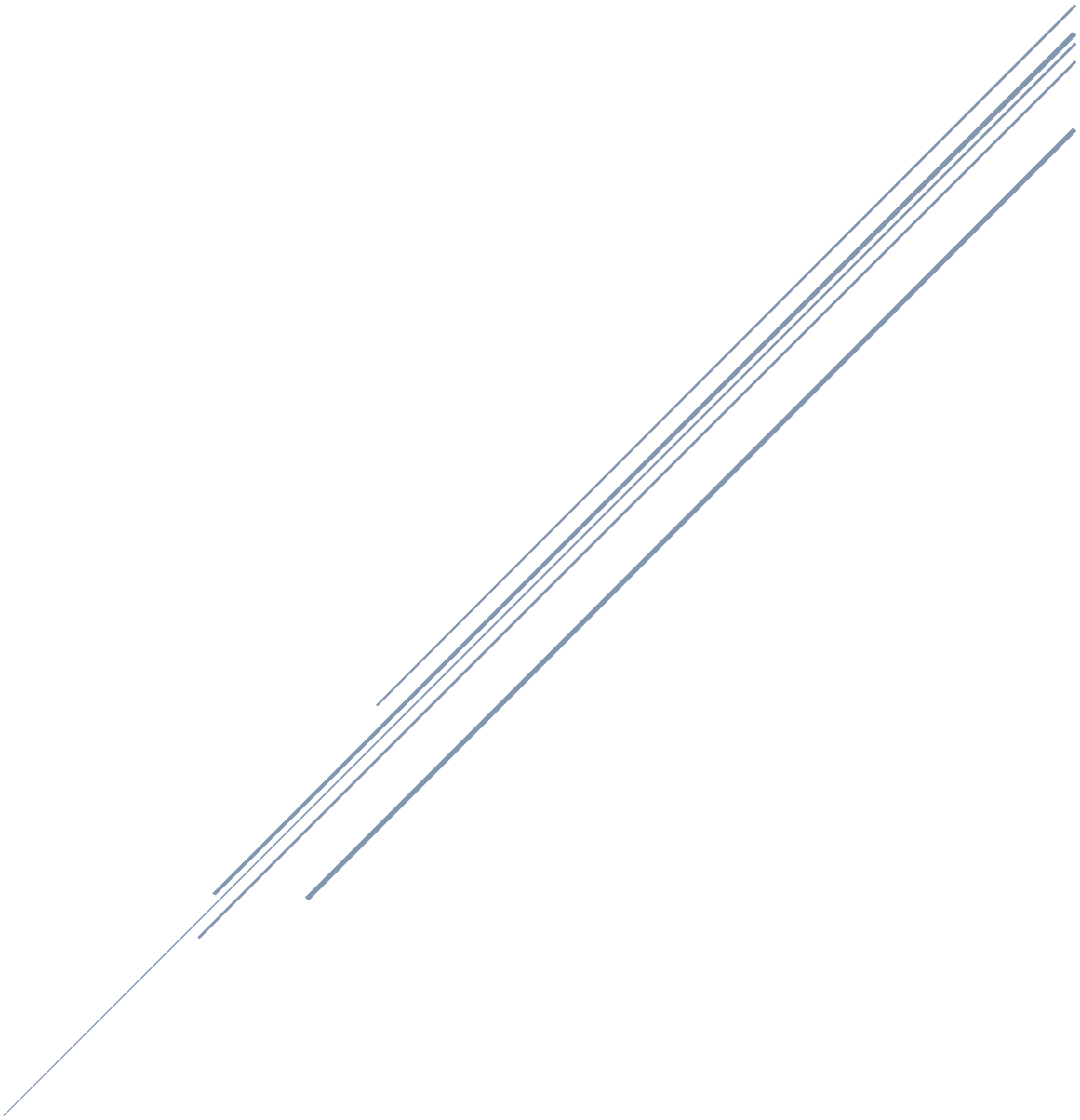
n NEURAL NETWORKS

Weather Classification

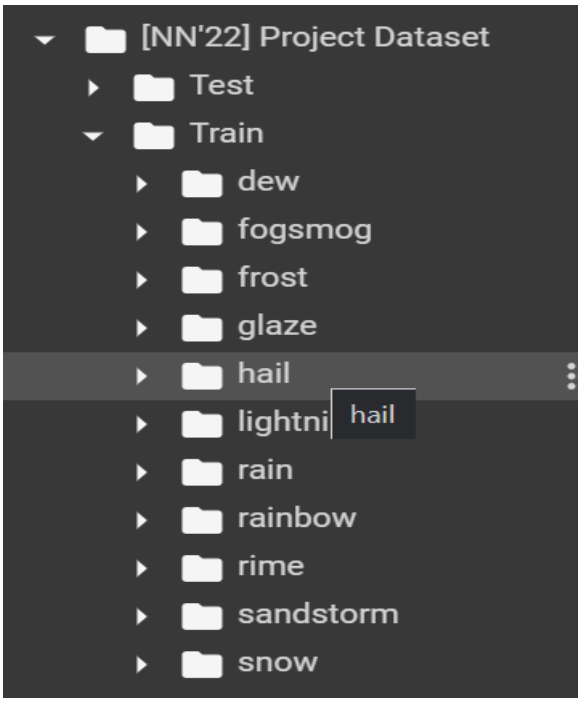


Weather Classification:

This notebook trains and tests a neural network using tensorflow and keras to classify images of weather conditions into 11 classes: ['snow', 'fogsmog', 'rainbow', 'hail', 'sandstorm', 'lightning', 'frost', 'rain', 'rime', 'glaze', 'dew']

Data preparation:

Dataset is split into train and test and train must be split into 11 folders, one for each