Project Report

1. 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 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 greyscale 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

Classification of electrocardiogram (ECG) signals **plays an important role in clinical diagnosis of heart disease**. It serves as a representation of the electrical activity of the heart and is frequently employed in the search for abnormal cardiac rhythms or morphological changes. Every year, more than 300 million ECGs are taken globally2. Due to the overwhelming diagnostic workload, cardiac arrhythmias based on ECG are improperly diagnosed. Therefore, the broad digitization of ECG data and the use of automatic categorization methods have become increasingly popular. Deep neural network-based models have recently seen considerable success in the automatic classification of cardiac arrhythmia. Although most models during the training phase independently extract the intrinsic features of each lead in the 12-lead ECG, this leaves out inter-lead features. To achieve the automatic categorization of normal rhythm and eight cardiac arrhythmias, we here offer an universal model based on the two-dimensional ECG and ResNet with detached squeeze-and-excitation modules (DSE-ResNet).

The original 12-lead ECG is cut into a two-dimensional plane and rendered to look like a greyscale image. The two-dimensional ECG's internal and inter-lead features are simultaneously extracted using DSE-ResNet. The goal of 2D image classification is to recognise and represent the characteristics contained in an image as a distinct grey level in relation to the real object or kind of land cover that these features actually represent on the ground. The most crucial aspect of digital image analysis is probably picture classification. Short-time Fourier transform is used to convert the one-dimensional ECG time series signals into two-dimensional spectrograms. The 2-D CNN model is intended to extract reliable information from the input spectrograms and has four convolutional layers and four pooling layers. On an MIT-BIH arrhythmia dataset that is available to the public, our suggested methodology is assessed. We classified similar types of arrhythmias with a state-of-the-art average classification accuracy of 99.11%, which is higher than earlier published data. Performance is noteworthy in other indices as well, such as sensitivity and specificity, which shows that the suggested strategy is effective.

2. LITERATURE SURVEY

2.1. 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.

2.2. References:

(i).TITLE: Multi-model Deep Learning Ensemble for ECG Heartbeat Arrhythmia Classification

The heartbeats are classified into different arrhythmia types using two proposed deep learning models. The first model is integrating the convolutional neural network (CNN) and long short-term memory (LSTM) network to extract useful features within the ECG signal. The second model combines several classical features with LSTM in order to effectively recognize abnormal classes. These deep learning models are trained using a bagging model then aggregated by a fusion classifier to form a robust unified model.

LIMITATIONS:

(1)The most ECG databases are not specific to their clinical context.

(2)The description of the patient population in which these ECGs were obtained is lacking. This is important in interpreting the methodology and clinical utility in context

(3)The algorithms are trained based on specific environments, and the generalized methodologies are ignored.

(ii).TITLE: CardioNet: An Efficient ECG Arrhythmia Classification System Using Transfer Learning

This paper presents a novel method of heartbeat classification from ECG using deep learning. An automated system named 'CardioNet' is devised that employs the principle of transfer learning for faster and robust classification of heartbeats for arrhythmia detection. It uses pre-trained architecture of DenseNet that is trained on ImageNet dataset of millions images. The weights obtained during training of DenseNet are used to fine-tune CardioNet learning on the ECG dataset, resulting a unique system providing faster training and testing.

LIMITATIONS:

1. One of the biggest limitations of transfer learning is the problem of negative transfer.

2.Transfer learning only works if the initial and target problems are similar enough for the first round of training to be relevant.

(iii).TITLE: ECG Arrhythmia Classification By Using Convolutional Neural Network And Spectrogram

The proposed approaches operates with a large volume of raw ECG time-series data and ECG signal spectrograms as inputs to a deep convolutional neural networks (CNN). Heartbeats are classified as normal (N), premature ventricular contractions (PVC), right bundle branch block (RBBB) rhythm by using ECG signals obtained from MIT-BIH arrhythmia database. The first approach is to directly use ECG time-series signals as input to CNN, and in the second approach ECG signals are converted into time-frequency domain matrices and sent to CNN. The most appropriate parameters such as number of the layers, size and number of the filters are optimized heuristically for fast and efficient operation of the CNN algorithm. The proposed system demonstrated high classification rate for the time-series data and spectrograms by using deep learning algorithms without standard feature extraction methods.

LIMITATIONS:

1. Lack of ability to be spatially invariant to the input data.
2. Lots of training data is required.

2.3.Problem Statement Definition:

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.

In deep learning, a convolution al neural network (CNN/ConvNet) is a class of 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 that expresses how the shape of one is modified by the other.

Electrocardiogram (ECG) is mostly used for the clinical diagnosis of cardiac arrhythmia due to its simplicity, non-invasiveness, and reliability.

Recently, many models based on the deep neural networks have been applied to the automatic classification of cardiac arrhythmia with great success.ThiS model aims to realize the automatic classification of normal rhythm and 8 cardiac arrhythmias based on the 12-lead ECG records. The input *x* of the proposed model includes 2D ECG signals and basic information about the patients, and the output is the predicted labels corresponding to the normal rhythm and 8 cardiac arrhythmias.

Extracting ECG beats from the signal is important to identify arrhythmia type of the signal. In order to separate ECG signals into their heartbeats, heartbeat segmentation was applied to the signal. This project is aimed to find accurate arrhythmia detection algorithm based on heartbeat images and deep learning technique. For transforming each beat into 2-D images, image transformation was applied to the signal. After image transformation, 2-D CNN architecture was applied to the images and finally performance measures were evaluated.Our project aims at 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.

