

# CLASSIFICATION OF ARRHYTHMIA BY USING DEEP LEARNING WITH 2-D ECG SPECTRAL IMAGE REPRESENTATION

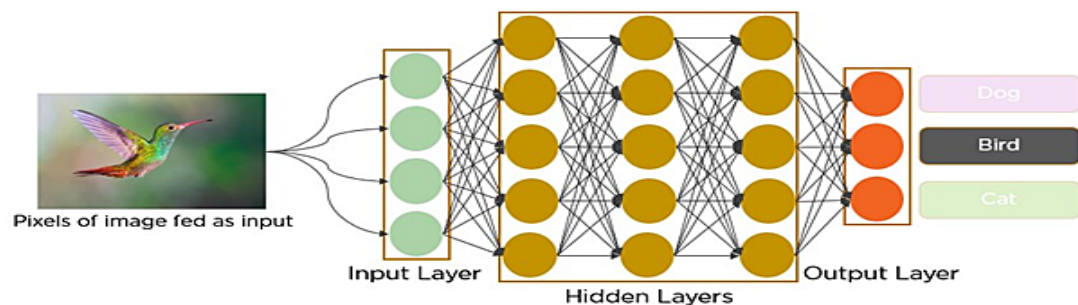
## INTRODUCTION:

### 1.1 OVERVIEW:

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

### 1.2 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 Convolutional Neural Networks.

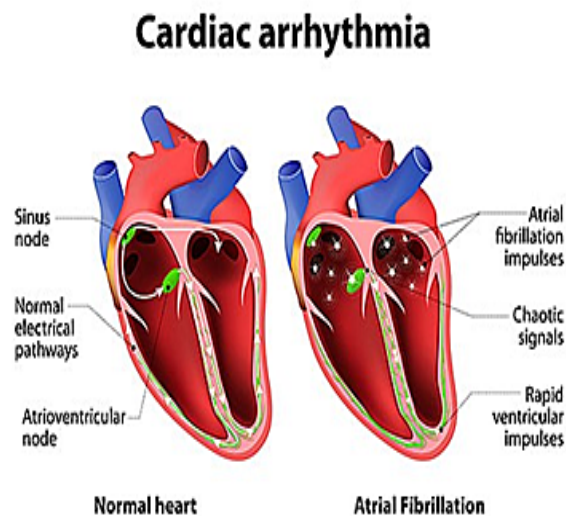


In deep learning, a convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. 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.

## 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.



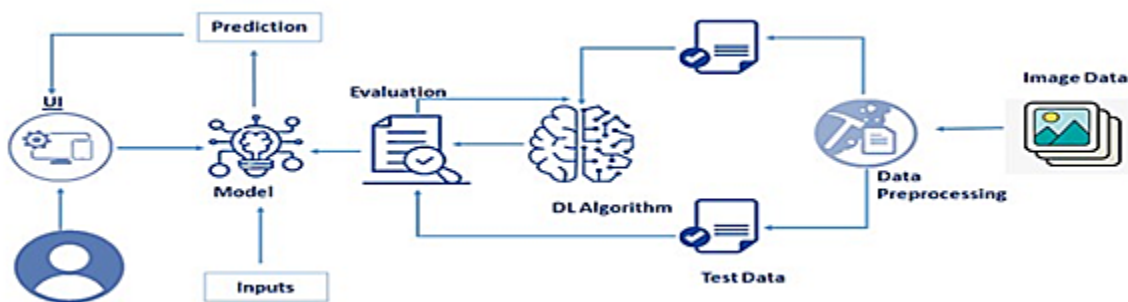
### 2.2 PROPOSED SOLUTION:

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.

## **THEORETICAL EXPERIENCE:**

### **3.1 BLOCK DIAGRAM:**



We will prepare the project by following the below steps:

1. We will be working with Sequential type of modeling.
2. We will be working with Keras capabilities.
3. We will be working with image processing techniques.
4. We will build a web application using the Flask framework.
5. Afterwards we will be training our dataset in the IBM cloud and building another model from IBM and we will also test it.

### **3.2 HARDWARE & SOFTWARE DESIGNING:**

#### **3.2.1 HARDWARE COMPONENTS USED:**

Since we are using the IBM cloud as a platform to execute this project we don't need any hardware components other than our system.

#### **3.2.2 SOFTWARE COMPONENTS USED:**

We will be using Anaconda Navigator which is installed in our system and Watson studio from the IBM cloud to complete the project.