class, with each named after the class of images it contains. I.e. all 'snow' images must be in :"/content/drive/MyDrive/[NN'22] Project Dataset/Train/snow" after mounting the drive



Preprocessing:

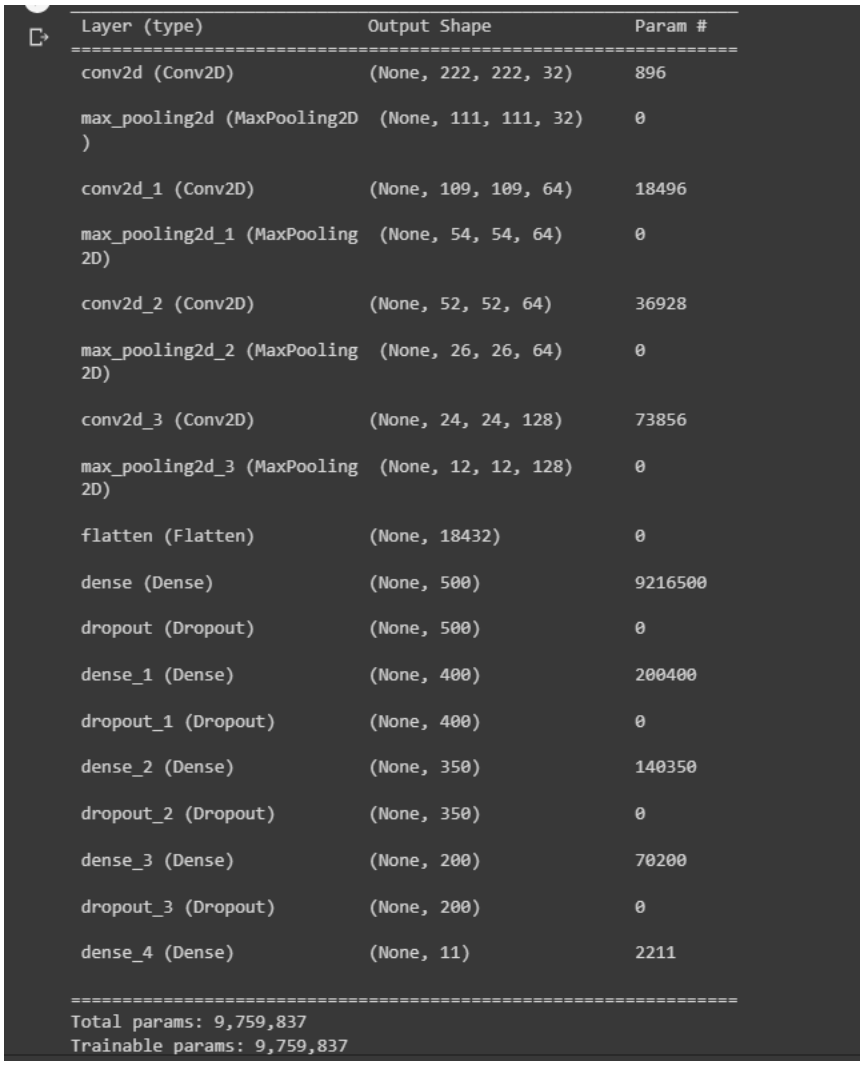
1. We the added data into folder “weather_data”
2. We splitted weather_data into train and validation
3. Train containing 80% of the data and validation containing 20%
4. We augmented the data using Image Generator [Horizontal flip/zooming]
5. We normalized the images.

