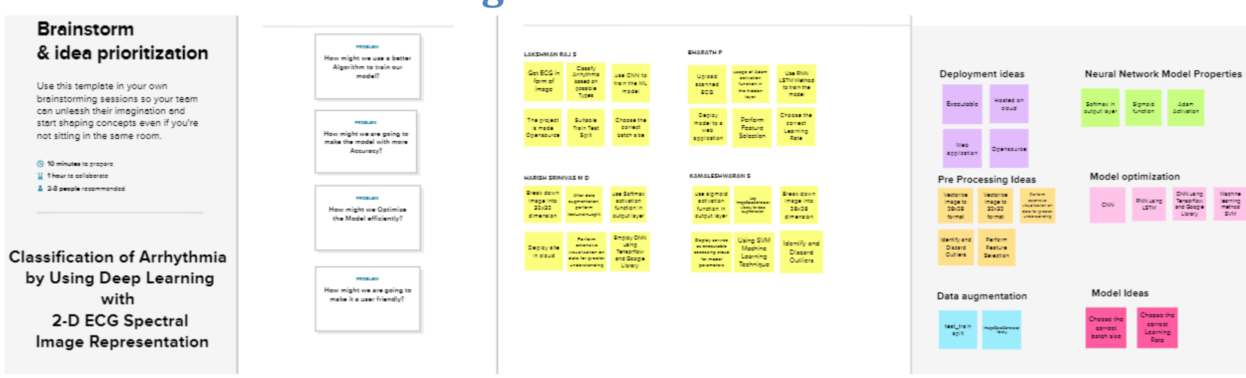


Ideation And Proposed Solution

Empathy Map Canvas



Ideation & Brainstorming



Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Arrhythmia is a representative type of CVD that refers to any irregular change from the normal heart rhythms. There are several types of arrhythmia, although a single arrhythmia heartbeat may not have a serious impact on life, continuous arrhythmia beats can result in fatal circumstances. This project should enable a medical practitioner to rapidly identify the exact type of arrhythmia of the patient (customer).
2.	Idea / Solution description	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.
3.	Novelty / Uniqueness	One of the major decisions had to be made was choosing the suitable programming language satisfying our goal for extracting knowledge from our data. After some searching the suitable decision has been made by selecting Python as the project programming language. Due to the fact that, a lot of tools and frameworks are available for Python to create powerful Artificial Neural Networks. Also IBM Watson helps to predict future outcomes,automate complex processes,and optimize user's time. And also the result accuracy will be increased from 70% which is the accuracy of the test results that the previous developed codes produced.
4.	Social Impact / Customer Satisfaction	This project can help decrease the time to diagnose the correct type of arrhythmia and enable doctors to spend more time and effort into treating the condition, which can dramatically increase satisfaction of the customer and free the doctor of more work.

Problem Solution fit

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

For this application, there are only three kinds of customers – doctors, medical assistants and patients.

6. CUSTOMER CONSTRAINTS CC

Classifying the correct arrhythmia type from ECG scans require much time and effort which can be better spent in treatment.

5. AVAILABLE SOLUTIONS AS

The process of arrhythmia classification is simple but time consuming. The solution is to automate it.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

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

By automating the classification process, a great deal of time and effort is saved for the medical practitioners.

9. PROBLEM ROOT CAUSE RC

The root cause of the problem is the inability of humans to do simple but repetitive tasks leading to mistakes in the process (which can lead to wrong treatment).

7. BEHAVIOUR BE

The customer (medical practitioners in this case) seek to automate the process. This enables them to treat patients faster (saving more lives) and enabling them to do more.

Focus on J&P, tap into BE, understand RC

Requirement Analysis

Functional Requirement

Functional Requirements:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Gmail
FR-2	User Confirmation	Confirmation via Email
FR-3	User profile	Display personal details and medical history of users. Access ECG images saved, along with the classification results.
FR-4	Process image	The trained CNN model processes the input image to classify the Arrhythmia.
FR-5	User Input	Upload image in jpeg/png format.
FR-6	Generate output	Display the classification label on the screen.