## **ANACONDA NAVIGATOR:-**

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

## **WATSON STUDIO:**

Watson Studio is one of the core services in Cloud Pak for Data as a Service. Watson Studio provides you with the environment and tools to solve your business problems by collaboratively working with data. You can choose the tools you need to analyze and visualize data, to cleanse and shape data, or to build machine learning models.

This illustration shows how the architecture of Watson Studio is centered around the project. A project is a workspace where you organize your resources and work with data. Watson Studio projects fully integrate with the catalogs and deployment spaces:

- Deployment spaces are provided by the Watson Machine Learning service.
- You can easily move assets between projects and deployment spaces.

## **EXPERIMENTAL INVESTIGATIONS:-**

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:

1. **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.
2. **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.

3. **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.

4. **DROPOUT LAYER:** There is usually a dropout layer before the fullyconnected 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.

## FLOW CHART & RESULTS WITH SCREENSHOTS:

### 5.1 FLOW CHART & RESULTS BY TRAINING MODEL IN LOCAL MACHINE:-

#### A. DATASET COLLECTION:

The dataset contains six classes:

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

#### B. 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.

```
In [3]: 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
        from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

- **Configure ImageDataGenerator Class:**

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

1. Image shifts via the width\_shift\_range and height\_shift\_range arguments.
2. Image flips via the horizontal\_flip and vertical\_flip arguments.
3. Image rotates via the rotation\_range argument
4. Image brightness via the brightness\_range argument.
5. Image zooms via the zoom\_range argument.

```
In [4]: train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True)
```

```
In [5]: test_datagen = ImageDataGenerator(rescale = 1./255)
```

- **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}

```
In [6]: x_train = train_datagen.flow_from_directory("/content/data/train", target_size = (64,64), batch_size = 32, class_mode = "categorical")
```

Found 15341 images belonging to 6 classes.

```
In [7]: x_test = test_datagen.flow_from_directory("/content/data/test", target_size = (64,64), batch_size = 32, class_mode = "categorical")
```

Found 6825 images belonging to 6 classes.

```
In [8]: x_train.class_indices
```

```
Out[8]: {'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.

## C. 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:**

```
In [9]: from tensorflow.keras.models import Sequential
        from tensorflow.keras.layers import Dense, Convolution2D, MaxPooling2D, Flatten
```

- **Initializing the model:**

Keras has 2 ways to define a neural network:

1. Sequential
2. 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.

Initialize the model

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

### Adding CNN layers

```
In [11]: model.add(Convolution2D(32,(3,3),activation="relu",strides=(1,1),input_shape=(64,64,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
conv2d (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
flatten (Flatten)	(None, 30752)	0
=====		
Total params: 896		
Trainable params: 896		
Non-trainable params: 0		

- **Adding Hidden Layers:**

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

```
In [12]: model.add(Dense(500,activation="relu"))
model.add(Dense(500,activation="relu"))
```

- **Adding Output Layer:**

```
In [13]: model.add(Dense(6,activation="softmax"))
```

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.

```
In [14]: model.compile(loss="categorical_crossentropy",optimizer="adam",metrics=['accuracy'])
         len(x_train)
```

- **Training the model:**

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

```
In [15]: model.fit(x_train,epochs=5,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_test))

Epoch 1/5
480/480 [=====] - 140s 289ms/step - loss: 1.0137 - accuracy: 0.6721 - val_loss: 0.6701 - val_accuracy: 0.7538
Epoch 2/5
480/480 [=====] - 142s 296ms/step - loss: 0.3082 - accuracy: 0.9022 - val_loss: 0.5529 - val_accuracy: 0.8101
Epoch 3/5
480/480 [=====] - 140s 291ms/step - loss: 0.1732 - accuracy: 0.9475 - val_loss: 0.4951 - val_accuracy: 0.8379
Epoch 4/5
480/480 [=====] - 137s 286ms/step - loss: 0.1226 - accuracy: 0.9627 - val_loss: 0.5920 - val_accuracy: 0.8503
Epoch 5/5
480/480 [=====] - 139s 290ms/step - loss: 0.0892 - accuracy: 0.9727 - val_loss: 0.4156 - val_accuracy: 0.8752
```

- **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.

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

- **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.