Training:

Architectures:

We will use CNN with architecture, VGG16, and AlexNet:

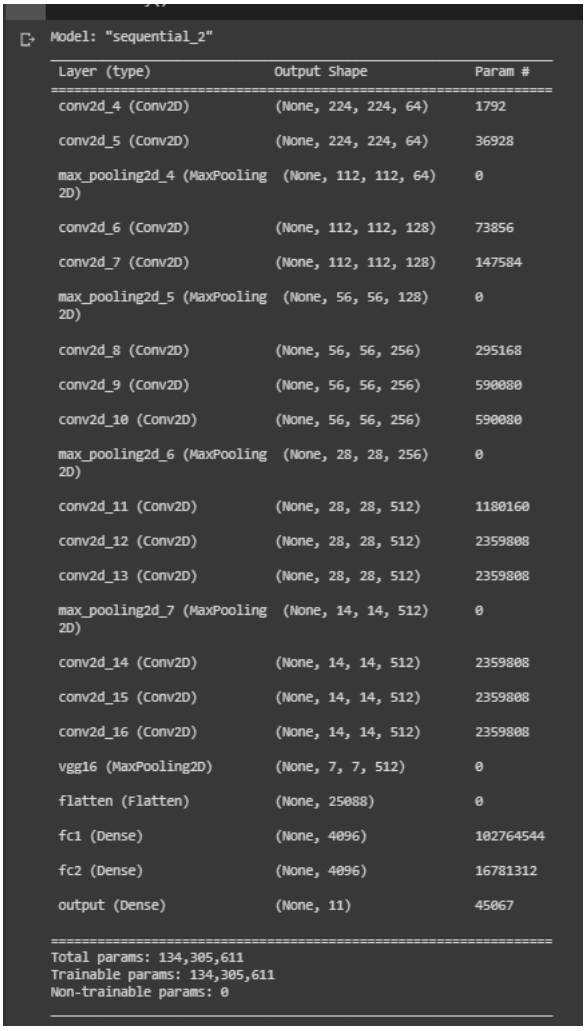
- **CNN Architecture:**



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 222, 222, 32)	896
max_pooling2d (MaxPooling2D)	(None, 111, 111, 32)	0
conv2d_1 (Conv2D)	(None, 109, 109, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 64)	0
conv2d_2 (Conv2D)	(None, 52, 52, 64)	36928
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 64)	0
conv2d_3 (Conv2D)	(None, 24, 24, 128)	73856
max_pooling2d_3 (MaxPooling2D)	(None, 12, 12, 128)	0
flatten (Flatten)	(None, 18432)	0
dense (Dense)	(None, 500)	9216500
dropout (Dropout)	(None, 500)	0
dense_1 (Dense)	(None, 400)	200400
dropout_1 (Dropout)	(None, 400)	0
dense_2 (Dense)	(None, 350)	140350
dropout_2 (Dropout)	(None, 350)	0
dense_3 (Dense)	(None, 200)	70200
dropout_3 (Dropout)	(None, 200)	0
dense_4 (Dense)	(None, 11)	2211