3. **IDEATION & PROPOSED SOLUTION**

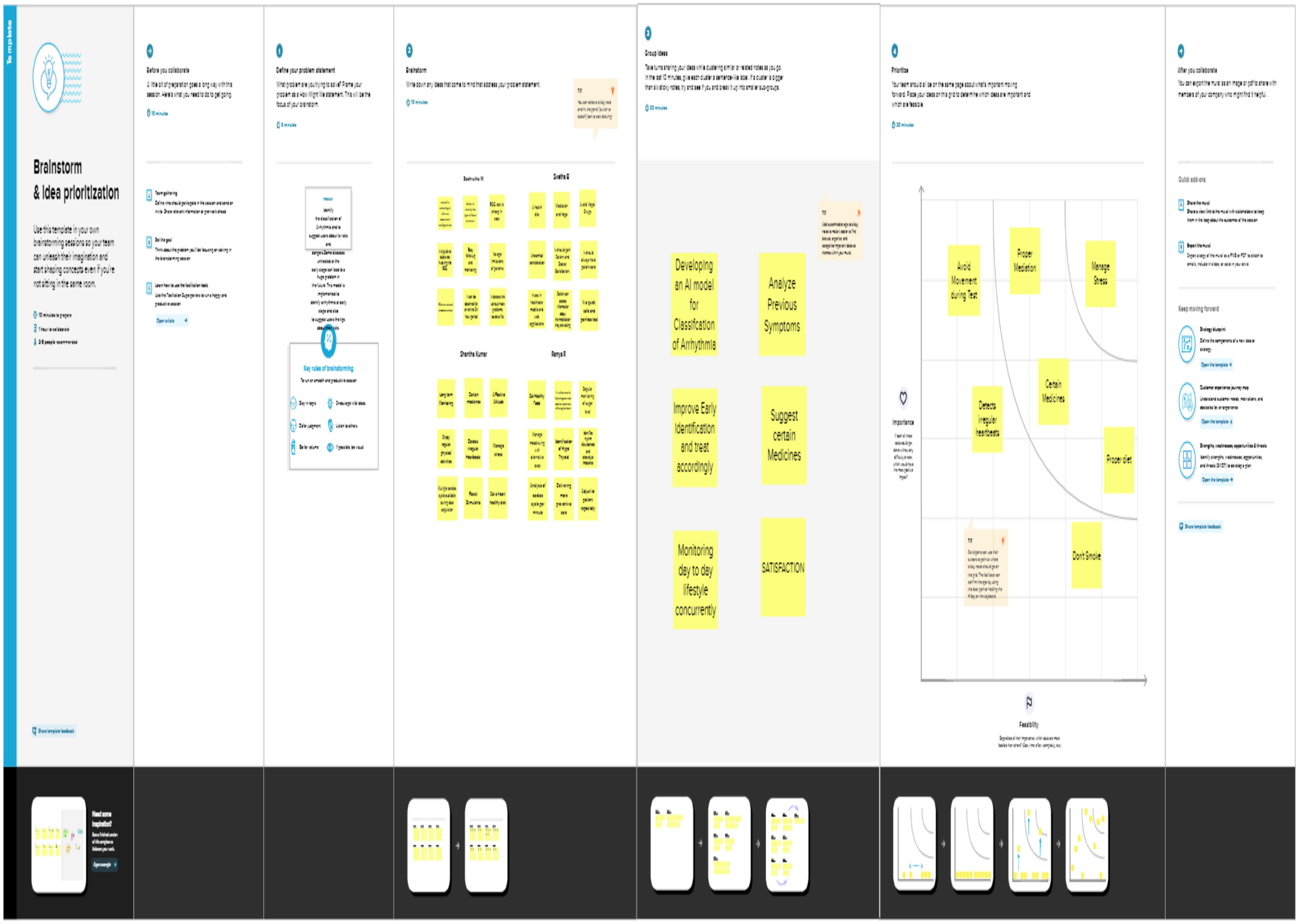
3.1 Empathy Map Canvas

An empathy map canvas serves as a foundation for outstanding user experiences, which focus on providing the experience customers want rather than forcing design teams to rely on guesswork.

Empathy map canvases help identify exactly what it is that users are looking for so brands can deliver. They can be particularly beneficial for getting teams on the same page about who users are and what they want from the brand.

3.2 Ideation & Brainstorming

Ideation and the practise of brainstorming, a particular method for coming up with fresh ideas, are frequently closely related. The main distinction between ideation and brainstorming is that whereas brainstorming is nearly often done in groups, ideation is typically seen as being more of a solitary endeavour. A group of people are frequently gathered for a brainstorming session to generate either fresh, general ideas or solutions to specific problems or circumstances.



Template
Brainstorm
& idea prioritization
Developing an AI model for Classifcation of Arrhythmia
Analyze Previous Symptoms
Improve Early Identification and treat accordingly
Suggest certain Medicines
Monitoring day to day lifestyle concurrently
SATISFACTION
Avoid Movement during Test
Proper Mediation
Manage Stress
Certain Medicines
Detects irregular heartbeats
Proper diet
Don't Smoke
Importance
Feasibility

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

> **TIP**
> You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Sashrutha M

- consult a cardiologist after an immediate configuration
- Helps to identify the type of heart diseases
- ECG test is cheap in cost
- It is quick to take a test involving the ECG.
- Easy followup and monitoting
- No age limitations of patients
- Effective internal communications
- It can be obtained for an entire 24-hour period
- It detects the various heart problems, saves a life.

Swetha B

- A health diet
- Mediation and Yoga
- Avoid illegal Drugs
- Anatomical consideration
- It should gain Patient and Doctor Satisfaction.
- It should always have patient care
- invest in healthcare mobile and web applications
- Patient can access information about the mediation they are taking
- It is quick, safe and painless test

Shantha Kumar

- Long term Monitoring
- Certain medicines
- A Positive Attitude
- Enjoy regular physical activities
- Detects Irregular Heartbeats
- Manage stress
- Multiple cardiac cycle available during data acquisition
- Resist Stimulants
- Eat a heart healthy diet

Ramya R

- Eat Healthy Food
- It is often used to help diagnose and monitor conditions affecting the heart
- Regular monitoring of sugar level
- Manage mood swing with alternative rates
- Identification of Hypo Thyroid
- Identifies rhythm disturbances and electrolyte imbalance
- Analysis of cardiac cycle per minute
- Delivering more preventive care
- Adjust to patient repository

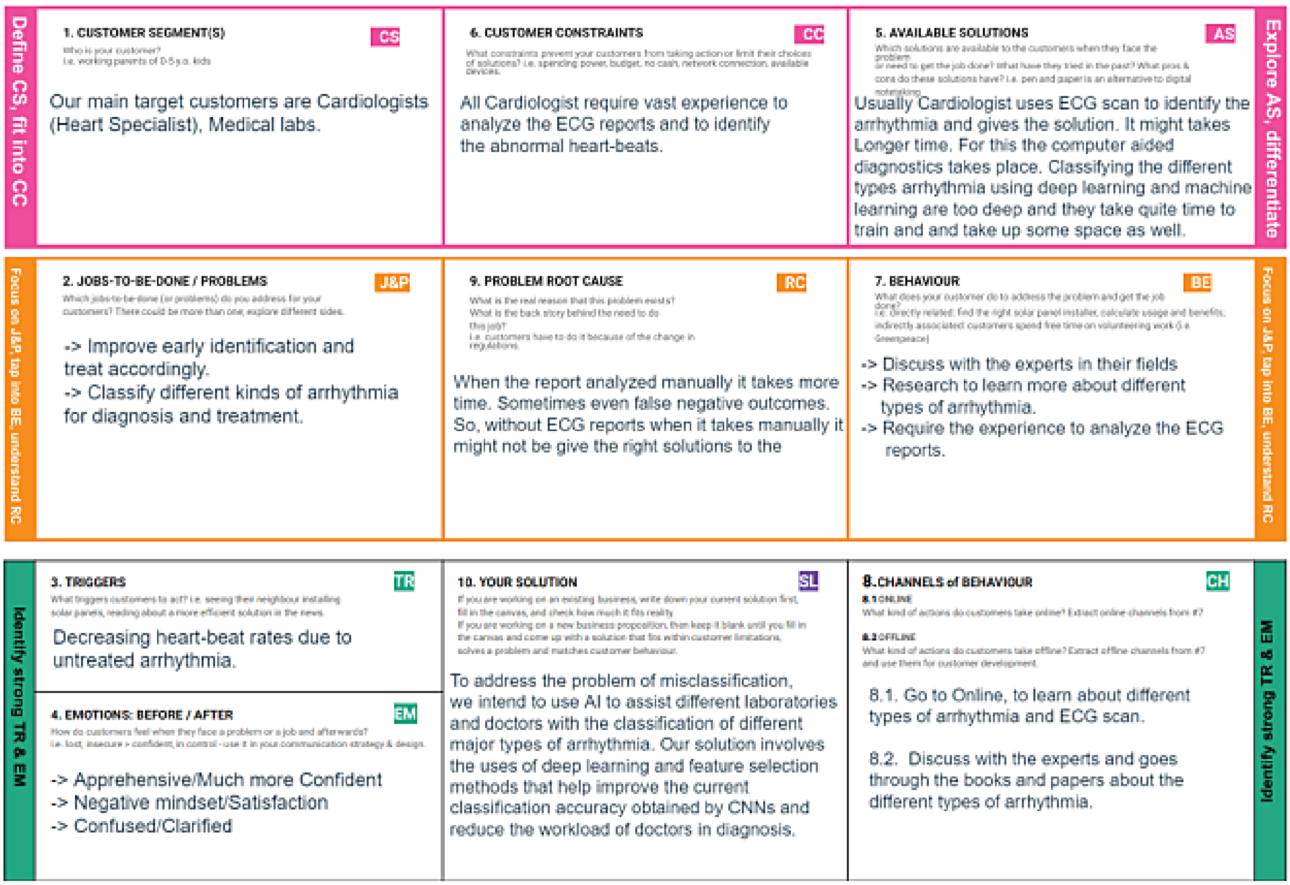
3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	An "ambulatory electrocardiogram" or an ECG) about the size of a postcard or digital camera that the patient will be using for 1 to 2 days, or up to 2 weeks. The test measures the movement of electrical signals or waves through the heart. These signals tell the heart to contract (squeeze) and pump blood. The patient will have electrodes taped to your skin. It's painless, although some people have mild skin irritation from the tape used to attach the electrodes to the chest.They can do everything but shower or bathe while wearing the electrodes. After the test period, patient will go back to see your doctor. They will be downloading the information.Arrhythmia can take many different forms, such as tachycardia, ventricular fibrillation, premature contraction, and atrial fibrillation. Continuous arrhythmia beats can lead to deadly situations, even though a single arrhythmia heartbeat may not have a serious influence on life. In this study, we develop a convolutional neural network (CNN)-based technique for electrocardiogram (ECG) arrhythmia classification. The objective is to divide ECG into seven categories—one for normal ECG and the other six for various arrhythmias—using deep two-dimensional CNN on grayscale ECG images.

