Assignment – 2

Network Architecture Search, Hyper parameter Tuning, and Transfer Learning

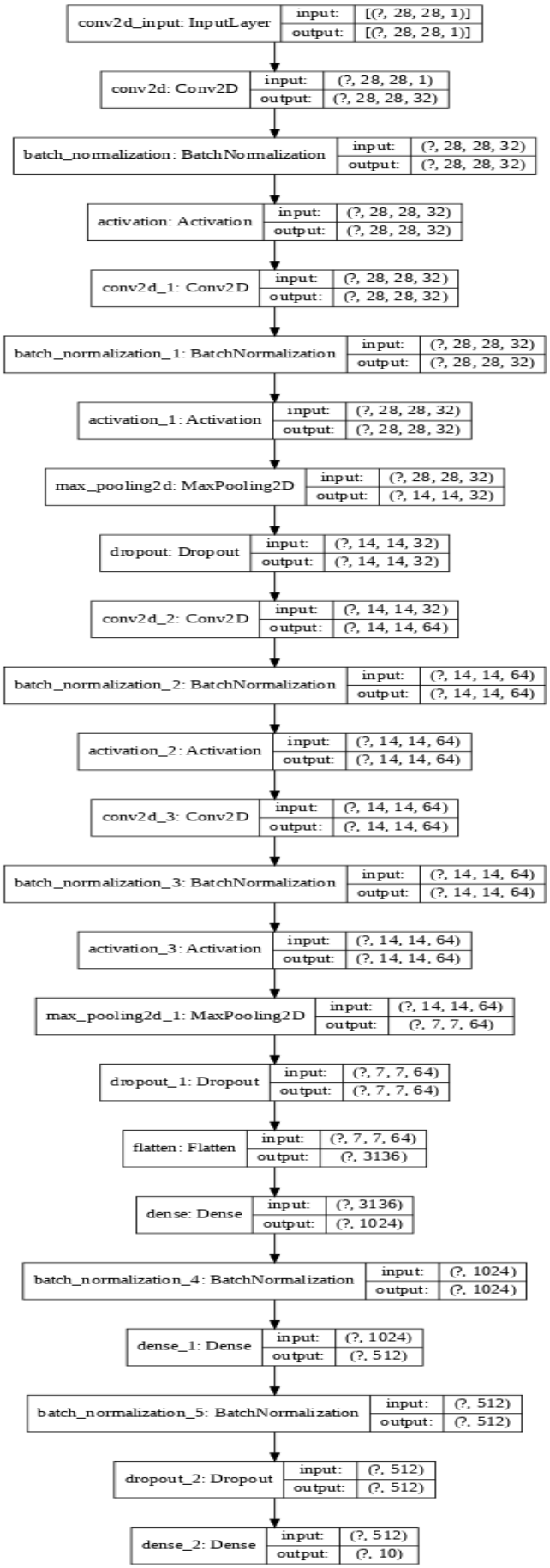
Question : 1

Part A:

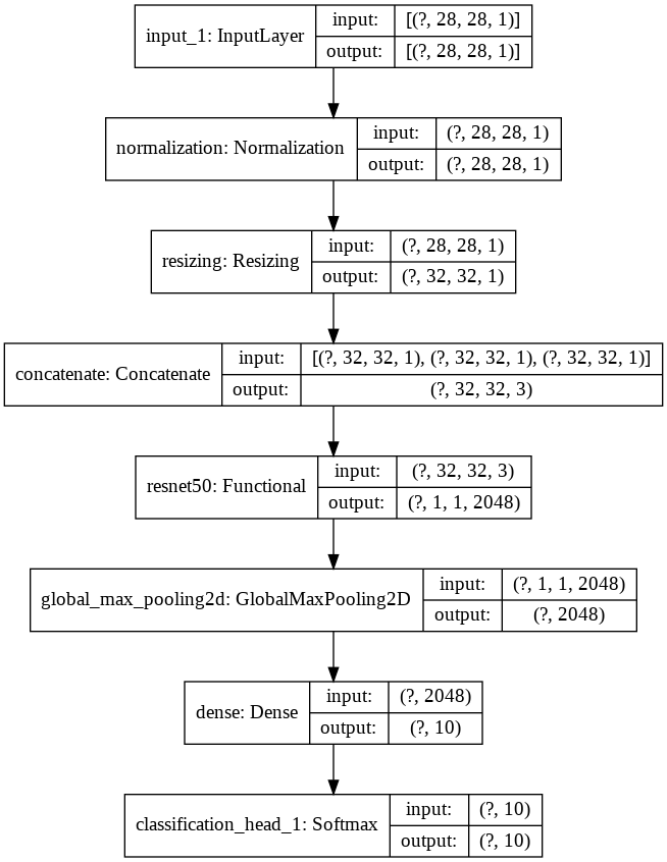
The model Architecture consist of two convolution layer with the first layer initialized with he_uniform kernel initializer. Each convolution layer has a BatchNormalization and an Activation layer with RELU activation. The 2nd and the 4th Conv2d layer have a Maxpooling layer with dropout of 0.3. The first two layers have a filter size of 32 and kernel size of 5. The 3rd and 4th layer have a filter size of 64 and kernel size of 3. The Convo layers are followed by a flatten layer and three dense layer. The first two layers are followed by a BatchNormalization layer and has Relu activation. The second Dense layer has a dropout of 0.20. There are 1024 neurons in the first layer and 512 neuron in the second Dense layer. The final layer is a softmax output layer with 10 neurons for classification. The model has a total of 3,831,018 parameters with 3,827,562 trainable and 3,456 non trainable parameters. The model accuracy was 92.43% on test dataset.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d (Conv2D)              (None, 28, 28, 32)        832       
_________________________________________________________________
batch_normalization (BatchNo (None, 28, 28, 32)        128       
_________________________________________________________________
activation (Activation)      (None, 28, 28, 32)        0         
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 28, 28, 32)        25632     
_________________________________________________________________
batch_normalization_1 (Batch (None, 28, 28, 32)        128       
_________________________________________________________________
activation_1 (Activation)    (None, 28, 28, 32)        0         
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 32)        0         
_________________________________________________________________
dropout (Dropout)            (None, 14, 14, 32)        0         
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 14, 64)        18496     
_________________________________________________________________
batch_normalization_2 (Batch (None, 14, 14, 64)        256       
_________________________________________________________________
activation_2 (Activation)    (None, 14, 14, 64)        0         
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 14, 14, 64)        36928     
_________________________________________________________________
batch_normalization_3 (Batch (None, 14, 14, 64)        256       
_________________________________________________________________
activation_3 (Activation)    (None, 14, 14, 64)        0         
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 64)          0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 7, 7, 64)          0         
_________________________________________________________________
flatten (Flatten)            (None, 3136)              0         
_________________________________________________________________
dense (Dense)                (None, 1024)              3212288   
_________________________________________________________________
batch_normalization_4 (Batch (None, 1024)              4096      
_________________________________________________________________
dense_1 (Dense)              (None, 512)               524800    
_________________________________________________________________
batch_normalization_5 (Batch (None, 512)               2048      
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 10)                5130      
=================================================================
Total params: 3,831,018
Trainable params: 3,827,562
Non-trainable params: 3,456
_________________________________________________________________
```

Hand Written model



Auto Keras model



Part B:

I use automodel in autokeras with ResNet architecture, I have set Normalize to true and Data Augmentation to false. When I kept Data Augmentation to true the model does not generalize properly. The model that Autokeras derived is an Functional model with a input layer, normalization,resizing,concatenate, resnet50, global max pooling2d followed by dense layer, and a classification head with resnet activation. There are a total of 23,608,205 params and 23,555,082 trainable params, 53,123 non-trainable params.

