**Image Classification of Heartrate Images using CNN**

**Introduction:-**

The image classification of Heartrate images using CNN Model. To train the model to heartrate images(ECG) of different classes that are present in the dataset are Left Bundle Branch block, Normal , Premature Ventricular Contraction, Premature Atrial contraction, Right Bundle Branch Block, Ventricular Fibrillation. Build Convolution Neural Network and external bias is added, and Dense layers are added based on the accuracy of the model.

**Installation:-**

Install the required libraries for Image Classification Tensorflow, Keras and Basic libraries that are Numpy, pandas, OpenCV, Split-folder. To Install all these libraries, open the vscode new terminal , use this commends.

Python -m –upgrade pip.

pip install numpy.

pip install pandas.

pip install tensorflow.

pip install Keras.

pip install opencv.

pip install split-folder.

Pip install sk-learn.

**Creating Virtualenv:-**

First, we create Virtualenv on your project folder . we need to install Virtualenv. commands to install the Virtualenv open the new terminal follow the code below.

1.pip install Virtualenv.

2.python -m venv/file\_path/venv.

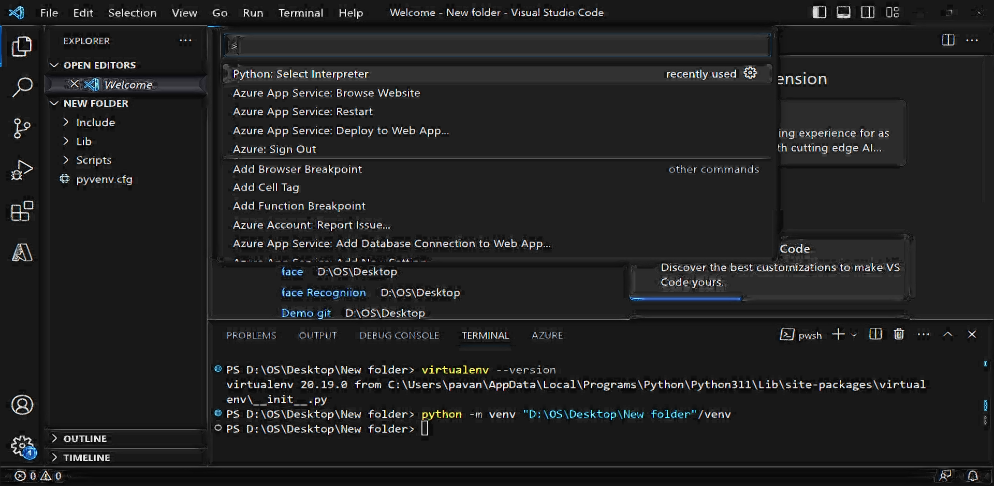
A screenshot of a computer

Description automatically generated

When you run the second command include, lib and scripts are the three folders are created in the project folder.

3.click (ctrl ­+ shift + P)

4.select Interpreter and click on the browse



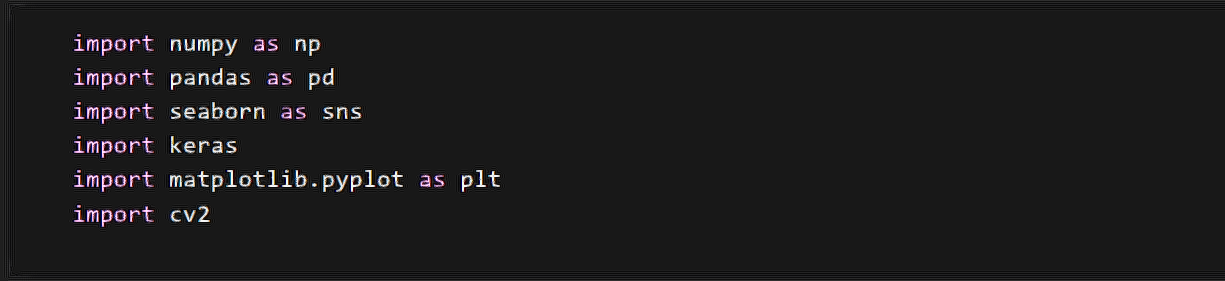
5.open the project folder and click on scripts and click on python.exe.

After creating the Virtualenv install all the required libraries in the project folder.

