PROJECT DEVELOPMENT PHASE

SPRINT- II

Team ID	PNT2022TMID03593
Project Name	Project - Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation

MODEL BUILDING

Import the Library

In [7]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
```

Initializing the Model

In [8]:
```
model = Sequential()
```

Adding CNN Layers

In [9]:
```
model.add(Convolution2D(32, (3,3), input_shape = (64,64,3), activation = "relu"))
```

In [10]:
```
model.add(MaxPooling2D(pool_size = (2,2)))
```

In [11]:
```
model.add(Convolution2D(32, (3,3), activation = "relu"))
```

In [12]:
```
model.add(MaxPooling2D(pool_size = (2,2)))
```

In [13]:
```
model.add(Flatten())
```

Adding Dense Layers

In [14]:
```
model.add(Dense(units=128, kernel_initializer='random_uniform', activation="relu"))
```

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

In [16]:
```
model.add(Dense(units=128, kernel_initializer='random_uniform', activation="relu"))
```

In [17]:
```
model.add(Dense(units=128, kernel_initializer='random_uniform', activation="relu"))
```

In [18]:
```
model.add(Dense(units=128, kernel_initializer='random_uniform', activation="relu"))
```

In [19]:
```
model.add(Dense(units=6, kernel_initializer='random_uniform', activation="softmax"))
```

Configure the Learning Process

In [20]:
```
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 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
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
_________________________________________________________________
```

In [21]:
```
model.compile(optimizer = "adam", loss = "categorical_crossentropy", metrics = ["accuracy"])
```

Train the Model

In [22]:
```
model.fit(x_train, steps_per_epoch = len(x_train), epochs=9, validation_data=x_test,\
                    validation_steps = len(x_test))
```

```
Epoch 1/9
154/154 [==============================] - 149s 969ms/step - loss: 1.4548 - accuracy: 0.4762 - val_loss: 1.6820 - val_accuracy:
0.3193
Epoch 2/9
154/154 [==============================] - 134s 868ms/step - loss: 1.4318 - accuracy: 0.4788 - val_loss: 1.6579 - val_accuracy:
0.3193
Epoch 3/9
154/154 [==============================] - 137s 891ms/step - loss: 1.4272 - accuracy: 0.4788 - val_loss: 1.6783 - val_accuracy:
0.3193
Epoch 4/9
154/154 [==============================] - 144s 936ms/step - loss: 1.1741 - accuracy: 0.5423 - val_loss: 1.6035 - val_accuracy:
0.3736
Epoch 5/9
154/154 [==============================] - 511s 3s/step - loss: 0.9962 - accuracy: 0.6025 - val_loss: 1.4393 - val_accuracy: 0.
3871
Epoch 6/9
154/154 [==============================] - 456s 3s/step - loss: 0.8180 - accuracy: 0.6689 - val_loss: 1.2940 - val_accuracy: 0.
5320
Epoch 7/9
154/154 [==============================] - 169s 1s/step - loss: 0.5755 - accuracy: 0.8024 - val_loss: 1.1081 - val_accuracy: 0.
6913
Epoch 8/9
154/154 [==============================] - 157s 1s/step - loss: 0.4236 - accuracy: 0.8564 - val_loss: 0.7821 - val_accuracy: 0.
7596
Epoch 9/9
154/154 [==============================] - 183s 1s/step - loss: 0.3076 - accuracy: 0.8992 - val_loss: 0.7221 - val_accuracy: 0.
8180
```

Out[22]:
```
<tensorflow.python.keras.callbacks.History at 0x1e4d281a430>
```

Test the Model

In [24]:
```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```

In [27]:
```
model = load_model("ECG.h5")
img = image.load_img("C:/Users/Admin/Desktop/data/prediction/fig_2114-n.png", target_size = (64,64))
```

In [28]:
```
x = image.img_to_array(img)
```

In [29]:
```
import numpy as np
```

In [30]:
```
x = np.expand_dims(x,axis = 0)
```

In [31]:
```
pred = model.predict(x)
y_pred=np.argmax(pred)
y_pred
```

Out[31]: 1

In [32]:
```
index=['Left Bundle Branch Block',
       'Normal',
       'Premature Atrial Contraction',
       'Premature Ventricular Contractions',
       'Right Bundle Branch Block',
       'Ventricular Fibrillation']

result = str(index[y_pred])
result
```

Out[32]: 'Normal'