1.	Idea / Solution description	We propose a model to identify and test patients for various cardiac vascular arrhythmias. This investigation pushes us to identify various types of arrhythmia using Deep Learning algorithm. Convolutional Neural Network (CNN), a DL method effective in classifying signals, is the approach we utilise in this case. CNN is used to learn features automatically from time-domain electrocardiogram signals. We propose a web application in which the classification image is chosen by the user. The image is fed into the trained model, and the mentioned class is shown on the webpage.
1.	Novelty / Uniqueness	a. The proposed model predicts image arrhythmias with a high accuracy. b. Early detection of arrhythmias enables a better understanding of the cause of the disease, initiation of therapeutic interventions and development of appropriate therapies. c. CNN takes less detection time.

1.	Social Impact / Customer Satisfaction	The feature that has been carefully modified takes the place of manually derived features, and this analysis will assist cardiologists in successfully screening patients for cardiac sickness. The ECG Dataset was used to train and test the CNN, and from the signal, seven different forms of arrhythmia were identified.
1.	Business Model (Revenue Model)	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.Hence these middle -income countries can use web application to predict arrythemia.
1.	Scalability of the Solution	It would be interesting to explore the use of optimization techniques to find a feasible design and solution.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.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

a.Functional requirement

1).Performance Requirements

- The system needs to be reliable.
- If unable to process the request, then appropriate error message.
- Our cameras are loaded within a few seconds.

2). Safety Requirements

- The details need to be maintained properly.
- Users must be authenticated.
- The face or the personal information of the user should be kept safe.

3). Security Requirements

- The details of the user must be safe and secure.
- Sharing of details.

b. Non-Functional requirements

1). Reliability:

The system reliability can be ensured with the reliable mode of delivery of requests

and responses between the front end and the back end.

2). Availability:

The availability of the system shall be ensured when the servers are started properly.

The service can be invoked only with both the application and the servers running.

3). Maintainability:

The system is also easily maintainable. The application can be enhanced without

affecting other parts of the system. The features can be added to the app that supports changes and reuse.

4). Testability:

It shows how well the system or component facilitates to perform tests to

determine whether the predefined test criteria have been met.

5. PROJECT DESIGN

a.Data Flow Diagrams

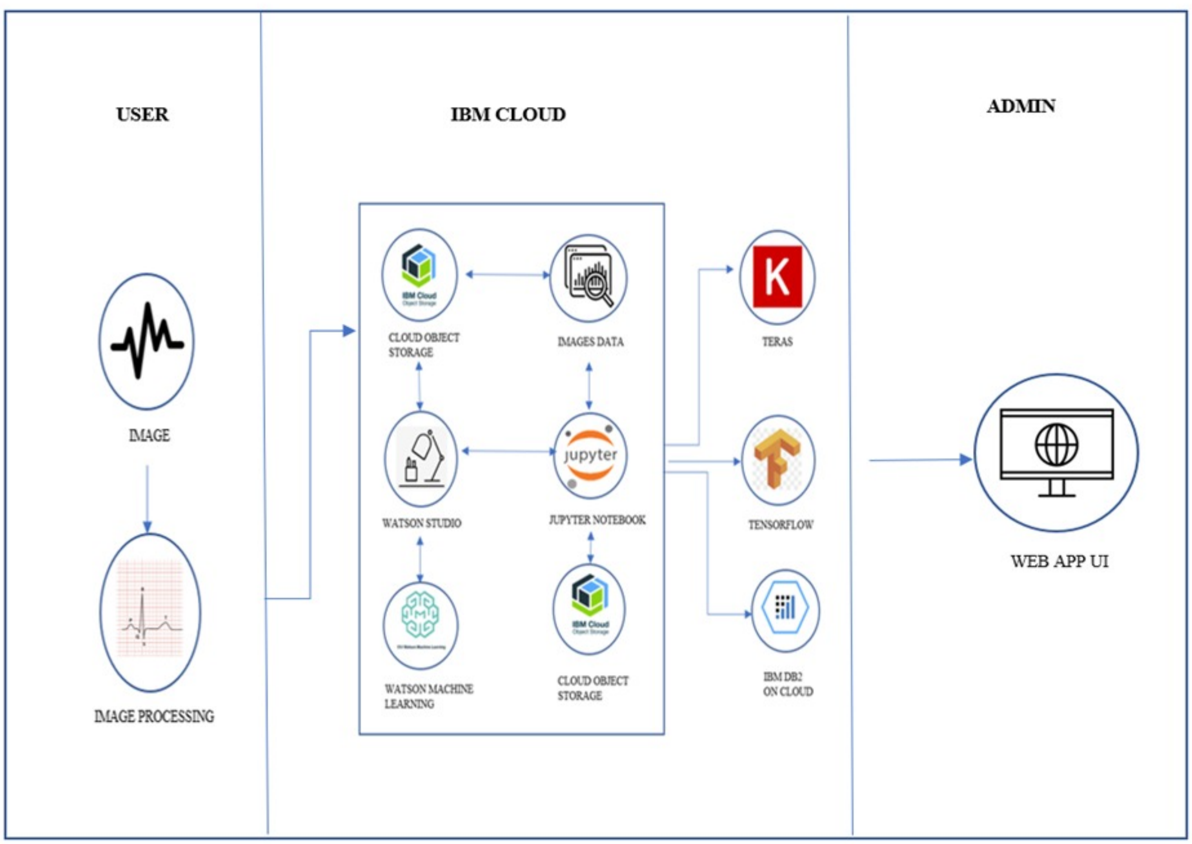
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

It shows how data enters and leaves the system, what changes the information, and where data is stored.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

b.Solution & Technical Architecture

Technical Architecture (TA) is a form of IT architecture that is used to design computer systems. It involves the development of a technical blueprint with regard to the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met.



c. User Stories

Use the below template to list all the user stories for the product.

