

Project Development Phase

Delivery of Sprint 2

Date	05 November 2022
Team ID	PNT2022TMID33910
Project Name	Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation

Task 1:

Model Building:

Adding CNN Layers:

Code:

```
#ADDING CNN LAYERS

model.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu')) #convolution layer
model.add(MaxPooling2D(pool_size=(2, 2))) #MaxPooling2D for downsampling the input

model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten()) #flatten the dimension of the image
```

Adding Dense Layers:

Code:

```
#ADDING DENSE LAYERS

model.add(Dense(32)) #deeply connected neural network layers.
model.add(Dense(6, activation='softmax'))
```

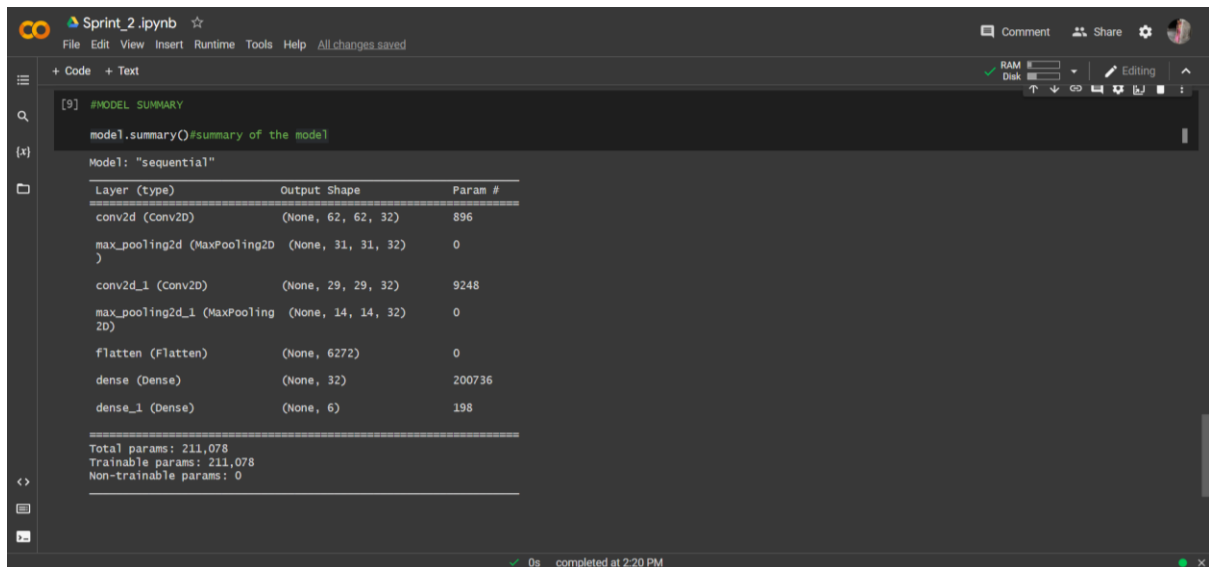
Model Summary:

Code:

```
#MODEL SUMMARY

model.summary() #summary of the model
```

Output:



A screenshot of a Jupyter Notebook interface titled "Sprint_2.ipynb". The code cell contains the command `model.summary()` with a comment `#summary of the model`. The output displays the model's architecture in a table format.

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 (MaxPooling2D)	(None, 14, 14, 32)	0
flatten (Flatten)	(None, 6272)	0
dense (Dense)	(None, 32)	200736
dense_1 (Dense)	(None, 6)	198

Below the table, the following statistics are listed:

- Total params: 211,078
- Trainable params: 211,078
- Non-trainable params: 0

Configure the Learning Process:

Code:

```
#CONFIGURE THE LEARNING PROCESS

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

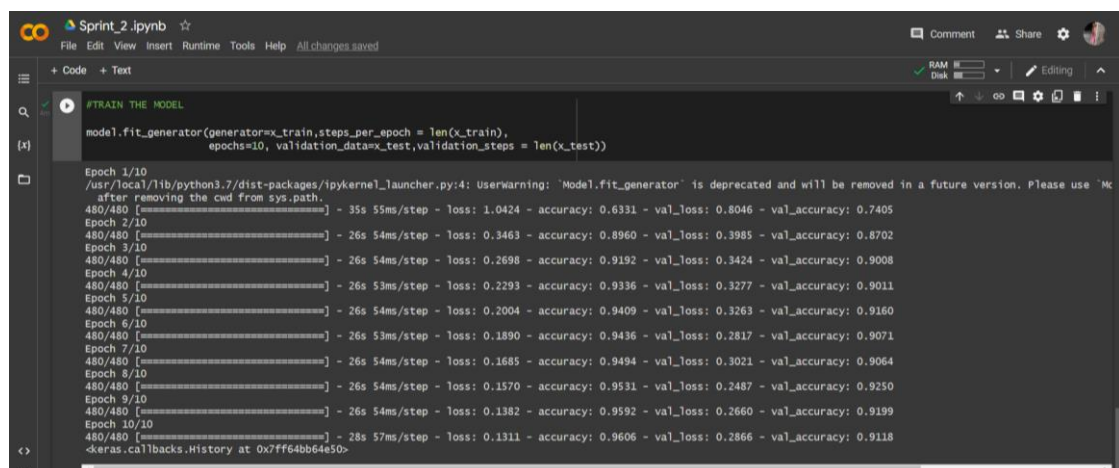
Train the Model:

Code:

```
#TRAIN THE MODEL

model.fit_generator(generator=x_train, steps_per_epoch = len(x_train),
                    epochs=10, validation_data=x_test, validation_steps = len(x_test))
```

Output:



A screenshot of a Jupyter Notebook interface titled "Sprint_2.ipynb". The code cell contains the command `model.fit_generator(generator=x_train, steps_per_epoch = len(x_train), epochs=10, validation_data=x_test, validation_steps = len(x_test))` with a comment `#TRAIN THE MODEL`. The output shows the training progress for 10 epochs.

```
Epoch 1/10
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: UserWarning: 'Model.fit_generator' is deprecated and will be removed in a future version. Please use 'Model.fit' instead.
after removing the cwd from sys.path.
480/480 [=====] - 35s 55ms/step - loss: 1.0424 - accuracy: 0.6331 - val_loss: 0.8046 - val_accuracy: 0.7405
Epoch 2/10
480/480 [=====] - 26s 54ms/step - loss: 0.3463 - accuracy: 0.8960 - val_loss: 0.3985 - val_accuracy: 0.8702
Epoch 3/10
480/480 [=====] - 26s 54ms/step - loss: 0.2698 - accuracy: 0.9192 - val_loss: 0.3424 - val_accuracy: 0.9008
Epoch 4/10
480/480 [=====] - 26s 53ms/step - loss: 0.2293 - accuracy: 0.9336 - val_loss: 0.3277 - val_accuracy: 0.9011
Epoch 5/10
480/480 [=====] - 26s 54ms/step - loss: 0.2004 - accuracy: 0.9409 - val_loss: 0.3263 - val_accuracy: 0.9160
Epoch 6/10
480/480 [=====] - 26s 53ms/step - loss: 0.1890 - accuracy: 0.9436 - val_loss: 0.2817 - val_accuracy: 0.9071
Epoch 7/10
480/480 [=====] - 26s 54ms/step - loss: 0.1685 - accuracy: 0.9494 - val_loss: 0.3021 - val_accuracy: 0.9064
Epoch 8/10
480/480 [=====] - 26s 54ms/step - loss: 0.1570 - accuracy: 0.9531 - val_loss: 0.2487 - val_accuracy: 0.9250
Epoch 9/10
480/480 [=====] - 26s 54ms/step - loss: 0.1382 - accuracy: 0.9592 - val_loss: 0.2660 - val_accuracy: 0.9199
Epoch 10/10
480/480 [=====] - 28s 57ms/step - loss: 0.1311 - accuracy: 0.9606 - val_loss: 0.2866 - val_accuracy: 0.9118
<keras.callbacks.History at 0x7ff64bb64e50>
```

Save the Model:

Code:

```
#SAVE THE MODEL

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

Test the Model:

Code:

```
#TEST THE MODEL

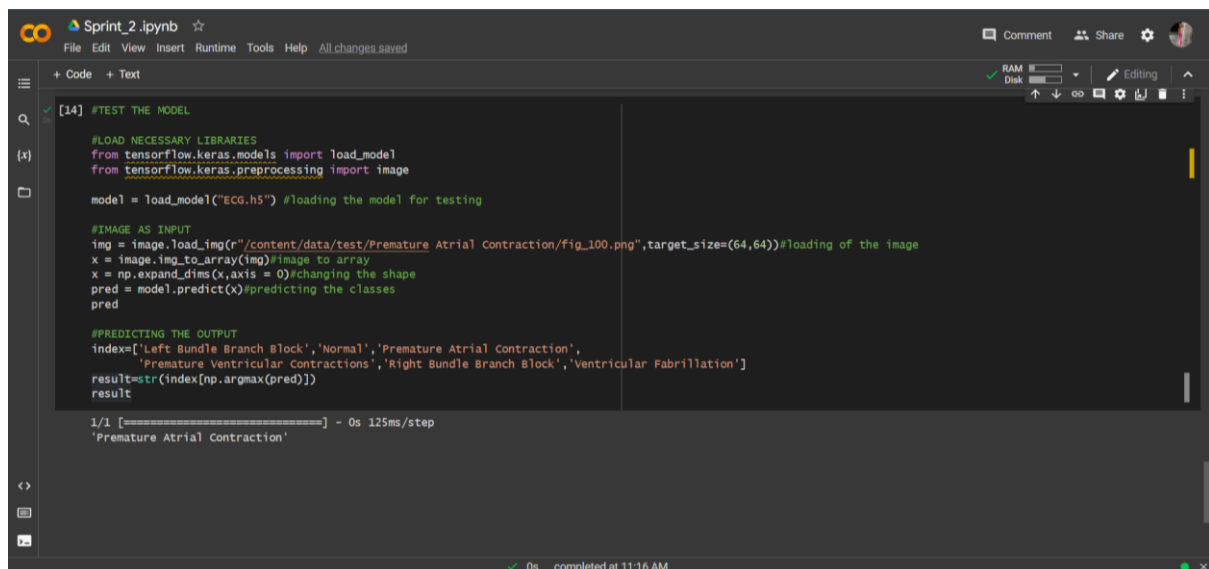
#LOAD NECESSARY LIBRARIES
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image

model = load_model("ECG.h5") #loading the model for testing

#IMAGE AS INPUT
img = image.load_img(r"/content/data/test/Premature Atrial Contraction/fig_100.png",target_size=(64,64))#loading of the image
x = image.img_to_array(img)#image to array
x = np.expand_dims(x,axis = 0)#changing the shape
pred = model.predict(x)#predicting the classes
pred

#PREDICTING THE OUTPUT
index=['Left Bundle Branch Block','Normal','Premature Atrial Contraction',
      'Premature Ventricular Contractions','Right Bundle Branch Block',
      'Ventricular Fibrillation']
result=str(index[np.argmax(pred)])
result
```

Output:



```
Sprint_2.ipynb
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

[14] #TEST THE MODEL

#LOAD NECESSARY LIBRARIES
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image

model = load_model("ECG.h5") #loading the model for testing

#IMAGE AS INPUT
img = image.load_img(r"/content/data/test/Premature Atrial Contraction/fig_100.png",target_size=(64,64))#loading of the image
x = image.img_to_array(img)#image to array
x = np.expand_dims(x,axis = 0)#changing the shape
pred = model.predict(x)#predicting the classes
pred

#PREDICTING THE OUTPUT
index=['Left Bundle Branch Block','Normal','Premature Atrial Contraction',
      'Premature Ventricular Contractions','Right Bundle Branch Block',
      'Ventricular Fibrillation']
result=str(index[np.argmax(pred)])
result

1/1 [=====] - 0s 125ms/step
'Premature Atrial Contraction'
```