

Project Development Phase

Delivery of Sprint 1

Date	31 October 2022
Team ID	PNT2022TMID36404
Project Name	Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation

Task 1:

Download the dataset:

The dataset has been downloaded and the drive link is given

https://drive.google.com/drive/folders/1h_v0ja8sMe4FbeYO85fGH7Zgsa2UTOHG?usp=share_link

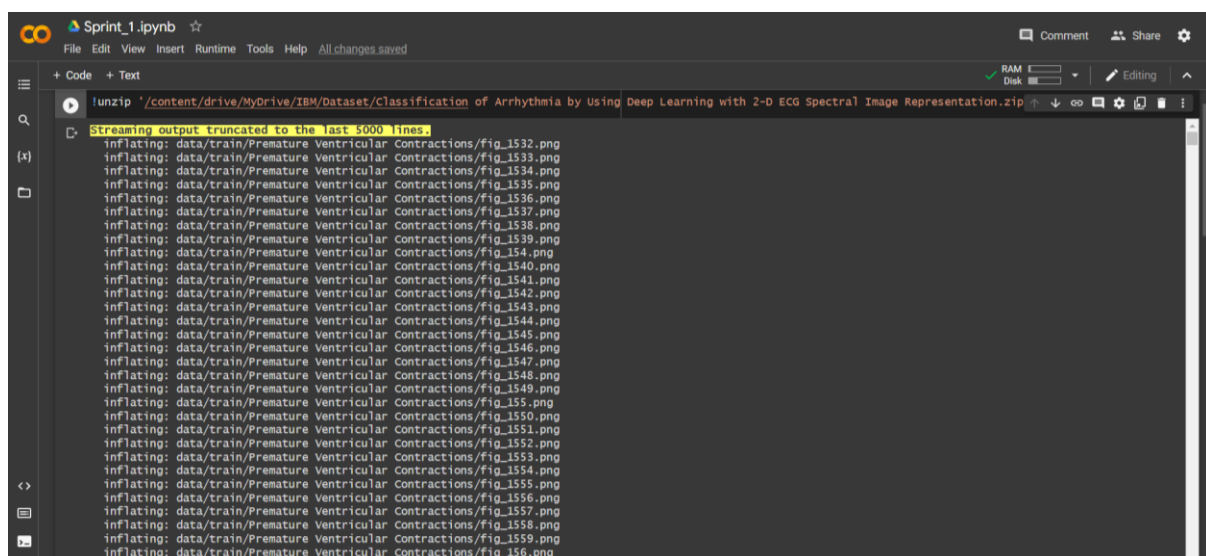
Run the dataset (Unzipping the dataset):

Code:

```
#UNZIPPING THE DATASET

!unzip '/content/drive/MyDrive/IBM/Dataset/Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation.zip'
```

Output:



```
Sprint_1.ipynb
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

!unzip '/content/drive/MyDrive/IBM/Dataset/Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation.zip'

Streaming output truncated to the last 3000 lines.
inflating: data/train/Premature Ventricular Contractions/fig_1532.png
inflating: data/train/Premature Ventricular Contractions/fig_1533.png
inflating: data/train/Premature Ventricular Contractions/fig_1534.png
inflating: data/train/Premature Ventricular Contractions/fig_1535.png
inflating: data/train/Premature Ventricular Contractions/fig_1536.png
inflating: data/train/Premature Ventricular Contractions/fig_1537.png
inflating: data/train/Premature Ventricular Contractions/fig_1538.png
inflating: data/train/Premature Ventricular Contractions/fig_1539.png
inflating: data/train/Premature Ventricular Contractions/fig_154.png
inflating: data/train/Premature Ventricular Contractions/fig_1540.png
inflating: data/train/Premature Ventricular Contractions/fig_1541.png
inflating: data/train/Premature Ventricular Contractions/fig_1542.png
inflating: data/train/Premature Ventricular Contractions/fig_1543.png
inflating: data/train/Premature Ventricular Contractions/fig_1544.png
inflating: data/train/Premature Ventricular Contractions/fig_1545.png
inflating: data/train/Premature Ventricular Contractions/fig_1546.png
inflating: data/train/Premature Ventricular Contractions/fig_1547.png
inflating: data/train/Premature Ventricular Contractions/fig_1548.png
inflating: data/train/Premature Ventricular Contractions/fig_1549.png
inflating: data/train/Premature Ventricular Contractions/fig_155.png
inflating: data/train/Premature Ventricular Contractions/fig_1550.png
inflating: data/train/Premature Ventricular Contractions/fig_1551.png
inflating: data/train/Premature Ventricular Contractions/fig_1552.png
inflating: data/train/Premature Ventricular Contractions/fig_1553.png
inflating: data/train/Premature Ventricular Contractions/fig_1554.png
inflating: data/train/Premature Ventricular Contractions/fig_1555.png
inflating: data/train/Premature Ventricular Contractions/fig_1556.png
inflating: data/train/Premature Ventricular Contractions/fig_1557.png
inflating: data/train/Premature Ventricular Contractions/fig_1558.png
inflating: data/train/Premature Ventricular Contractions/fig_1559.png
inflating: data/train/Premature Ventricular Contractions/fig_156.png
```