Non Functional Requirement

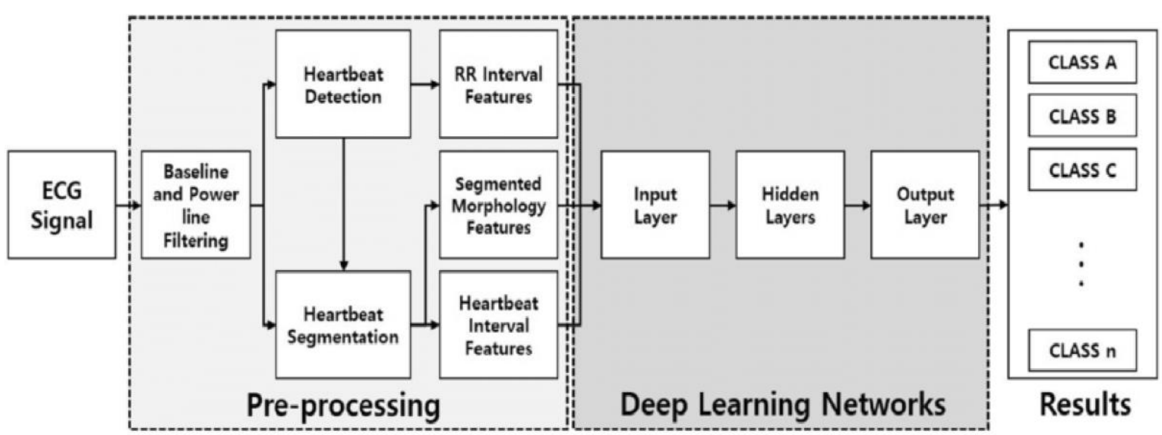
Non-functional Requirements:

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	It is a user-friendly application that allows a user to upload an ECG image to classify Arrhythmia.
NFR-2	**Security**	Data is not used for any other purposes other than processing. Only users can view their profile and personal information.
NFR-3	**Reliability**	The application is defect free, deployed with high accuracy CNN model which provides the correct prediction for the given input.
NFR-4	**Performance**	High-accuracy models are used for classification thereby increasing the performance of the application.
NFR-5	**Availability**	The application can be accessed anytime from anywhere with an internet connection.
NFR-6	**Scalability**	The system must be scalable to process multiple images. Multiple users must be able to access the system simultaneously without traffic.

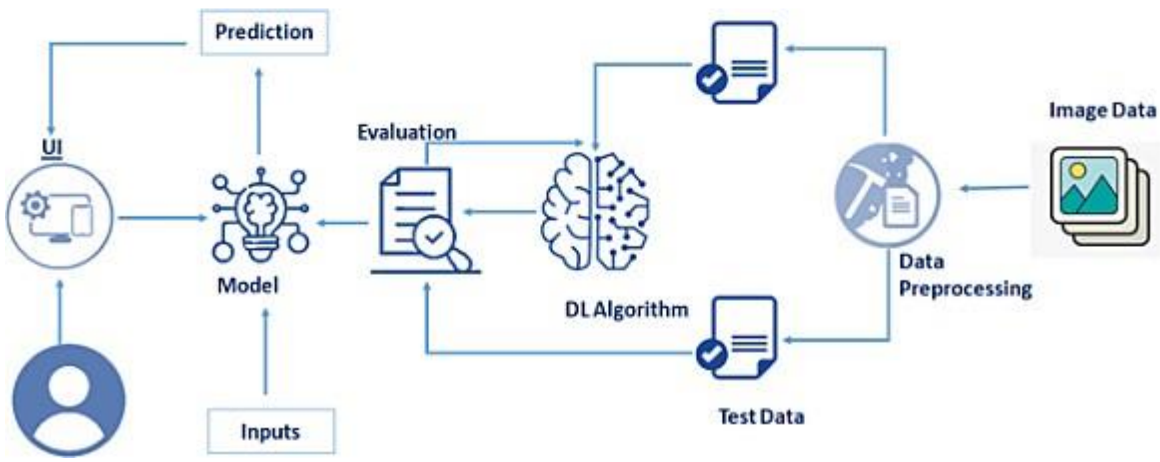
Project Design

Data Flow Diagram

Data Flow Diagram:



Solution Architecture



Technical Architecture

Technical Architecture:

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How the user interacts with the application e.g.Web UI, Mobile App.	HTML, CSS, JavaScript / Angular Js / React Js
2.	Application Logic-1	The logic for a process in the application	Python
3.	Application Logic-2	It's a symbolic math toolkit that performs a variety of tasks including deep neural network training and inference using dataflow and differentiable programming	Tensorflow
4.	Cloud Database	A global technology company that provides hardware, software, cloud-based services and cognitive computing.	IBM Cloud
5.	File Storage	Breaks up data into blocks and then stores those blocks as separate pieces, each with a unique identifier.	IBM Cloud