```
Model: "functional_1"
_________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
=================================================================================================
input_1 (InputLayer)         [(None, 28, 28, 1)]       0
_________________________________________________________________________________________________
normalization (Normalization)(None, 28, 28, 1)         3           input_1[0][0]
_________________________________________________________________________________________________
resizing (Resizing)          (None, 32, 32, 1)         0           normalization[0][0]
_________________________________________________________________________________________________
concatenate (Concatenate)    (None, 32, 32, 3)         0           resizing[0][0]
                                                                   resizing[0][0]
                                                                   resizing[0][0]
_________________________________________________________________________________________________
resnet50 (Functional)        (None, 1, 1, 2048)        23587712    concatenate[0][0]
_________________________________________________________________________________________________
global_max_pooling2d (GlobalMax (None, 2048)           0           resnet50[0][0]
_________________________________________________________________________________________________
dense (Dense)                (None, 10)                20490       global_max_pooling2d[0][0]
_________________________________________________________________________________________________
classification_head_1 (Softmax) (None, 10)             0           dense[0][0]
=================================================================================================
Total params: 23,608,205
Trainable params: 23,555,082
Non-trainable params: 53,123
_________________________________________________________________________________________________
```

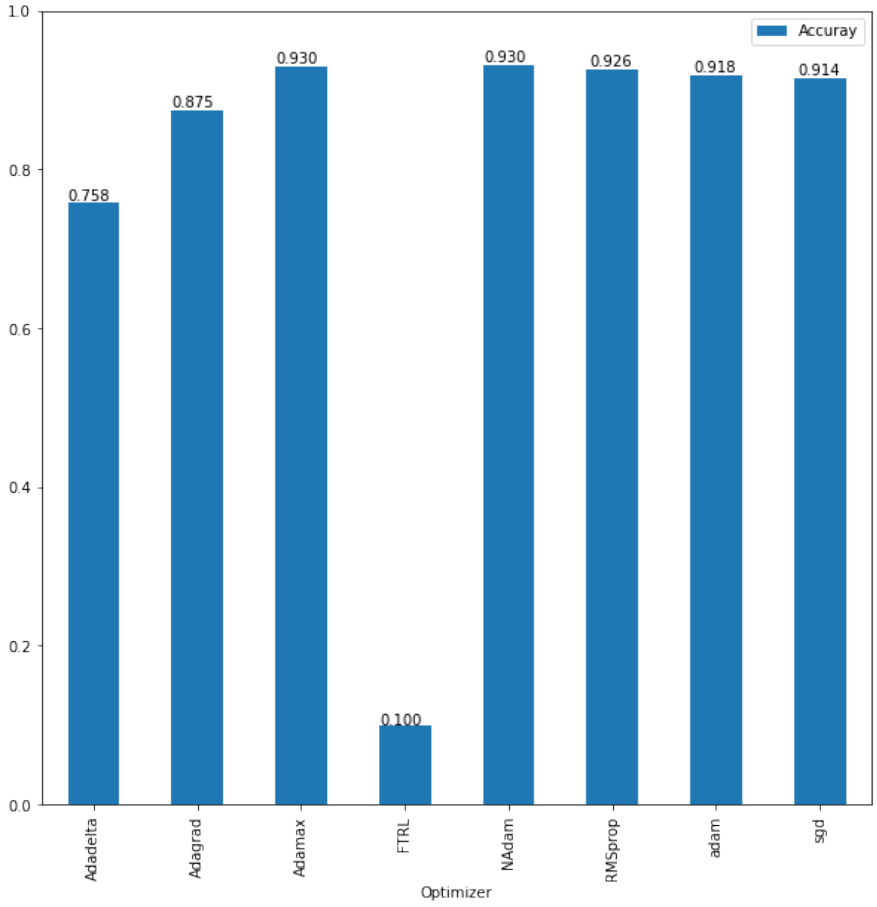
Observation:

1. The Autokeras model took a very long time to run. It was able to explore multiple type of architectures.
2. I was able to tune different type of parameters such as augmentation, normalization, block type.
3. The autokeras model I obtained from running the Automodel gave lesser accuracy than the hand written model. Where the test accuracy of 92.10%.

Question : 2

Part A:

I ran 10 epochs using all the 8 optimizers mentioned in keras optimizers. Using Hparams optimizer. I found NAdam To be the best Optimizer from the keras Optimizers with an accuracy of 93.05% accuracy.



	Optimizer	Accuray
0	Adadelta	0.7580
1	Adagrad	0.8747
2	Adamax	0.9299
3	FTRL	0.1000
4	NAdam	0.9305
5	RMSprop	0.9257
6	adam	0.9183
7	sgd	0.9140

PART B:

I used the Nadam optimizer and ran Hparam optimization with relu,selu and elu activation on 4 convolution layers and the best configuration is as follows.

Layer	Old Activation	New Activation
Conv2d(filter size 32, kernel size [5,5])	relu	Selu
Conv2d (filter size 32, kernel size [5,5])	relu	relu
Conv2d (filter size 64, kernel size [3,3])	relu	relu
Conv2d (filter size 64, kernel size [3,3])	relu	relu

Dense(1024, activation='relu')	relu	relu
Dense(512, activation='relu')	relu	relu