Requirement (Epic)	User Type	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Registration	Customer (Web user)	USN-1	As a web user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-4
		USN-2	As a web user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-4
		USN-3	As a web user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-4
		USN-4	As a web user, I can register for the application through Gmail	I can register & access the dashboard with Gmail	Medium	Sprint-4
Login		USN-5	As a web user, I can log into the application by entering email & password	I can access my account	High	Sprint-4
Dashboard		USN-6	As a web user, I can view the picture on the webpage once I uploaded it.	I get and “Upload Successful” pop-up	Medium	Sprint-4
		USN-7	As a web user,I can	Result is	High	Sprint-4

			view the results for the image I uploaded.	displayed.		
Login	Customer Care Executive	USN-1	As a Customer Care Executive, I can login with my credentials.	I can access my account	High	Sprint-4
Dashboard		USN-2	As a Customer Care Executive, I can see all the information, I can view the picture on the webpage once I uploaded it	I get and “Upload Successful” pop-up	High	Sprint-4
Responsiblities		USN-3	As a Customer Care Executive, I am able to resolve the customers’ complaints.	Manage and resolve complaints	High	Sprint-4
Login	Administrator	USN-1	As an administartor, I can login with my credentials.	I can access my account/dashboard	High	Sprint-4
Dashboard		USN-2	As an administartor, I can see all the information, I can view the picture on the webpage once I uploaded it.	I can see all the information in the dashboard.	Low	Sprint-4
Responsibilities		USN-3	As an administrator, I can implement security measures and review web content	I can implement security measures.	High	Sprint-4

6. **PROJECT PLANNING & SCHEDULING**

Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation

Use the below template to create product backlog and sprint schedule

Sprint	User Story Number	Functional Requirement (Epic)	User Story / Task	Story Points	Priority	Team Members
Sprint-1	USN-1	Dataset	Downloaded the dataset.	2	High	Sashrutha .M
Sprint-1	USN-2		Image preprocessing	1	High	Ramya.R
Sprint-1	USN-3		Import the ImageDataGenerator library	2	Low	Shantha Kumar.S
Sprint-1	USN-4		Configure ImageDataGenerator class	2	Medium	Swetha.B
Sprint-1	USN-5		Apply ImageDataGenerator functionality to trainingset and testingset.	1	High	Sashrutha .M
Sprint -2	USN-6		Import the libraries	1	Medium	Shantha Kumar.S
	USN-7		Initialise the model	2	Low	Ramya.R
	USN-9		Add the CNN layers	2	High	Swetha.B
	USN-10		Add Dense	2	Low	Shantha

			layers			Kumar.S
	USN-11		Configure the learning process	1	Low	Ramya.R
	USN-12		Train the model	3	High	Sashrutha .M
	USN-13		Save the model	2	Medium	Shantha Kumar.S
	USN-14		Test the model	3	High	Sashrutha .M
Sprint-3	USN-15		Create HTML files	2	Medium	Swetha.B
	USN-16		Build the python Code	2	High	Sashrutha .M
	USN-17		Run the App	2	Medium	Ramya.R
Sprint-4	USN-18		Create the IBM cloud account	2	Medium	Swetha.B
	USN-19		Train the model on IBM Watson	4	High	Sashrutha .M

Project Tracker, Velocity & Burndown Chart

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Comp (as on Planned End D
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	8

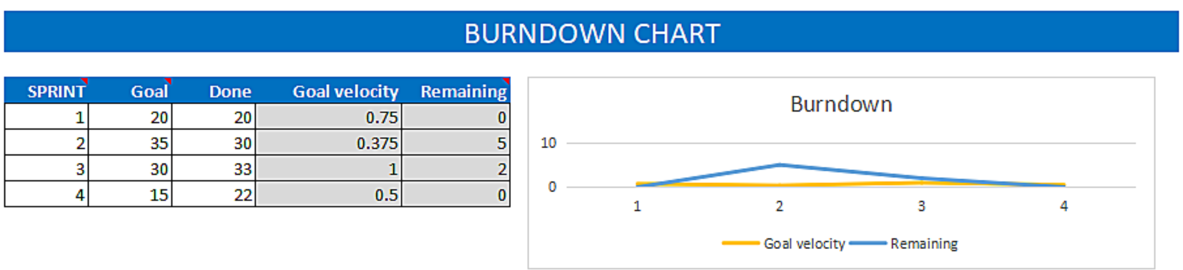
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	16
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	6
Sprint-4	20	3 Days	13 Nov 2022	15 Nov 2022	6

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

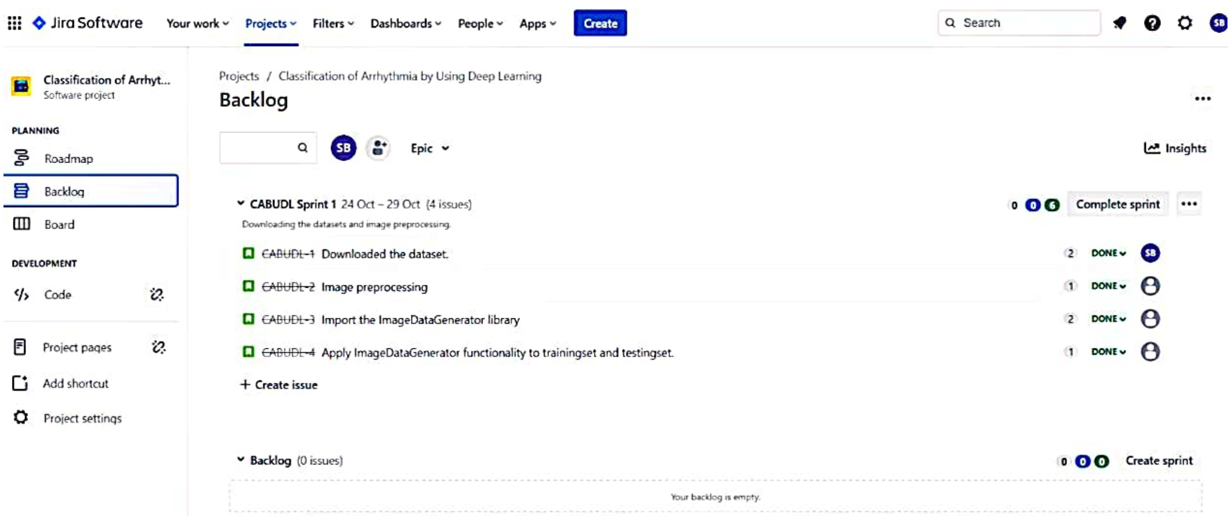
Burndown Chart:

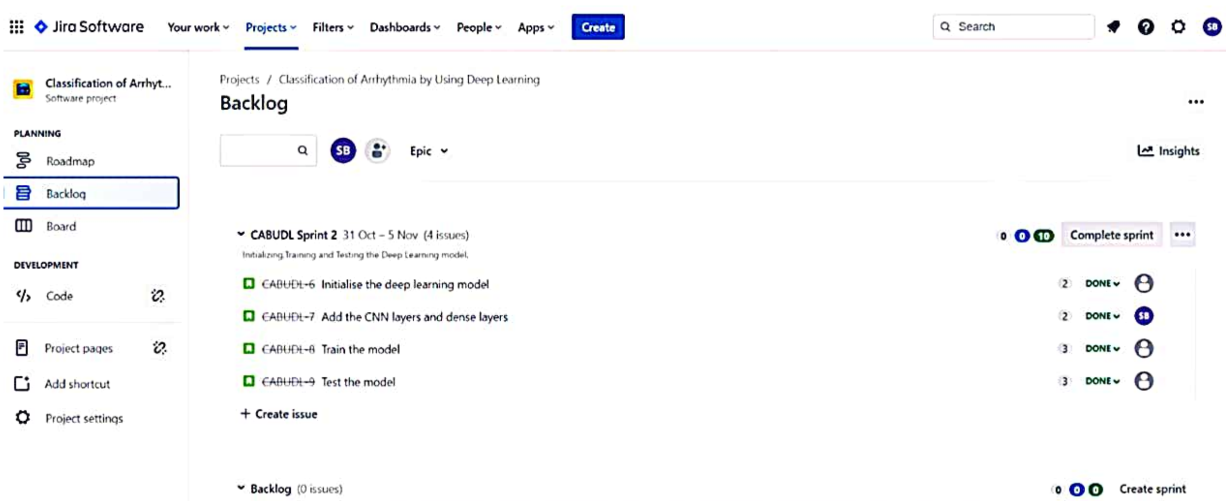
A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