**Extensions:-**

Add the required extensions in vscode to run the codes and create the python file(.py) and Jupyter(.ipynb) and many more . click on the extensions left side and add the python ,pylance, jupyter, Azure, git, python indent and based on your requirement.

**Project:-**Importing the required libraries to image classification model.

****

Numpy works with the arrays, and it also works with the functions are algebra, Fourier Transformer and metrices. Numpy documentation click [here](https://numpy.org/doc/stable/user/quickstart.html).

Pandas are DataFrame. It is used to do data manipulation, data analysis, loading and saving the data in DataFrame format and statistical analysis and many things. Pandas’ documentation click [here](https://pandas.pydata.org/docs/user_guide/index.html).

Matplotlib is a visualization library. To visualize the graphs like line graphs and bar graphs and also used to show the images. Matplotlib documentation click [here](https://matplotlib.org/stable/tutorials/introductory/pyplot.html).

Tensorflow and Keras are the main libraries and all the Deep Learning models and different layers and Activation functions and Optimizers are present in those two libraries to see the documentation of [Tensorflow](https://www.tensorflow.org/api_docs) and [Keras](https://keras.io/api/).

A black screen with a white text

Description automatically generated

Importing the required layers and models. For image classification model we consider Sequential. It allows to specify the neural network. Sequential as the input to output. We build the required number of layers between the input and output.convoluton2D and MaxPooling and Flatten are add in the sequential.

After we consider the Sequential model Feature extraction is started the image should be resized as the matrix.

A diagram of layers

Description automatically generated

**A black rectangular object with a white border

Description automatically generated**

Importing the image from keras.preprocessing techniques. The Keras is the module they are so many preprocessing techniques from this technique importing of image is used to read the image into workspace.

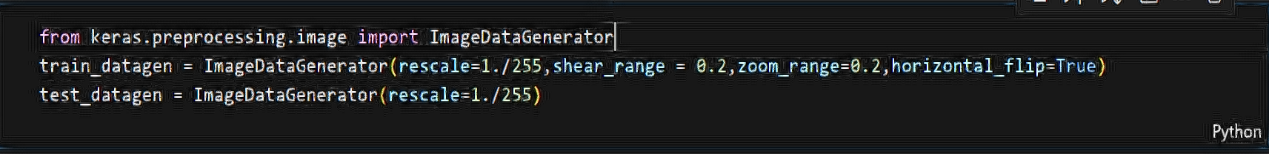
The load\_model is used to load the full convolution Neural Network when we saved after completed the model.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 7 | 6 | 3 | 9 |
| 1 | 3 | 0 | 1 | 8 |
| 0 | 4 | 2 | 9 | 1 |
| 9 | 2 | 0 | 6 | 6 |

The img\_to\_array is used to convert the image to array format.

**RGB Image -------------->**

**array**

****

We need to import the ImageDataGenerator it is used to change the original data to random data and rescale the image the image pixel should be 1 to 255 and divided by the pixels of 255 and the image data should be the 0 to 1 format. Apply the rescale to Train Dataset and Test Dataset.

A black screen with colorful text

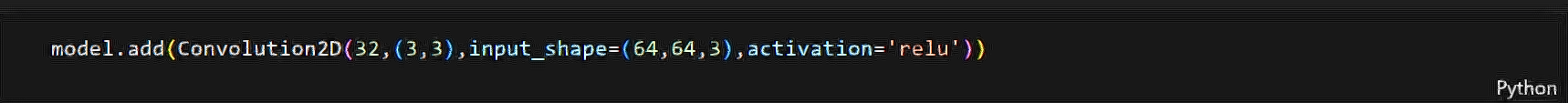
Description automatically generated

Load the train Dataset and Test Dataset with the train\_datagen and test\_datagen enter the file path that means location of your ECG image dataset. In my dataset 6 class are present in train and test dataset.

A black background with white text

Description automatically generated

After we imported all the libraries and imported the required models and layer and load the images dataset. Then we start the model. From the above code model in name we consider. The sequential is from input to output we can pass series of required neural layers in the sequential in between the input to output.