```
In [17]: import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model=load_model('ECG.h5')
img=image.load_img("/content/data/test/Right Bundle Branch Block/fig_101.png",target_size=(64,64))
img
```

Out[17]: 

```
In [18]: x=image.img_to_array(img)
x
```

```
In [22]: pred=model.predict(x)

1/1 [=====] - 0s 79ms/step
```

```
In [23]: pred
```

```
Out[23]: array([[0., 0., 0., 0., 1., 0.]], dtype=float32)
```

```
In [24]: index=['Left Bundle Branch Block',
'Normal',
'Premature Atrial Contraction',
'Premature Ventricular Contractions',
'Right Bundle Branch Block',
'Ventricular Fibrillation']
index[np.argmax(pred)]
```

```
Out[24]: 'Right Bundle Branch Block'
```

## D. 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

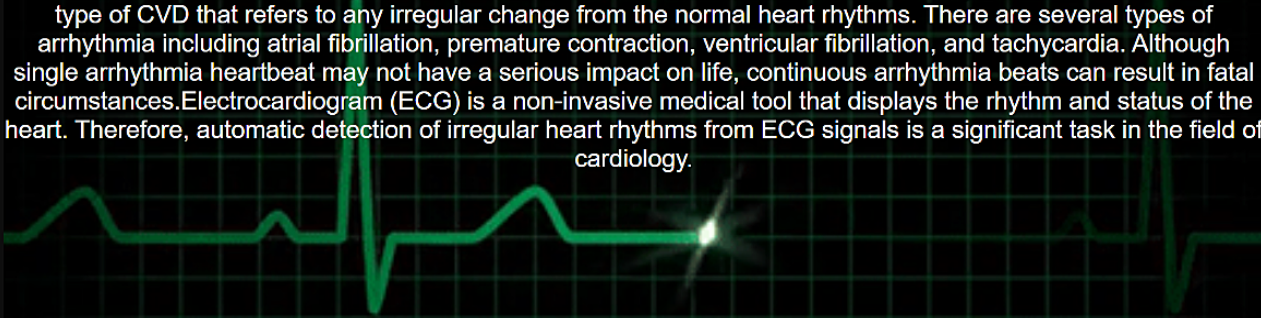
- **Building HTML Pages:**
  - We use HTML to create the front end part of the web page.
  - Here, we created 4 html pages- home.html, predict\_base.html, predict.html, information.

- [home.html](#) displays the home page.

Home Info Predict

## ECG arrhythmia classification using CNN

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




- [information.html](#) displays all important details to be known about ECG.

Home Info Predict

## ECG

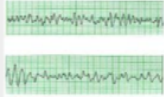
**Normal**

If the test is normal, it should show that your heart is beating at an even rate of 60 to 100 beats per minute. Many different heart conditions can show up on an ECG, including a fast, slow, or abnormal heart rhythm, a heart defect, coronary artery disease, heart valve disease, or an enlarged heart.



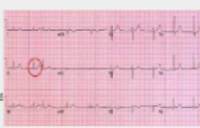
**Ventricular Fibrillation:**

Ventricular fibrillation (VF) is due to multiple wavelet reentrant electrical activity and is manifested on electrocardiogram (ECG) by altered baseline undulations that are irregular in timing and morphology. VF is the presenting rhythm for about 70% of patients in cardiac arrest.