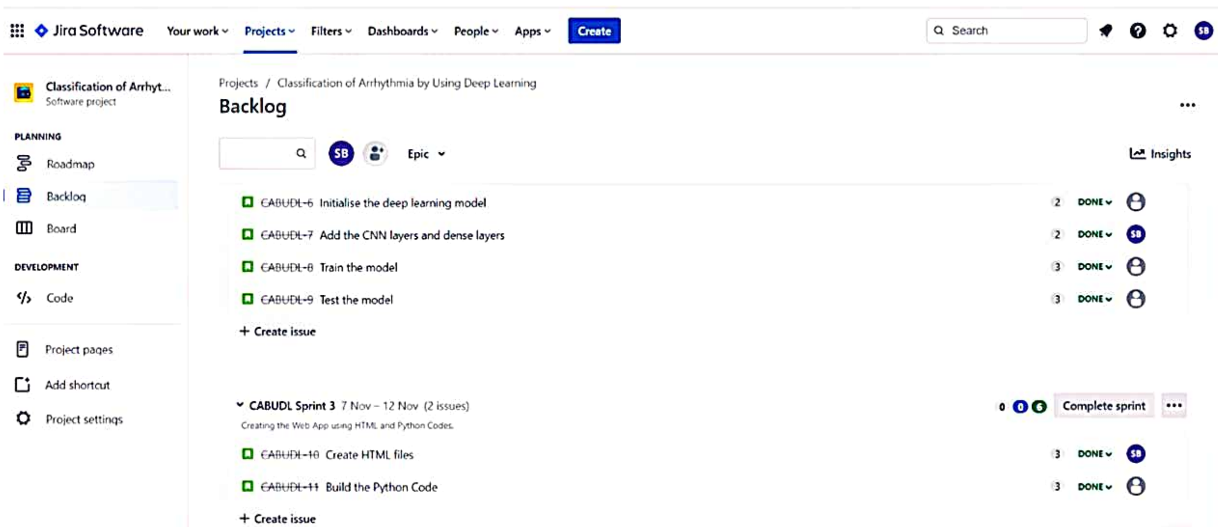

BURNDOWN CHART

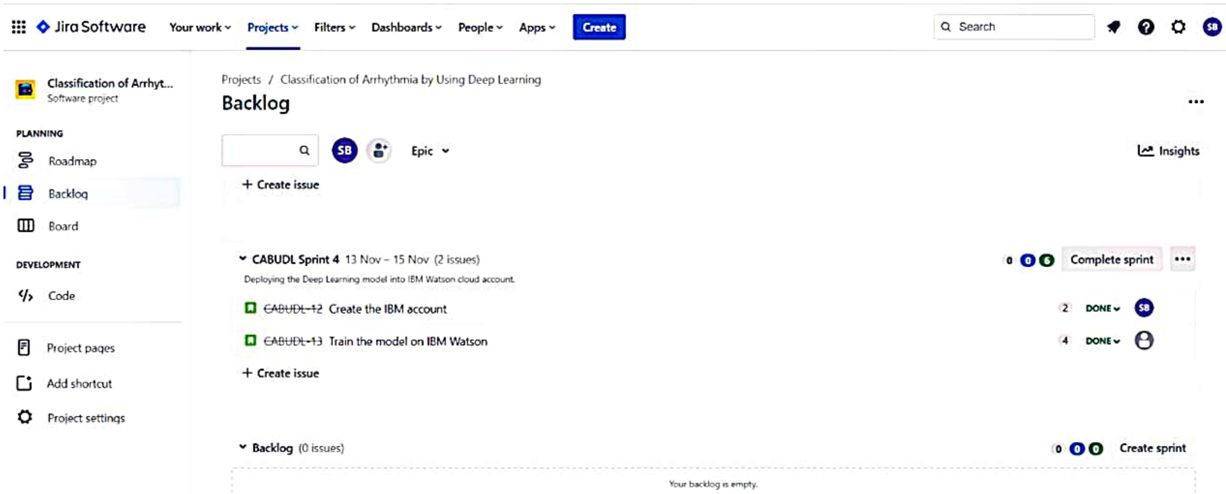
SPRINT	Goal	Done	Goal velocity	Remaining
1	20	20	0.75	0
2	35	30	0.375	5
3	30	33	1	2
4	15	22	0.5	0

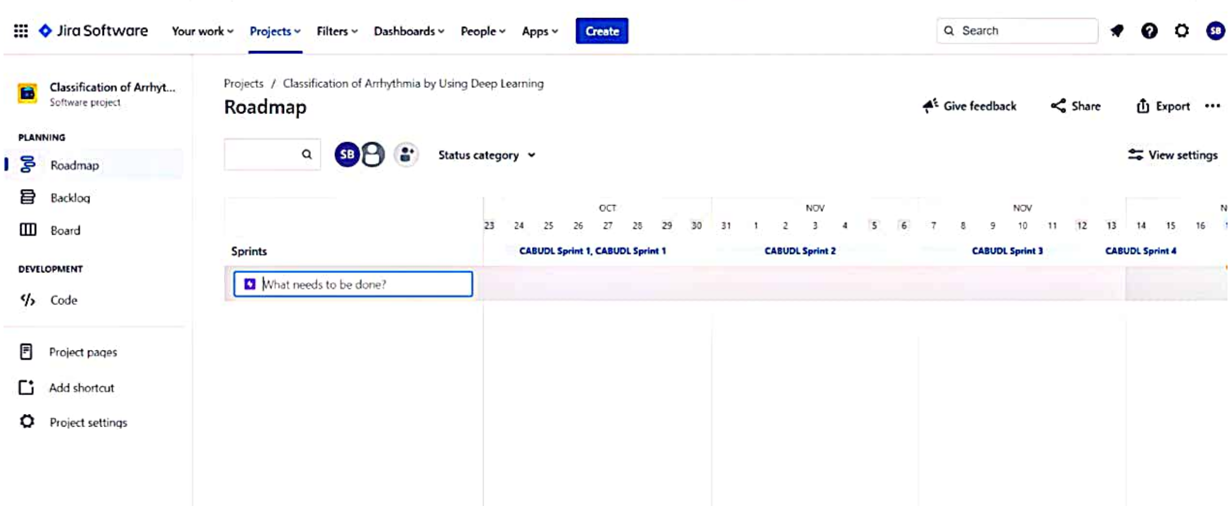
Reports from JIRA











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

In this project, we have deployed our training model using CNN on IBM Watson studio and in our local machine. We are deploying 4 types of CNN layers in a sequential manner , starting from : Convolutional layer 2D:A 2-D convolutional layer applies sliding convolutional filters to 2-D input. The layer convolves the input by moving the filters along the input vertically and horizontally and computing the dot product of the weights and the input, and then adding a bias term. Pooling Layer :Pooling layers are used to reduce the dimensions of the feature maps. Thus, it reduces the number of parameters to learn and the amount of computation performed in the network. The pooling layer summarises the features present in a region of the feature map generated by a convolution layer. Fully-Connected layer :After extracting features from multiple convolution layers and pooling layers, the fully-connected layer is used to expand the connection of all features. Finally, the SoftMax layer makes a logistic regression classification. Fully-connected layer transfers the weighted sum of the output of the previous layer to the activation function. Dropout Layer :There is usually a dropout layer before the fully-connected layer. The dropout layer will temporarily disconnect some neurons from the network according to the certain probability during the training of the convolution neural network, which reduces the joint adaptability between neuron nodes, reduces overfitting, and enhances the generalization ability of the network.

Dataset Collection: The datset has been downloaded from the following link.

https://drive.google.com/file/d/16SUrk6lMaakmVf4axGNDub3joHl-XdBT/view

The dataset contains six classes:

Left Bundle Branch Block

Normal

Premature Atrial Contraction

Premature Ventricular Contractions

Right Bundle Branch Block

Ventricular Fibrillation

Image Preprocessing: Image Pre-processing includes the following main tasks Import 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.