Table 2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Building user interfaces based on UI components.	React Js
2.	Security Implementations	OWASP is a non-profit foundation that works to improve the security of software.	OWASP
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	3-tier architecture
4.	Availability	Justify the availability of applications (e.g. use of load balancers, distributed servers, etc.)	Distributed Server
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDNs) etc.	Cache

User Stories

User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Doctor	Image Analysis	USN-1	As a doctor, I can easily assess the input and processed image data to cross verify my diagnosis of the arrhythmia type.	The image analysis is correct.	High	Sprint-1
		USN-2	As a cardiologist, I find this application very useful as it saves much time spent in diagnosing from several ECG scans of heart patients a day.	The image analysis is faster than my diagnosis.	High	Sprint-1
Patient	Dashboard	USN-3	As a patient, I can easily access the image analysis results which I can later use with other doctors.	I can quickly access my results.	Medium	Sprint-2
		USN-4	As a customer, I find the results to be easily accessible to check whether my heart is doing fine.	I find the data useful for future health check-ups.	Low	Sprint-2
Medical assistant	Result Display	USN-5	As an assistant to a cardiologist, I find it very easy to generate a report to the doctor from the results presented in the application.	The results are exactly what the doctor needs.	High	Sprint-1
		USN-6	As a medical practitioner responsible for diagnosing, my task has become incredibly fast and accurate thanks to this application.	The results are displayed quickly.	Low	Sprint-2

Project Planning And Scheduling

Sprint Planning And Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	5	Medium	Lakshman
Sprint-3		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	Medium	Kamalesh
Sprint-4		USN-3	As a user, I can register for the application through mobile OTP method	5	Low	Barath
Sprint-4		USN-4	As a user, I can register for the application through Gmail	5	Medium	Harish
Sprint-3	Login	USN-5	As a user, I can log into the application by entering email & password	3	Medium	Harish, Barath
Sprint-1	Homepage	USN-6	As a user, the homepage must properly define the Arrhythmia, its causes and effects and understand how the application helps in solving the problem.	5	High	Kamalesh
Sprint-1	More Information Page	USN-7	As a user, I must be able to comprehend all medical jargon related to Arrhythmia such as ECG, Coronary Heart Disease, Cardiomyopathy and its types.	5	High	Lakshman

Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	9	30 Oct 2022
Sprint-2	26	6 Days	31 Oct 2022	05 Nov 2022	24	07 Nov 2022
Sprint-3	9	6 Days	07 Nov 2022	12 Nov 2022	8	14 Nov 2022
Sprint-4	13	6 Days	14 Nov 2022	19 Nov 2022	12	20 Nov 2022

Coding And Solutioning

The application first requires a model that has the capability to correctly classify the arrhythmia type from an input ECG spectral image. To enable the model to do this, it is ‘trained’ with a large image dataset (of ECG scans) and ‘tested’ for accuracy. Some snapshots of this are shown below:

Image Preprocessing

In [ ]:
```
from keras.preprocessing.image import ImageDataGenerator
home_path = r'E:/IBM project/Dataset/'
```

In [ ]:
```
train_data = ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_data = ImageDataGenerator(rescale=1./255)
```

In [ ]:
```
x_train = train_data.flow_from_directory(directory = home_path + r'/train/', target_size=(64,64), batch_size=32, class_mode='categorical')
x_test = test_data.flow_from_directory(directory = home_path + r'/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.
```

In [ ]:
```
x_train.class_indices
```

Out[ ]:
```
{'Left Bundle Branch Block': 0,
 'Normal': 1,
 'Premature Atrial Contraction': 2,
 'Premature Ventricular Contractions': 3,
 'Right Bundle Branch Block': 4,
 'Ventricular Fibrillation': 5}
```

Importing Libraries

In [ ]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Convolution2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Flatten
```

Initializing the Model

In [ ]:
```
model = Sequential()
```

Adding CNN Layers

In [ ]:
```
model.add(Convolution2D(32,(3,3),input_shape = (64,64,3),activation = "relu"))
```

In [ ]:
```
model.add(MaxPooling2D(pool_size = (2,2)))
```

In [ ]:
```
model.add(Convolution2D(32,(3,3),activation='relu'))
```

In [ ]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

In [ ]:
```
model.add(Flatten())
```

In [ ]:
```
#Adding Dense Layers
```

In [ ]:
```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