**Premature atrial contractions:**

Premature atrial contractions (PACs) are extra heartbeats that begin in one of your heart's two upper chambers (atria). Most people will experience PACs at some point in their lives, and they are often a result of stress, caffeine, or alcohol. They are often more noticeable while at rest. Premature atrial contractions occasionally may be caused by heart disease but usually happen spontaneously and without apparent cause.




**Premature ventricular contractions:**

Ventricular contractions (PVCs) are extra heartbeats that begin in one of the heart's two lower pumping chambers (ventricles). These extra beats disrupt the regular heart rhythm, sometimes causing a sensation of a fluttering or a skipped beat in the chest. In the vast majority of cases, PVCs have no known cause and occur spontaneously. Common known etiologies include excess caffeine consumption, excess catecholamines, [4] high levels of anxiety, and electrolyte abnormalities.




**Right bundle branch block:**

Right bundle branch block (RBBB) is an electrocardiogram finding resulting in a widened QRS and electrocardiographic vector changes. Although usually benign, this finding can represent underlying myocardial disease and is a predictor of mortality in certain patient populations.



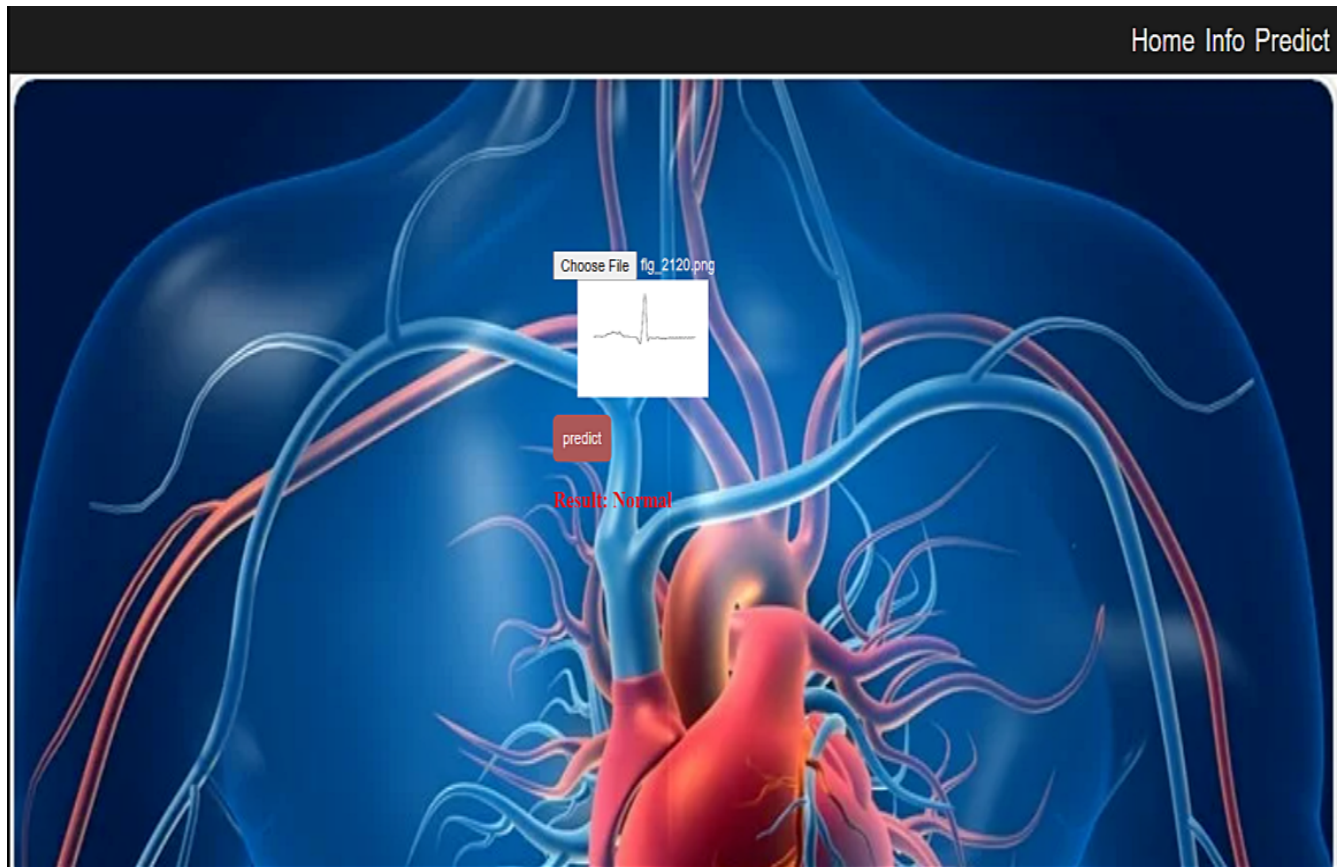
**Left bundle branch block:**

Left bundle branch block (LBBB) occurs when something blocks or disrupts the electrical impulse that causes your heart to beat. This block leads to an abnormal heart rhythm. A diagnosis of left bundle branch block often means that you have an underlying heart condition.



© All Rights Reserved

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



- **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.