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.

Applying ImageDataGenerator functionality to the trainset and test set: We will apply ImageDataGenerator functionality to Trainset and Testset 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}

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.

IMAGE PREPROCESSING

```
In [5]:
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```

Importing the ImageDataGenerator Class

```
In [6]:
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

Configuring and applying the ImageDataGenerator class to test and train dataset

```
In [7]:
train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoom_range = 0.2,horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)
```

```
In [8]:
x_train = train_datagen.flow_from_directory("/content/data/train",target_size = (64,64),batch_size = 32,class_mode = "categorical")
x_test = test_datagen.flow_from_directory("/content/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.
```

```
In [9]:
x_train.class_indices

Out[9]:
{'Left Bundle Branch Block': 0,
 'Normal': 1,
 'Premature Atrial Contraction': 2,
 'Premature Ventricular Contractions': 3,
 'Right Bundle Branch Block': 4,
 'Ventricular Fibrillation': 5}
```

Model Building We are ready with the augmented and pre-processed image data,we will begin our build our model by following the below steps: Import the model building Libraries:

Initializing 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: 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 downsample the input.

The flatten layer flattens the input.

Adding Hidden Layers: Dense layer is deeply connected neural network layer. It is most common and frequently used layer.

MODEL BUILDING

Importing the necessary libraries

In [12]:
```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```

In [10]:
```
model = Sequential()
```

Adding CNN layers

In [11]:
```
model.add(Convolution2D(32,(3,3),input_shape = (64,64,3),activation = "relu"))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Convolution2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
```

Adding Dense Layers

In [13]:
```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
model.add(Dense(units = 6,kernel_initializer = "random_uniform",activation = "softmax"))
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
_________________________________________________________________
```

Adding Output Layer:

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.

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.

Training the model: We will train our model with our image dataset. fit_generator functions used to train a deep learning neural network.

Configure the learning process

```
In [15]: model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

```
In [14]: model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
         model.fit_generator(generator=x_train,steps_per_epoch = len(x_train), epochs=9, validation_data=x_test,validation_steps = len(x_test))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future versi
on. Please use `Model.fit`, which supports generators.

Epoch 1/9
480/480 [==============================] - 113s 233ms/step - loss: 1.4495 - accuracy: 0.4787 - val_loss: 1.6448 - val_accuracy: 0.3193
Epoch 2/9
480/480 [==============================] - 103s 215ms/step - loss: 1.1114 - accuracy: 0.5711 - val_loss: 1.4731 - val_accuracy: 0.4845
Epoch 3/9
480/480 [==============================] - 99s 206ms/step - loss: 0.7514 - accuracy: 0.7075 - val_loss: 0.8935 - val_accuracy: 0.6825
Epoch 4/9
480/480 [==============================] - 99s 207ms/step - loss: 0.4033 - accuracy: 0.8554 - val_loss: 0.7506 - val_accuracy: 0.7764
Epoch 5/9
480/480 [==============================] - 99s 205ms/step - loss: 0.2274 - accuracy: 0.9291 - val_loss: 0.4675 - val_accuracy: 0.8697
Epoch 6/9
480/480 [==============================] - 101s 211ms/step - loss: 0.1619 - accuracy: 0.9495 - val_loss: 0.4392 - val_accuracy: 0.8664
Epoch 7/9
480/480 [==============================] - 99s 206ms/step - loss: 0.1333 - accuracy: 0.9596 - val_loss: 0.4400 - val_accuracy: 0.8686
Epoch 8/9
480/480 [==============================] - 99s 205ms/step - loss: 0.1070 - accuracy: 0.9669 - val_loss: 0.6352 - val_accuracy: 0.8462
Epoch 9/9
480/480 [==============================] - 101s 210ms/step - loss: 0.0979 - accuracy: 0.9683 - val_loss: 0.5050 - val_accuracy: 0.8636
Out[14]:
```

Saving 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.

Saving the model

```
In [16]: model.save('ECG.h5')

In [17]: from tensorflow.keras.models import load_model
         from tensorflow.keras.preprocessing import image

In [18]: model=load_model('ECG.h5')

In [26]: img=image.load_img("/content/data/test/Ventricular Fibrillation/VFEfig_125.png",target_size=(64,64))

In [27]: x=image.img_to_array(img)

In [28]: import numpy as np

In [29]: x=np.expand_dims(x,axis=0)
```

Testing the model: Load necessary libraries and load the saved model using load_model 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.

Testing the model

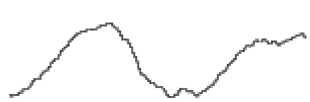
```
In [30]: pred = model.predict(x)
         y_pred=np.argmax(pred)
         y_pred

1/1 [==============================] - 0s 20ms/step
Out[30]: 5

In [31]: index=['left Bundle Branch block',
               'Normal',
               'Premature Atrial Contraction',
               'Premature Ventricular Contraction',
               'Right Bundle Branch Block',
               'Ventricular Fibrillation']
         result = str(index[y_pred])
         result

Out[31]: 'Ventricular Fibrillation'
```

The unknown image uploaded is:



Here the output for the uploaded result is Ventricular Fibrillation.

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: We use HTML to create the front end part of the web page. Here, we created 5 html pages- home.html, predict_base.html, predict.html, information.html and types.html

home.html displays the home page.

information.html displays all important details to be known about ECG.

predict-base.html and predict.html accept input from the user and predicts the values

types.html displays the types of arrythmia.

Building server-side script: We will build the flask file 'app.py' which is a web framework written in python for server-side scripting. 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.**

Running The App:

Navigate to the localhost (http://127.0.0.1:5000/)where you can view your web page.

8. TESTING

a.Test Cases

- Usability Testing - this is how well the user can access the different

features in the system and how easy it is to use.

- GUI Software Testing - this is to check if graphically that the program

looks how it was intended and the GUI works as intended.

- Security testing - this would be to check if important information is secure

and if there are certain access restrictions that they work.

- Accessibility - how easy is it for various users including users with

disabilities to use the system.

- Reliability Testing - to check that the system works for a long period of

time and does not constantly crash.

b. User Acceptance Testing

Acceptance testing is a test conducted to determine if the requirements of a specification or contract are met. It may involve chemical tests, physical tests, or performance tests. In the case of software, acceptance testing performed by the customer is known as user acceptance testing (UAT), end-user testing, site(acceptance) testing, or field (acceptance) testing.

9. RESULTS

Performance Metrics

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

1.The flexibility that 2-D CNN models offer in terms of data augmentation is an advantage of employing them. Data augmentation is not advantageous for 1-D ECG signals since it might alter the meaning of the data. The CNN model can, however, learn the different types of data with 2-D spectrograms, and augmentation aids in expanding the pool of training data.

2.Its capacity to assist doctors in making clinical choices on ECG recordings in a reliable manner. Additionally, it was designed to be as straightforward as possible while providing the best performance.

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.

DISVANTAGES:

1.The data is too large so it is complex to use.
2.Needs High Performing systems.
Not useful for identifying the different stages of Arrhythmia disease.
Not useful in monitoring motor symptoms

11. CONCLUSION

A deep learning model which is obtaining in this model, which has high success rate and rapid querying can help in the diagnosis of arrhythmia. An accurate taxonomy of ECG signals is extremely helpful in the prevention and diagnosis of CVDs. Deep CNN has proven useful in enhancing the accuracy of diagnosis algorithms in the fusion of medicine and modern machine learning technologies. Unlike traditional methods which is used 1D ECG signal input to neural networks, we have been used 2D ECG images as input data. Therefore, there is no need to apply pre-signal processing methods to the input data. Furthermore, the effect of the noise in the signal is minimized. In addition, the obtained ECG images are sent to the networks in a single-decker depth, thus, avoiding complex network structures. The deep learning results show that proposed CNN architecture accurate for classification these arrhythmia types. In addition, this developed deep learning model can quickly and accurately query for real-time arrhythmia diagnosis.