Task 2:

Image Preprocessing:

Import ImageDataGenerator Library:

Code:

```
#IMPORTING THE IMAGEDATAGENERATOR LIBRARY

from keras.preprocessing.image import ImageDataGenerator
```

Configure ImageDataGenerator class:

Code:

```
#SETTING PARAMETER FOR IMAGE DATA AUGMENTATION TO THE TRAINING DATA

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

#IMAGE DATA AUGMENTATION TO THE TESTING DATA

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

Apply ImageDataGenerator Functionality to Trainset and Testset:

Code:

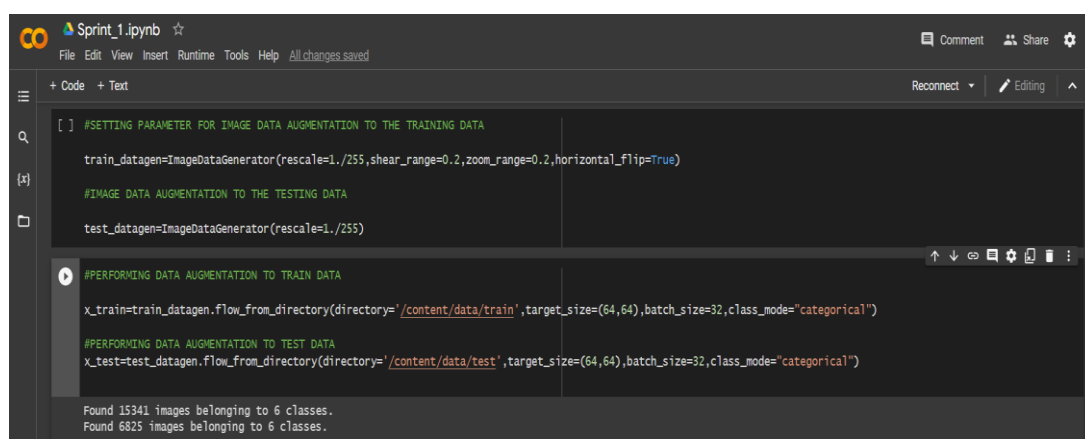
```
#PERFORMING DATA AUGMENTATION TO TRAIN DATA

x_train=train_datagen.flow_from_directory(directory='/content/data/train',target_size=(64,64),batch_size=32,class_mode="categorical")

#PERFORMING DATA AUGMENTATION TO TEST DATA

x_test=test_datagen.flow_from_directory(directory='/content/data/test',target_size=(64,64),batch_size=32,class_mode="categorical")
```

Output:

A screenshot of a Jupyter Notebook interface. The top bar shows the file name 'Sprint_1.ipynb' and various icons for file operations, runtime, and help. The left sidebar contains icons for file explorer, search, and other notebook functions. The main area displays two code cells. The first cell contains code for initializing ImageDataGenerator for training and testing. The second cell contains code for loading data from directories. The output of the second cell is visible at the bottom, showing the number of images found for each class.

```
[ ] #SETTING PARAMETER FOR IMAGE DATA AUGMENTATION TO THE TRAINING DATA
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)

#IMAGE DATA AUGMENTATION TO THE TESTING DATA
test_datagen=ImageDataGenerator(rescale=1./255)

#PERFORMING DATA AUGMENTATION TO TRAIN DATA
x_train=train_datagen.flow_from_directory(directory='/content/data/train',target_size=(64,64),batch_size=32,class_mode="categorical")

#PERFORMING DATA AUGMENTATION TO TEST DATA
x_test=test_datagen.flow_from_directory(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.

Task 3:

Model Building:

Import Libraries:

Code:

```
#IMPORTING LIBRARIES

import numpy as np #used for numerical analysis

import tensorflow #open source used for both ML and DL for computation

from tensorflow.keras.models import Sequential #it is a plain stack of
layers

from tensorflow.keras import layers #A layer consists of Tensor-
in Tensor-out computation function

#DENSE LAYER IS THE REGULAR DEEPLY CONNECTED NURAL NETWORK LAYER

from tensorflow.keras.layers import Dense,Flatten

# FLATTEN-USED FOR FLATTENING THE INPUT OR CHANGE THE DIRECTION

from tensorflow.keras.layers import Conv2D,MaxPooling2D #convolution La
yer
```

Initialize Model:

Code:

```
#INITIALIZING MODEL

model=Sequential()
```