**Convolution 2D Layer:-**

The convolution neural network is to reduce the images into a form of easier to process, without losing any features that are important for getting a good prediction. For example, consider an image of height and width is 4\*4 and consider the kernel of 2\*2 then image is multiply with the kernel the image size should be 3\*3 the figure is shown in below.

|  |  |  |  |
| --- | --- | --- | --- |
| **5** | **3** | 1 | 0 |
| **6** | **7** | 8 | 1 |
| 0 | 4 | 2 | 9 |
| 9 | 2 | 0 | 8 |

|  |  |
| --- | --- |
| 0 | -1 |
| 1 | 2 |

|  |  |  |
| --- | --- | --- |
| 17 |  |  |
|  |  |  |
|  |  |  |

Convolution layer Kernel 3\*3

The kernel multiply with the first four boxes in the convolution layer and the calculation is 0\*5 + -1\*3 + 6\*1 + 2\*7 = 17.

|  |  |  |  |
| --- | --- | --- | --- |
| 5 | 3 | 1 | 0 |
| 6 | 7 | 8 | 1 |
| 0 | 4 | 2 | 9 |
| 9 | 2 | 0 | 8 |

|  |  |  |
| --- | --- | --- |
| 17 | 32 |  |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| 0 | -1 |
| 1 | 2 |

Convolution Layer Kernel 3\*3

The kernel multiply with the second four boxes in the convolution layer and the calculation is 0\*3 + -1\*1 + 1\*7 + 2\*8 = 32. Do this calculation to the full convolution layer then the image should reduce . considering the kernel size based on your requirement. The kernel values of 0,-1,1,2 are not the kernel values those are only for example. The real kernel values are taken by system.

Convolution2d(32,(3,3)) means the 32 is the features we extracting from image of convolution2d and (3,3) means the 3\*3 kernel is multiply with the convolution2d.lets take an example.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

3\*3 kernel

6\*6 convolution2d 4\*4 convolution

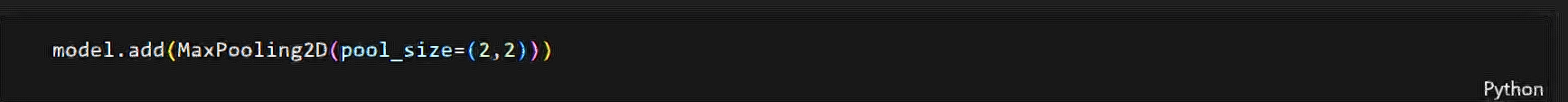
For example we consider the 6\*6 convolution2d than multiply with the 3\*3 kernel the multiplication process explain in the above, after that the image should be resized into the 4\*4 convolution.

The Input\_shape(64,64,3) means the input image of the 64 \*64 pixels and the 3 means the 3 channels of RGB.

The convolution2d is add into sequential. One layer is add and after that we add the pooling layer.

**Activation Function:-**

Activation function is the crucial part in the model is used to control how well the network model learns the training dataset. Activation function is used to whether the neural network activated or not . They are so many activation functions to choose the which activation function is based on the model. For the CNN model most used activation functions are sigmoid , Relu ,Tanh , Softmax.

****

**MaxPooling Layer:-**

The MaxPooling layer is used to take the output of convolution layer and consider the MaxPooling layer of 2\*2 matrix they are different pooling layers like AvgPooling and MaxPooling layer. By considering which Pooling layer check the which pooling layer gives the best accuracy.

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | 5 | 2 | 4 |
| 6 | 0 | **7** | 3 |
| 1 | 3 | 4 | 7 |
| 2 | 8 | 5 | 2 |

|  |  |  |
| --- | --- | --- |
| 9 |  |  |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| 9 | 5 |
| 6 | 0 |

Output from the convolution layer MaxPooling layer After Pooling

When we apply the MaxPooling layer to the output from the convolution layer when we consider the first four boxes then MaxPooling layer takes the value is 9 which is the max value in those four boxes.