FUTURE SCOPE

In future studies, the deep learning model can be expanded by increasing the types and numbers of arrhythmias. The proposed scheme can help experts diagnose CVDs by referring to the automated classification of ECG signals. The present research uses only a single-lead ECG signal. The effect of multiple lead ECG data to further improve experimental cases will be studied in future work.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.

12. APPENDIX

Source Code

IMAGE PREPROCESSING

In [5]:

```
from keras.models import Sequential
from keras.layers import Dense
```

```
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```
Importing the ImageDataGenerator Class

In [6]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```
Configuring and applying the ImageDataGenerator class to test and train dataset

In [7]:
```
train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoom_range = 0.2,horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)
```
In [8]:
```
x_train = train_datagen.flow_from_directory("/content/data/train",target_size = (64,64),batch_size = 32,class_mode = "categorical")
x_test = test_datagen.flow_from_directory("/content/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.

In [9]:
```
x_train.class_indices
```
Out[9]:
```
{'Left Bundle Branch Block': 0,
 'Normal': 1,
 'Premature Atrial Contraction': 2,
 'Premature Ventricular Contractions': 3,
 'Right Bundle Branch Block': 4,
 'Ventricular Fibrillation': 5}
```
MODEL BUILDING

Importing the necessary libraries

In [12]:

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```

In [10]:

```
model = Sequential()
```

Adding CNN layers

In [11]:

```
model.add(Convolution2D(32,(3,3),input_shape = (64,64,3),activation =
"relu"))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Convolution2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
```

Adding Dense Layers

In [13]:

```
model.add(Dense(units = 128,kernel_initializer =
"random_uniform",activation = "relu"))
model.add(Dense(units = 128,kernel_initializer =
"random_uniform",activation = "relu"))
model.add(Dense(units = 128,kernel_initializer =
"random_uniform",activation = "relu"))
model.add(Dense(units = 128,kernel_initializer =
"random_uniform",activation = "relu"))
model.add(Dense(units = 128,kernel_initializer =
"random_uniform",activation = "relu"))
model.add(Dense(units = 6,kernel_initializer =
"random_uniform",activation = "softmax"))
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 [15]:

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

In [14]:

```
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=
['accuracy'])
model.fit_generator(generator=x_train,steps_per_epoch = len(x_train),
epochs=9, validation_data=x_test,validation_steps = len(x_test))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2:
UserWarning: `Model.fit_generator` is deprecated and will be removed in
a future version. Please use `Model.fit`, which supports generators.

Epoch 1/9
480/480 [==============================] - 113s 233ms/step -
loss: 1.4495 - accuracy: 0.4787 - val_loss: 1.6448 - val_accuracy: 0.3193
Epoch 2/9
480/480 [==============================] - 103s 215ms/step -
loss: 1.1114 - accuracy: 0.5711 - val_loss: 1.4731 - val_accuracy: 0.4845
Epoch 3/9
480/480 [==============================] - 99s 206ms/step -
loss: 0.7514 - accuracy: 0.7075 - val_loss: 0.8935 - val_accuracy: 0.6825
Epoch 4/9
480/480 [==============================] - 99s 207ms/step -
loss: 0.4033 - accuracy: 0.8554 - val_loss: 0.7506 - val_accuracy: 0.7764
Epoch 5/9
480/480 [==============================] - 99s 205ms/step -
loss: 0.2274 - accuracy: 0.9291 - val_loss: 0.4675 - val_accuracy: 0.8697
Epoch 6/9
480/480 [==============================] - 101s 211ms/step -
loss: 0.1619 - accuracy: 0.9495 - val_loss: 0.4392 - val_accuracy: 0.8664
Epoch 7/9
480/480 [==============================] - 99s 206ms/step -
loss: 0.1333 - accuracy: 0.9596 - val_loss: 0.4400 - val_accuracy: 0.8686
```

```
Epoch 8/9
480/480 [==============================] - 99s 205ms/step -
loss: 0.1070 - accuracy: 0.9669 - val_loss: 0.6352 - val_accuracy: 0.8462
Epoch 9/9
480/480 [==============================] - 101s 210ms/step -
loss: 0.0979 - accuracy: 0.9683 - val_loss: 0.5050 - val_accuracy: 0.8636
Out[14]:
```

Saving the model

In [16]:

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

In [17]:

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```

In [18]:

```
model=load_model('ECG.h5')
```

In [26]:

```
img=image.load_img("/content/data/test/Ventricular
Fibrillation/VFEfig_125.png",target_size=(64,64))
```

In [27]:

```
x=image.img_to_array(img)
```

In [28]:

```
import numpy as np
```

In [29]:

```
x=np.expand_dims(x,axis=0)
```

Testing the model

In [30]:

```
pred = model.predict(x)
y_pred=np.argmax(pred)
y_pred
1/1 [==============================] - 0s 20ms/step
```

Out[30]:

5

In [31]:

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

Out[31]:

'Ventricular Fibrillation'

In [ ]:

Home.html:

```
<!DOCTYPE html>
<html>
<head>
<title>Home</title>
<style>
body{
  background-color:lightblue;

 background-repeat:no-repeat;
   padding:0;
   margin:0;
}
.image {
```

```
background:url('Arrthmyia.jpg');
background-size:cover;
}
.navbar
{
margin: 0px;
padding:20px;
background-color:white;
opacity:0.6;
color:yellow;
font-family:'Roboto',sans-serif;
font-style: italic;
border-radius:20px;
font-size:25px;
}

a:hover{
background-color:blue;
color:red;
border-radius:15px;0
font-size:30px;
padding-left:10px;
}
a{
color:red;
float:right;
text-decoration:none;
font-style:normal;
padding-right:20px;}
p{
font-color:"black";
font-style:normal;
font-size:30px;
font-family="Times New Roman"
```

```
}
</style>
</head>
<body class="image">
<div class="navbar">
<a href="/upload" >Predict</a>
<a href="/info">Info</a>
<a href="/types">Types</a>
<a href="/about">Home</a>
<br>
</div>
<center><b class="pd"><font color="white" size="15" font-family="Times New Roman" >ECG Arrhythmia classification</font></b></center>
<div>
<br>
<center>
<p><font color="white" style="text-align:center;" >A heart arrhythmia is an irregular heartbeat. Heart rhythm problems (heart arrhythmias) occur<br> when the electrical signals that coordinate the heart's beats don't work properly.<br> The faulty signaling causes the heart to beat too fast (tachycardia), too slow (bradycardia) or irregularly.<br>

Heart arrhythmias may feel like a fluttering or racing heart and may be harmless.<br> A heart-healthy lifestyle can help prevent heart damage that can trigger certain heart arrhythmias.
</p>
<font color="black" size="5">
<head>
<style>
div.container {
  text-align: center;
}

ol.myOL {
```

```
  display: inline-block;
  text-align: left;
}
</style>
</head>
<body>
</body>
 </font>
</div>
</body>
</html>
```

Info.html:

```
<!DOCTYPE html>
<html>
<head>
<title>Information</title>
</head>
<frameset rows = "7%,20%, 60%">
<frame src="frame_3.html">
<frame src="frame_1.html">
<frame src="frame_2.html">
</frameset>
</html>
```

Types.html:

```
<!DOCTYPE html>
<html>
<head>
<title>Types</title>
<style>
p,ol,ul{
```

```
  color: black;
  text-indent: 30px;
  font-family: "Times New Roman";
font-size:20px;
}
div.container {
  text-align: center;
}
ol.myUL {
  display: inline-block;
  text-align: left;
}
.navbar
{
margin: 0px;
padding:20px;
background-color:white;
opacity:0.6;
color:yellow;
font-family:'Roboto',sans-serif;
font-style: italic;
border-radius:20px;
font-size:25px;
}

a:hover{
background-color:blue;
color:red;
border-radius:15px;0
font-size:30px;
padding-left:10px;
}
a{
color:red;
```

```
float:right;
text-decoration:none;
font-style:normal;
padding-right:20px;}
</style>
</head>
<body style="background-color:Aqua">
<div class="navbar">
<a href="/upload" >Predict</a>
<a href="/info">Info</a>
<a href="/types">Types</a>
<a href="/about">Home</a>
<br>
</div><br>
<center><b class="pd"><font color="black" size="25" font-family="Times New Roman" >Types Of Arrhythmia</font></b></center><br><br>
<center><img src="types.jpeg" width="800" height="900"></img></center>
<p>Arrhythmia describes a group of conditions that affect the heart’s natural rhythm. Different types of arrhythmias cause the heart to beat too fast, too slowly, or in an irregular pattern.
</p>
<center>
<p style="font-size:25px"><b>Types of arrhythmia</b> </p>
<div class="container">
  <ol class="myUL">
<li><b>Supraventricular arrhythmias:</b> Arrhythmias that begin in the atria (the heart’s upper chambers).<br>
           
    "Supra” means above. “Ventricular” refers to the lower chambers of the heart or ventricles.</li>
<br><li><b>Ventricular arrhythmias:</b> Arrhythmias that begin in the ventricles (the heart’s lower chambers).</li>
<br><li><b>Bradyarrhythmias:</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.

</div>
</center>
<p>1.What are the types of supraventricular arrhythmias?

Supraventricular arrhythmias begin in the atria or the upper chambers of your heart.
Types of supraventricular arrhythmias include:</p>

Paroxysmal supraventricular tachycardia (PSVT): A rapid but regular heart rhythm that comes from the atria. This type of arrhythmia begins and ends suddenly.

 Accessory pathway tachycardias (bypass tract tachycardias): A fast heart rhythm caused by an extra, abnormal electrical pathway or connection between the atria and ventricles. The impulses travel through the extra pathways as well as the usual route. This allows the impulses to travel around your heart very quickly, causing it to beat unusually fast (example: Wolff- Parkinson-White syndrome).

 AV nodal re-entrant tachycardia (AVNRT): A fast heart rhythm caused by the presence of more than one pathway through the atrioventricular (AV) node.

 Atrial tachycardia: A rapid heart rhythm that starts in the atria.

 Atrial fibrillation: A very common irregular heart rhythm. This happens when many impulses begin and spread through the atria, competing for a chance to travel through the AV node. This results in a disorganized rapid and irregular rhythm. Because the impulses are traveling through the atria in a disorderly fashion, there’s a loss of coordinated atrial contraction.

 Atrial flutter: An atrial arrhythmia caused by one or more rapid circuits in the atrium. Atrial flutter is usually more organized and regular than atrial fibrillation.

<p>2.What are the types of ventricular arrhythmias?

A ventricular arrhythmia begins in the heart’s ventricles. Types of ventricular arrhythmias include:</p>

Premature ventricular contractions (PVCs): Early, extra heartbeats that start out in the ventricles. Most of the time, PVCs don’t cause any symptoms or require treatment. This type of arrhythmia is common and can be related to stress, too much caffeine or nicotine, or exercise. They can be also be caused by heart disease or electrolyte imbalance. People who have several PVCs and/or symptoms associated with them should be evaluated by a cardiologist (heart doctor).

Ventricular tachycardia (V-tach): A rapid heartbeat that begins in the ventricles. The rapid rhythm keeps the heart from adequately filling with blood, and less blood is able to pump through the body. V-tach can be serious, especially in people with heart disease, and may be associated with more symptoms than other types of arrhythmia. A cardiologist should evaluate this condition.

Ventricular fibrillation (V-fib): An erratic, disorganized firing of impulses from the ventricles. The ventricles quiver and can’t generate an effective contraction, which results in a lack of blood being delivered to your body. This is a medical emergency that must be treated with cardiopulmonary resuscitation (CPR) and defibrillation (delivery of an energy shock to your heart muscle to restore a normal rhythm) as soon as possible.

Long QT: While this is not an arrhythmia, it can predispose someone to have one. The QT interval is the area on the ECG that represents the time it takes for the heart muscle to contract and then recover, or for the electrical impulse to fire and then recharge. When the QT interval is longer than normal, it increases the risk for “torsade de pointes,” a life-threatening form of ventricular tachycardia.

<p>3.What are the types of bradyarrhythmias?

A bradyarrhythmia is a slow heart rhythm that is usually caused by disease in the heart’s conduction system. Types of bradyarrhythmias include:</p>

Sinus node dysfunction: Slow heart rhythms due to an abnormal SA node.

Heart block: A delay or complete block of the electrical impulse as it travels from the sinus node to the ventricles. The level of the block or delay may occur in the AV node or HIS-Purkinje system. The heartbeat may be irregular and slow.

Premature heartbeats:
Premature heartbeats are extra beats that occur one at a time, sometimes in patterns that alternate with the normal heart beat. The extra beats may come from the top chamber of the heart (premature atrial contractions) or the bottom chamber (premature ventricular contractions).
A premature heartbeat may feel like your heart skipped a beat. These extra beats are generally not concerning, and they seldom mean you have a more serious condition. Still, a premature beat can trigger a longer-lasting arrhythmia, especially in people with heart disease. Occasionally, very frequent premature beats that last for several years may lead to a weak heart.
Premature heartbeats may occur when resting. Sometimes premature heartbeats are caused by stress, strenuous exercise or stimulants, such as caffeine or nicotine.

</body>
</html>

predict_base.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>Predict</title>
         <link  href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet">
                                                                      <script
src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
    <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
                                                                      <script
src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
          <link  href="{{  url_for('static',  filename='css/flask_main_style.css')  }}"
rel="stylesheet">
<style>
.bar
{
margin: 0px;
padding:20px;
background-color:white;
opacity:0.6;
color:black;
font-family:'Roboto',sans-serif;
font-style: italic;
border-radius:20px;
font-size:25px;
}
a
{
color:grey;
float:right;
text-decoration:none;
font-style:normal;
padding-right:20px;
}
a:hover{
background-color:black;
```

```
color:white;
border-radius:15px;0
font-size:30px;
padding-left:10px;
}

</style>
</head>

<body>
      <div class="bar">
<a href="/upload" >Predict</a>
<a href="/info">Info</a>
<a href="/types">Types</a>
<a href="/about">Home</a>
<br>
</div>
  <body style="background-color:AliceBlue">
<div id="maintext" style="position:absolute; margin:auto; width:700px; height:200px; text-align:center; top:0; bottom: 200; left: 0; right: 0;">
  <h1 class="text mb-2" > Please upload your image to be classified! </h1>
    <h4 class="text-brown mb-5" >(Please upload images less than 600kb in size)</h4>
 <div class="wrapingimage">
<img src="f2.jpg" class="right"
       width="250"
     height="250" align="center">
     <br>
     <div style="border:black; border-width:1px; border-style:solid;">
     <p align="center"> Learn the stage of your heart with a single click!</p>
    </div>
    <br>
    <br>
```

```
    <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/flask_main_js.js') }}" type="text/javascript"></script>
</footer>
</html>
```

GitHub & Project Demo Link

Github repository link : https://github.com/IBM-EPBL/IBM-Project-11172-1659274952

Demo video link : https://drive.google.com/file/d/1c7AgSOShQlE-sifmA0zZkkQo2mRdFGIQ/view?usp=sharing