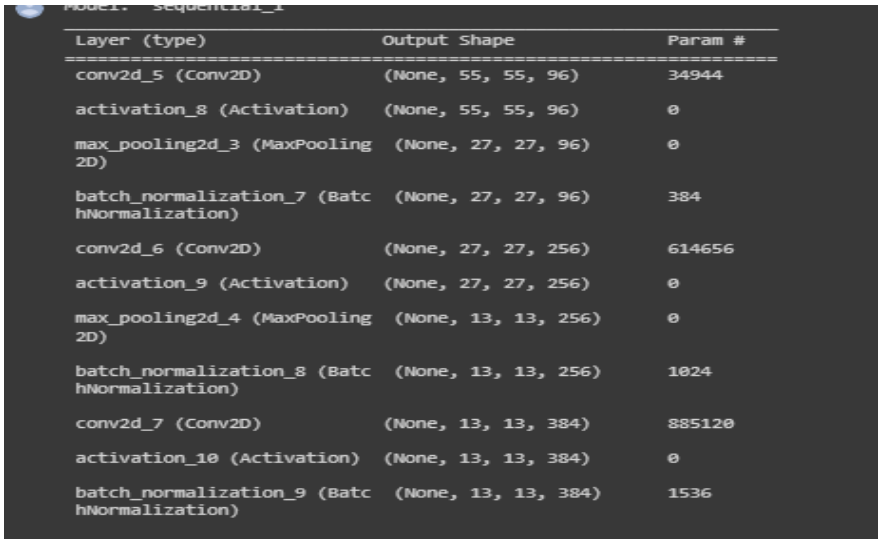
Total params: 9,759,837
Trainable params: 9,759,837

- **VGG Architecture:**

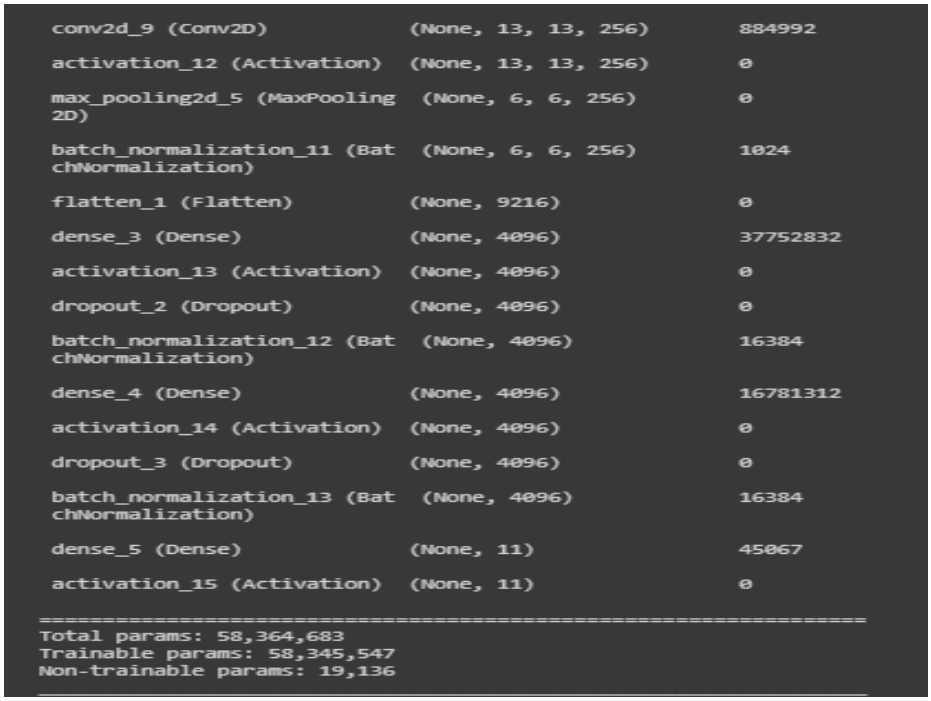


```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_4 (Conv2D)            (None, 224, 224, 64)      1792

conv2d_5 (Conv2D)            (None, 224, 224, 64)      36928

max_pooling2d_4 (MaxPooling  (None, 112, 112, 64)      0
2D)

conv2d_6 (Conv2D)            (None, 112, 112, 128)     73856

conv2d_7 (Conv2D)            (None, 112, 112, 128)     147584

max_pooling2d_5 (MaxPooling  (None, 56, 56, 128)       0
2D)

conv2d_8 (Conv2D)            (None, 56, 56, 256)       295168

conv2d_9 (Conv2D)            (None, 56, 56, 256)       590080

conv2d_10 (Conv2D)           (None, 56, 56, 256)       590080

max_pooling2d_6 (MaxPooling  (None, 28, 28, 256)       0
2D)

conv2d_11 (Conv2D)           (None, 28, 28, 512)       1180160

conv2d_12 (Conv2D)           (None, 28, 28, 512)       2359808

conv2d_13 (Conv2D)           (None, 28, 28, 512)       2359808

max_pooling2d_7 (MaxPooling  (None, 14, 14, 512)       0
2D)

conv2d_14 (Conv2D)           (None, 14, 14, 512)       2359808

conv2d_15 (Conv2D)           (None, 14, 14, 512)       2359808

conv2d_16 (Conv2D)           (None, 14, 14, 512)       2359808

vgg16 (MaxPooling2D)         (None, 7, 7, 512)         0

flatten (Flatten)            (None, 25088)             0

fc1 (Dense)                  (None, 4096)              102764544

fc2 (Dense)                  (None, 4096)              16781312

output (Dense)               (None, 11)                45067

=================================================================
Total params: 134,305,611
Trainable params: 134,305,611
Non-trainable params: 0
_________________________________________________________________
```