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | 5 | 2 | 4 |
| 6 | 0 | **7** | 3 |
| 1 | 3 | 4 | 7 |
| 2 | 8 | 5 | 2 |

|  |  |  |
| --- | --- | --- |
| 9 | 7 |  |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| 5 | 2 |
| 0 | 7 |

Output from the convolution layer MaxPooling layer After Pooling

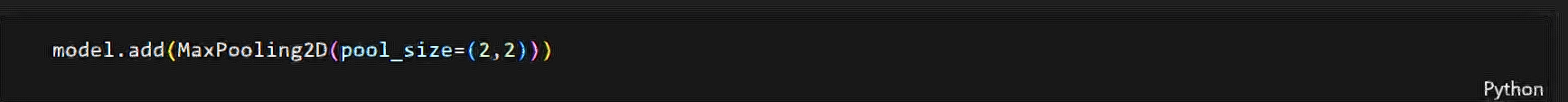
We consider the second four boxes. Then MaxPooling layer takes value is 7 which is the max value in those four boxes. Lets continue the process for all boxes present in the output from the convolution layer.

Pooling\_size(2,2) means size of the MaxPooling layer. After you add the pooling layer, the image has resized. This pooling layer added next to the convolution2d.

A black background with white text

Description automatically generated

Added the another convolution2D to extract the features from the output of MaxPooling layer after that added the another MaxPooling layer resize the image.

****

For adding all those layers to extract the features and reduce the size of the image. The four layers are added first layer is convolution2d and second layer MaxPooling layer and third layer convolution2d and fourth layer is MaxPooling layer, In this sequence first layer extract the key features after the resize the image in second layer and from the resized image the third layer extract the features after that fourth layer resize the image this is the process of CNN model and how many layers should added is based on the project your doing and based on the accuracy of the model we need to add the number of layers. After all this layers we add the flatten layer.

A black rectangular object with a black border

Description automatically generated

Flatten layer is used to convert the 2-dimensional array from the output of MaxPooling layer into a continuous linear vector. Let as consider an example

|  |
| --- |
| 0 |
| 1 |
| 2 |
| . |
| . |
| 8 |

Flatten layer

|  |  |  |
| --- | --- | --- |
| 0 | 1 | 2 |
| 3 | 4 | 5 |
| 6 | 7 | 8 |

Output from the MaxPooling layer

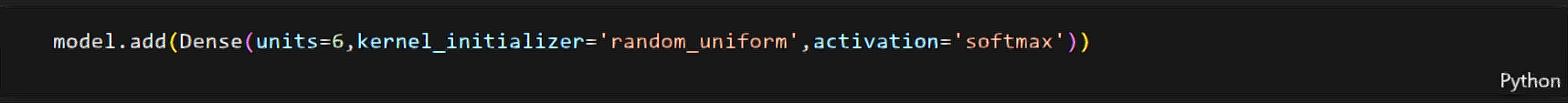
**Dense Layer :-**

After completed all the layers the single linear vector is connected to the full connected neural network is also dense layer. The dense layer means takes the input from the output of precise layers. We can add n number of dense layers. I am adding 6 Dense layers of activation function is Relu. The units as 128 means the output of the layers and the main point the last dense layer units should be the number of class present in your project the activation function should be SoftMax. SoftMax activation function is scales the numbers to probabilities of the each class.

A black background with many different colored lines

Description automatically generated

And the last Dense layer of units should be the 6 because of this project as 6 class are present. Activation function should be the SoftMax activation function.



A diagram of a network

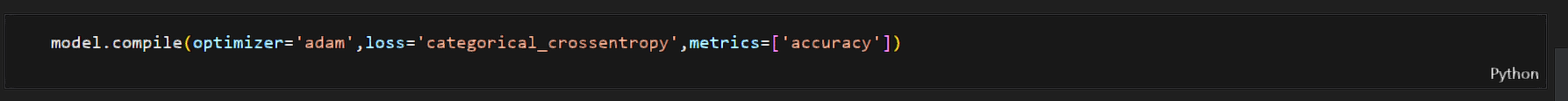
Description automatically generated

The basic fully connected neural network is shown above from the above green circles are the dense layer they are two layers are connected, the first layer of units are 6 and the second layer as 4 units. And the three classes are the output. For our project 6 dense layers are considered to be those green circle add 6 layers and units is 128 and 128 green circle are present in the first layer same for all the 4 layers and the last layer as units is 6 and 6 green circles are present in those layer.

A screenshot of a computer program

Description automatically generated

model.summary() is means the to check the how many layers are added to this model and number of dense layers and check the sequence of the layers are add to the model. To check the output shape height and width and features are shown from the MaxPooling layer the image is resized.