In [ ]:
```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

In [ ]:
```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

In [ ]:
```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

In [ ]:
```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

In [ ]:
```
model.add(Dense(units = 6,kernel_initializer = "random_uniform",activation = "softmax"))
```

In [ ]:
```
model.summary()
```

```
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 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 flatten (Flatten)           (None, 6272)              0

 dense (Dense)               (None, 128)               802944

 dense_1 (Dense)             (None, 128)               16512

 dense_2 (Dense)             (None, 128)               16512

 dense_3 (Dense)             (None, 128)               16512

 dense_4 (Dense)             (None, 128)               16512

 dense_5 (Dense)             (None, 6)                 774
```

```
=================================================================
Total params: 879,910
Trainable params: 879,910
Non-trainable params: 0
_________________________________________________________________
```

Configure the Learning Process

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

Training the model

```
In [ ]: model.fit_generator(generator=x_train,steps_per_epoch = len(x_train), epochs=3, validation_data=x_test,validation_steps = len(x_test))

Epoch 1/3
C:\Users\slaks\AppData\Local\Temp/ipykernel_35256/318287306.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future vers
ion. Please use `Model.fit`, which supports generators.
  model.fit_generator(generator=x_train,steps_per_epoch = len(x_train), epochs=3, validation_data=x_test,validation_steps = len(x_test))
480/480 [==============================] - 249s 519ms/step - loss: 1.1194 - accuracy: 0.5638 - val_loss: 1.1833 - val_accuracy: 0.4421
Epoch 2/3
480/480 [==============================] - 85s 178ms/step - loss: 0.5511 - accuracy: 0.7900 - val_loss: 0.8991 - val_accuracy: 0.8064
Epoch 3/3
480/480 [==============================] - 78s 162ms/step - loss: 0.2468 - accuracy: 0.9252 - val_loss: 0.5509 - val_accuracy: 0.8415
Out[ ]:
```

Saving the Model

```
In [ ]: model.save('Arrhythmia.h5')

In [ ]: from tensorflow.keras.models import load_model
        from tensorflow.keras.preprocessing import image
        import numpy as np

In [ ]: model=load_model('Arrhythmia.h5')
```

Testing the Model

Testing the Model

```
In [ ]: img=image.load_img("Data/verificationPhoto.png",target_size=(64,64))
        x=image.img_to_array(img)
        x=np.expand_dims(x,axis=0)
        pred = model.predict(x)
        y_pred=np.argmax(pred)
        classifications=['left Bundle Branch block',
               'Normal',
               'Premature Atrial Contraction',
               'Premature Ventricular Contraction',
               'Right Bundle Branch Block',
               'Ventricular Fibrillation']
        result = str(classifications[y_pred])
        result

1/1 [==============================] - 0s 273ms/step
Out[ ]: 'Premature Atrial Contraction'

In [ ]:
```