- **AlexNet Architecture:**



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 55, 55, 96)        34944

activation_8 (Activation)    (None, 55, 55, 96)        0

max_pooling2d_3 (MaxPooling  (None, 27, 27, 96)        0
2D)

batch_normalization_7 (Batc  (None, 27, 27, 96)        384
hNormalization)

conv2d_6 (Conv2D)            (None, 27, 27, 256)       614656

activation_9 (Activation)    (None, 27, 27, 256)       0

max_pooling2d_4 (MaxPooling  (None, 13, 13, 256)       0
2D)

batch_normalization_8 (Batc  (None, 13, 13, 256)       1024
hNormalization)

conv2d_7 (Conv2D)            (None, 13, 13, 384)       885120

activation_10 (Activation)   (None, 13, 13, 384)       0

batch_normalization_9 (Batc  (None, 13, 13, 384)       1536
hNormalization)
```



```
conv2d_9 (Conv2D)            (None, 13, 13, 256)       884992

activation_12 (Activation)   (None, 13, 13, 256)       0

max_pooling2d_5 (MaxPooling  (None, 6, 6, 256)         0
2D)

batch_normalization_11 (Bat  (None, 6, 6, 256)         1024
chNormalization)

flatten_1 (Flatten)          (None, 9216)              0

dense_3 (Dense)              (None, 4096)              37752832

activation_13 (Activation)   (None, 4096)              0

dropout_2 (Dropout)          (None, 4096)              0

batch_normalization_12 (Bat  (None, 4096)              16384
chNormalization)

dense_4 (Dense)              (None, 4096)              16781312

activation_14 (Activation)   (None, 4096)              0

dropout_3 (Dropout)          (None, 4096)              0

batch_normalization_13 (Bat  (None, 4096)              16384
chNormalization)