```

Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\lap\Desktop\Flask>python -m flask run
2022-11-15 16:50:44.651796: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlderror: cudart64_110.dll not found
2022-11-15 16:50:44.652398: I tensorflow/stream_executor/cuda/cudart_stub.cc:79] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2022-11-15 16:51:20.441380: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'nvcuda.dll'; dlderror: nvcuda.dll not found
2022-11-15 16:51:20.441897: W tensorflow/stream_executor/cuda/cuda_driver.cc:283] failed call to cuInit: UNKNOWN ERROR (303)
2022-11-15 16:51:20.610442: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-4RDSYMH6
2022-11-15 16:51:20.611759: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-4RDSYMH6
2022-11-15 16:51:20.664627: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [15/Nov/2022 16:51:48] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [15/Nov/2022 16:52:14] "GET /info HTTP/1.1" 200 -
127.0.0.1 - - [15/Nov/2022 16:52:22] "GET /upload HTTP/1.1" 200 -
127.0.0.1 - - [15/Nov/2022 16:52:22] "GET /upload HTTP/1.1" 200 -
127.0.0.1 - - [15/Nov/2022 16:52:22] "GET /static/js/main.js HTTP/1.1" 304 -
file save
1/1 [-----] - 4s 4s/step
1/1 [-----] - 4s 4s/step
prediction prediction [0]
127.0.0.1 - - [15/Nov/2022 16:53:15] "POST /predict HTTP/1.1" 200 -
[0]
127.0.0.1 - - [15/Nov/2022 16:53:15] "POST /predict HTTP/1.1" 200 -
file save
1/1 [-----] - 0s 59ms/step
prediction [1]
127.0.0.1 - - [15/Nov/2022 16:53:43] "POST /predict HTTP/1.1" 200 -

```

- **Running The App:**

```
C:\Users\lap\Desktop\Flask>python -m flask run
```

```

To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit

```

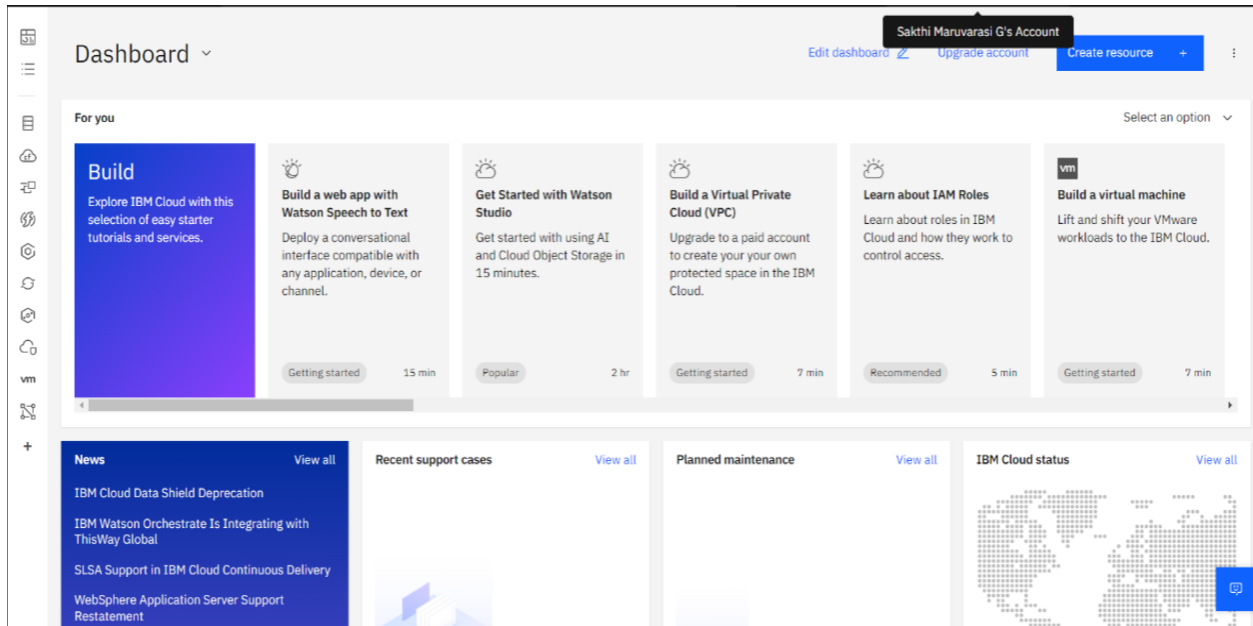
Navigate to the localhost (<http://127.0.0.1:5000/>) where you can view your web page.

## 5.2 Flow Chart & Results by training model in IBM WATSON STUDIO:

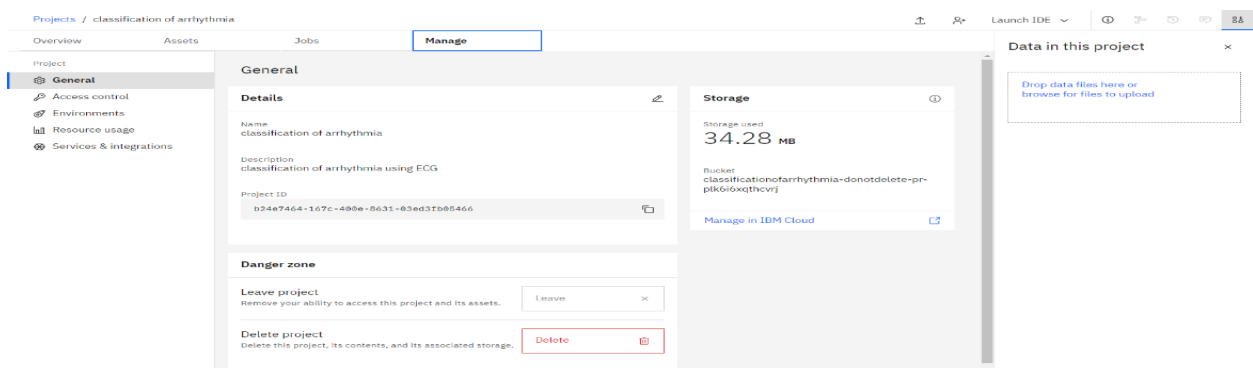
### A. Creating IBM cloud account:-

We have to create an IBM Cloud Account and should log in.

### B. Register for IBM Cloud account:-



### C. Deployment space in the watson studio:-



#### D. Apply CNN algorithm and save the model and deploy it using API key generated:-