Feature 1

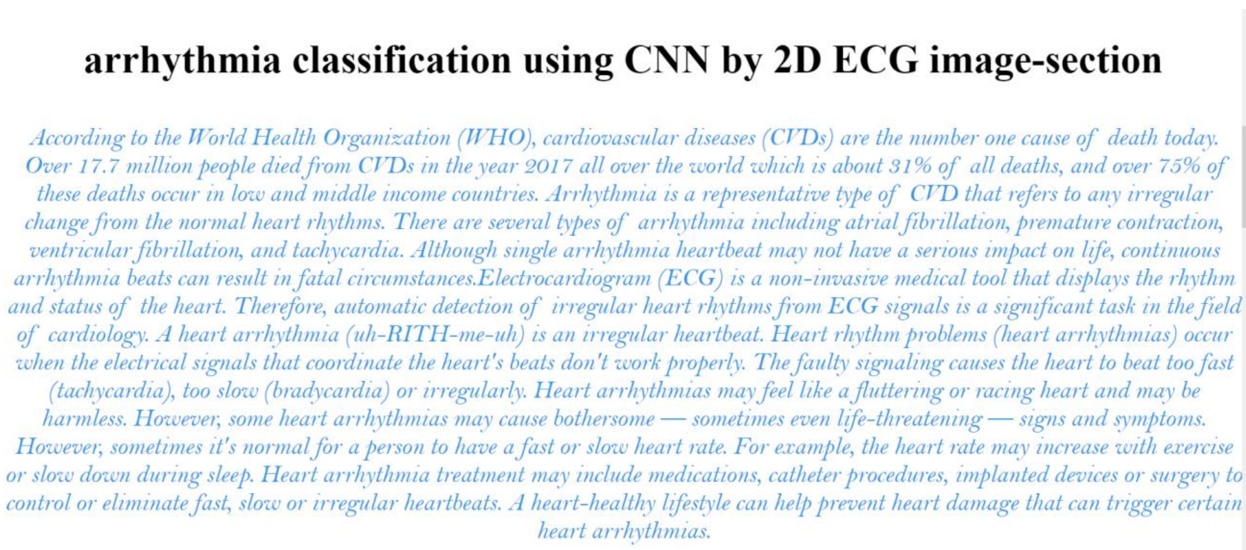
The first feature is the web page that enables users to upload ECG scan images and see results. A snapshot of the code for the webpage along with the image of it is shown below:

```
$(document).ready(function () {
    // Init
    $('.image-section').hide();
    $('.loader').hide();
    $('#result').hide();

    // Upload Preview
    function readURL(input) {
        if (input.files && input.files[0]) {
            var reader = new FileReader();
            reader.onload = function (e) {
                $('#imagePreview').css('background-image', 'url(' + e.target.result + ')');
                $('#imagePreview').hide();
                $('#imagePreview').fadeIn(650);
            }
            reader.readAsDataURL(input.files[0]);
        }
    }
    $("#imageUpload").change(function () {
        $('.image-section').show();
        $('#btn-predict').show();
        $('#result').text('');
        $('#result').hide();
        readURL(this);
    });

    // Predict
    $('#btn-predict').click(function () {
        var form_data = new FormData($('#upload-file')[0]);

        // Show loading animation
        $(this).hide();
        $('.loader').show();

        // Make prediction by calling api /predict
        $.ajax({
```

CSS Code:

```css
.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;
```

Webpage:



Feature 2

The second feature involves using python code to connect with the trained CNN model to perform the image classification procedure. A snapshot of the python code that performs this is shown below:

```
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

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

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

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

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

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


@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
```

Testing

Test Cases

Due to the application’s simplicity, the process of testing was limited to checking if the webpage was working properly or not and whether the python code ran as expected. The exact list of test cases is listed below:

Webpage Test Cases:

Test Case 1 – Functionality test; is the webpage actually working?

Fail – 3, Pass – 7, Status - OK

Test Case 2 – Speed test; is the webpage fast enough?

Fail – 2, Pass – 8, Status – OK

Python Code Test Cases:

Test Case 1 – Functionality test; is the code working as intended?

Fail – 0, Pass – 10, Status - OK

User Acceptence Testing

The user acceptance testing involved making one of the team members (user) upload an ECG image obtained from the internet onto the webpage and checking if the result is obtained and is correct (the arrhythmia type is known already).

The application worked – 10 times, out of 10 tests.

The application showed the results correctly – 9 times, out of 10 tests.

Given its perfect functionality record and very high accuracy, the user acceptance test has been passed by the user.

Results

Performance Metrics

The application works splendidly and the performance metrics are given below:

Accuracy – 90%

Functionality – 100%

Average Time Taken For Results – 2.45 seconds

User Rating – 4 stars

Advantages

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

Disadvantages

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

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, visuallyidentify the ECG signal is challenging and time-consuming.
- The advantages of the proposed CNN network have been put toevidence.
- It is endowed with an ability to effectively process the non-filtered dataset with its potential anti-noise features. Besides that, ten-foldcross-validation is implemented in this work to further demonstratethe robustness of the network.

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 studyis 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.

Appendix

References

- https://github.com/Anshuman151/ECG-Image-Based-Heartbeat-Classification-for-Arrhythmia-Detection-Using-IBM-Watson-Studio/blob/main/README.md
- https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/
- https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.convolution2dlayer.html;jsessionid=0a7e3bc26fabda07a5032030294b

Github Link:

https://github.com/IBM-EPBL/IBM-Project-23232-1659873226.git

THE END