After added all the layers to the model and compile the model, for compiling the model we take the optimizer, loss function, metrics.

**Optimizers:-**

Optimizers are used to adjust the weights in the fully connected neural network. They are so many optimizers are present. Weights should be initialize in the fully connected neural network. Lets take an example.

W1

W2

(w)n

Dense layer Dense layer

Based on the weights the check the neuron is activated are not and the weights should be take by system and some default weights should be initialize by

the system based on those weights we check the accuracy and if the accuracy is fine its ok then accuracy is not good we need to change the weights for this changing of weights we added the optimizers and this optimizers change the weights until the accuracy is good. We consider the Adam Optimizer.

**Adam Optimizer:-**

Adam optimizer main focus on the hyperparameter they are so many parameters are present Adam mainly focus on the learning rate. The learning rate should be minimum the point should be reach the global minima**.**

**Loss Function:-**

Loss function means how well the model is performing and compare the actual output and the predicted output. They are two types of loss function in classification problem.

* Binary cross entropy/log loss.
* Categorical cross entropy.

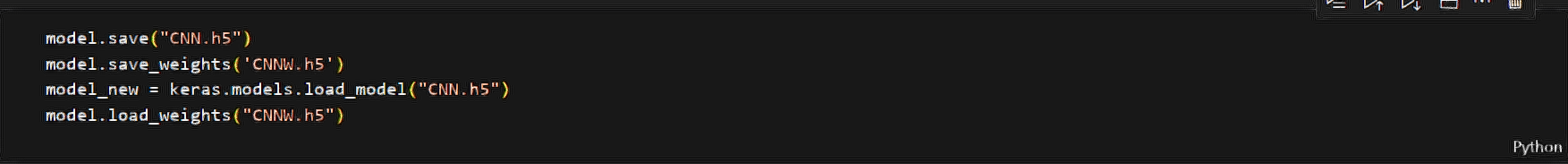
Binary cross entropy is used to binary class classification problem that means the output should be the 0 or 1 format in this classification only two class are present.

Categorical cross entropy is used to multi-class classification problem that means the output should be the n number of class. This project as multi-class classification that’s why we consider the categorical cross entropy.

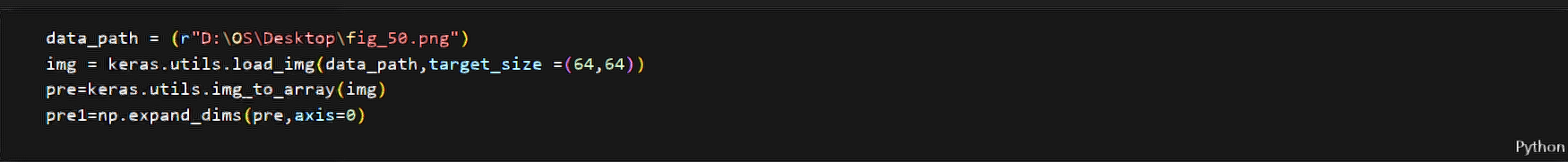
A screenshot of a computer

Description automatically generated

After compilation as done and model.fit\_generator to run the model give the training data to generator and steps\_per\_epoch means number of images taken for the 1 iteration for example we have 1000 images for steps\_per\_epoch is 100 images 10 iteration are taken for this project we consider the all the images and only iteration is taken same as the validation data and epochs means the full cycle of the training data and we consider 9 epochs that means the 9 times we run the full cycle and how many epochs should consider based on the accuracy and you should consider any number of epochs and if the accuracy is goes constant at the point you brake the epochs. In which number accuracy is more we should consider that number as epochs. Just check the last three epochs and the accuracy is 0.9594,0.9647,0.9692.



To save the model and weights and load the model and weights to run the above codes.



After load the model lets take an image to predict and path of an image is taken into the data\_path and load the image and the target\_size is 64,64 lets convert the image into the array format and next expand the features or dimensions and after the convert the image to an array then we pass into the model.

A screenshot of a computer

Description automatically generated

The image should be given to the model to predict the image belongs to which class. The model gives the probability to each class and which class as the highest probability that is the output that’s why we consider the np.argmax. Argmax is used to which class as the maximum probability and that class should be print as output. And class 2 as the maximum probability and consider the index of your class.