dense_5 (Dense)              (None, 11)                45067

activation_15 (Activation)   (None, 11)                0

=================================================================
Total params: 58,364,683
Trainable params: 58,345,547
Non-trainable params: 19,136
_________________________________________________________________
```

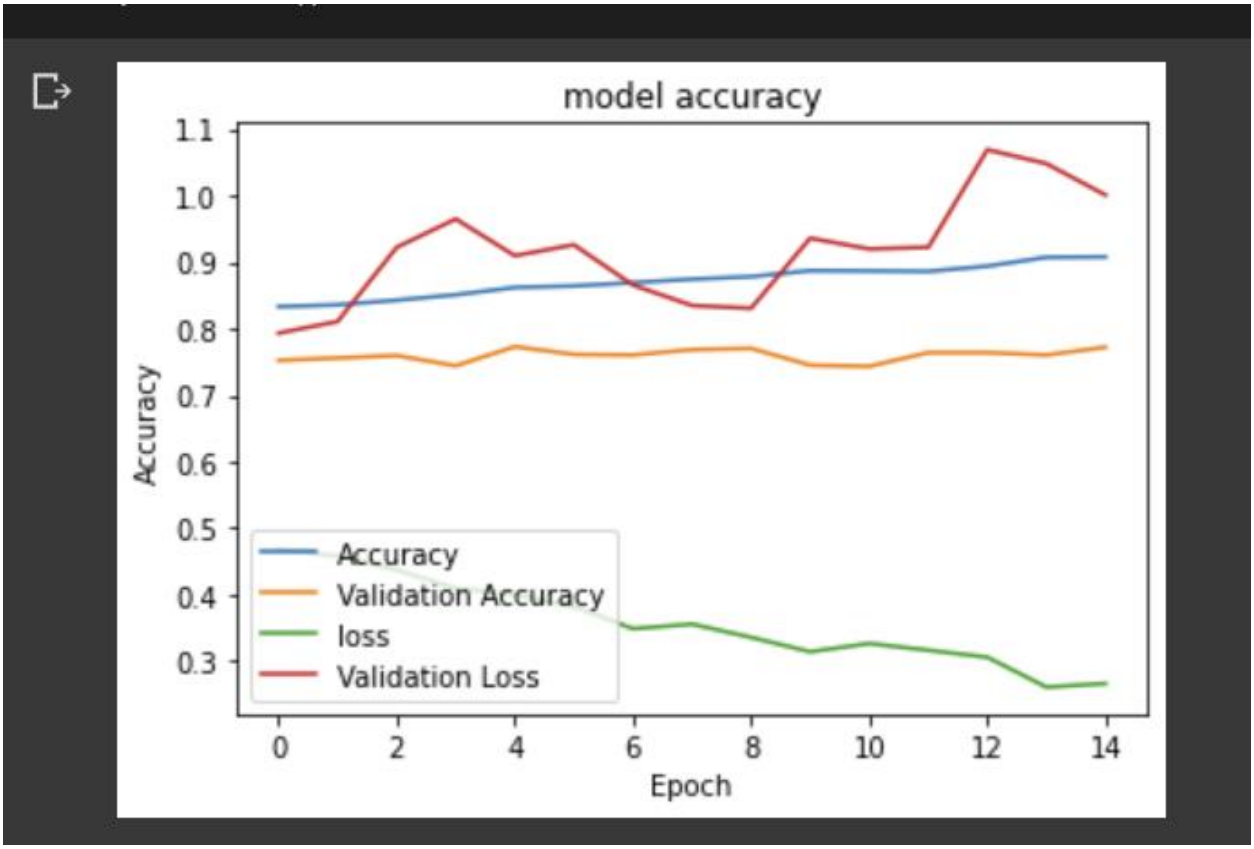
Hyper parameters:

- **Note:**

  Early stopping was applied on all the Models to prevent the over-fitting problem.

	CNN Architecture	**VGG16 Architecture**	**AlexNet Architecture**
Optimizer	Adam	SGD	Adam
Loss	Categorical Cross entropy	Categorical Cross entropy	Categorical Cross entropy
Metrics	Accuracy	Accuracy	Accuracy
Epochs	75	30	100 - Due to the early stopping, the model training stopped after 21 epochs

VGG:


model accuracy
Accuracy
Epoch
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
2
4
6
8
10
12
14
Accuracy
Validation Accuracy
loss
Validation Loss

Result:

CNN Architecture:

- It scored an accuracy of 83.12%.
- It has on Kaggle a test accuracy of 70.4% in public score and 73% in private score.

VGG16 Architecture:

- It scored an accuracy of 77%.
- It has a test accuracy of 55% in public score and 64% in private score.

AlexNet Architecture:

- It scored an accuracy of 66%.
- It has a test accuracy of 64.6% in public score and 69.5% in private score.

Conclusion:

The CNN Architecture scored the best accuracy, the AlexNet Architecture comes next, then the VGG16 Architecture.

As we see, CNN Architecture is the best model among the rest of the models mentioned above.