```
In [120... model.save('ECG.h5')
```

```
In [121... !tar -zcvf ECG-model_new.tgz ECG.h5
```

ECG.h5

```
In [134... ls -l
```

```
data/  
ECG.h5  
ECG-model_new.tgz  
my_model1.tar1.gz
```

```
In [95]: from ibm_watson_machine_learning import APIClient  
wml_credentials = {  
    "url": "https://us-south.ml.cloud.ibm.com",  
    "apikey": "m6NbiyMRA29yQcTssg5Wi0HkkZHyK-gib4hSGE3ppyVy"  
}  
client = APIClient(wml_credentials)
```

```
In [124... client = APIClient(wml_credentials)
```

```
In [125... client.spaces.list()
```

Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50

ID	NAME	CREATED
d68afc27-adb2-4919-a9d0-cd6acb8700f3	Model building	2022-11-11T06:11:54.400Z

```
In [126... 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'])
```

```
In [127... space_uid = guid_from_space_name(client, 'Model building')  
print("Space UID = " + space_uid)
```

Space UID = d68afc27-adb2-4919-a9d0-cd6acb8700f3

```
In [128... client.set_default_space(space_uid)
```

```
Out[128... 'SUCCESS'
```

```

In [130.. software_spec_uid = client.software_specifications.get_uid_by_name("tensorflow_rt22.1-py3.9")
software_spec_uid

Out[130.. 'acd9c798-6974-5d2f-a657-ce06e986df4d'

In [131.. model_details = client.repository.store_model(model='ECG-model_new.tgz',meta_props={
client.repository.ModelMetaNames.NAME:"ECG_Model",
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)

This method is deprecated, please use get_model_id()
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/ibm_watson_machine_learning/repository.py:1453: UserWarning: This method is deprecated, please
use get_model_id()
warn("This method is deprecated, please use get_model_id()")

In [132.. model_id

Out[132.. '7bc41052-6999-4af0-ba61-aebca5eaa72e'

In [133.. client.repository.download(model_id,'my_model1.tar1.gz')

```

**E. For downloading the model we have to run the last part of the above code in the local jupyter notebook:**

```

In [133.. client.repository.download(model_id,'my_model1.tar1.gz')

```

Hence we trained the model using IBM Watson.

## ADVANTAGES & DISADVANTAGES:

### 6.1 ADVANTAGES:

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

### 6.2 DISADVANTAGES:

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



## **APPLICATIONS :**

- It is useful for identifying the arrhythmia disease at an early stage.
- It is useful in detecting cardiovascular disorders

## **.CONCLUSION:**

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

## **FUTURE SCOPE:**

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

## **REFERENCES:**

- <https://github.com/smartinternz02/SI-GuidedProject-78335-1656736746>
- <https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/>

**THE END**

**TEAM MEMBERS:**

**SAKTHI MARUVARASI. G**

**NIVETHITHA. M**

**NISHANTH. P**

**PRASANNA BALAJI. R**

**NITHISHKUMAR. P**

**EMAIL:** sakthig1701@gmail.com