```
The best model combination is as follows:

combination no    combination_68
Activation1                selu
Activation2                relu
Activation3                relu
Activation4                relu
Accuracy                 0.9337
Name: 67, dtype: object
```

Part C:

Using the best configuration from the part B I build the model and changed the filters on the Convo layer and using Hparams. The best model configuration is as follows.

Layer	Activation	New layer configuration
Conv2d(filter size 32, kernel size [5,5])	Selu	Conv2d(filter size 32, kernel size [5,5])
Conv2d (filter size 32, kernel size [5,5])	relu	Conv2d (filter size 64, kernel size [5,5])
Conv2d (filter size 64, kernel size [3,3])	relu	Conv2d (filter size 64, kernel size [3,3])
Conv2d (filter size 64, kernel size [3,3])	relu	Conv2d (filter size 64, kernel size [3,3])
Dense(1024, activation='relu')	relu	Dense(1024, activation='relu')
Dense(512, activation='relu')	relu	Dense(512, activation='relu')

```
The best model combination is as follows:

combination no    combination_54
Activation1                   32
Activation2                   64
Activation3                   64
Activation4                   64
Accuracy                  0.9338
Name: 53, dtype: object
```

The final accuracy after exploration was 93.38% which is almost 1 % more than the initial accuracy.

Question 3:

Part A:

Base model: MobilenetV2 model.

The model architecture for transfer learning is as follows:

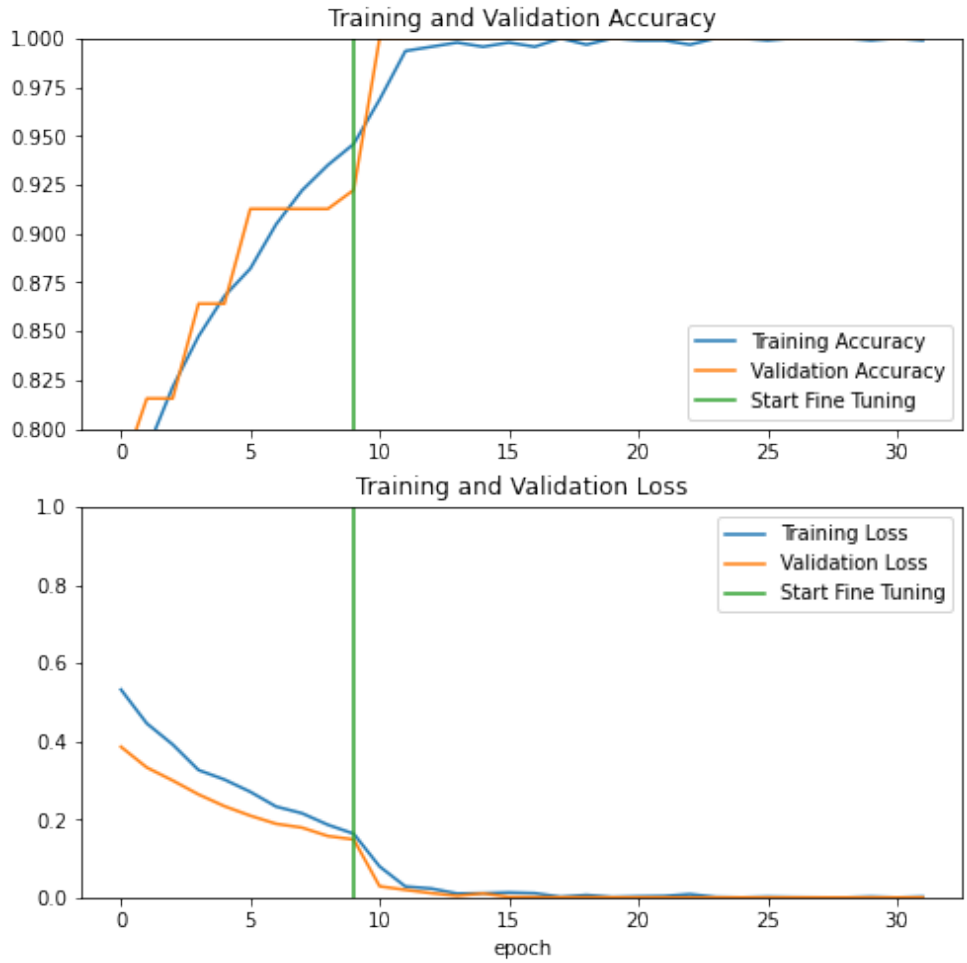
Input layer of shape (224,224,3) , data_augmentation layer , pre_processing layer, basemodel, global average layer, Dropout of 0.4 drop out. And dense layer with predictions.

The validation accuracy of the model is given as follows.

Pretuning accuracy	Post fine tuning accuracy
0.9223	1.0000

Training and validation graph for first 10 epochs

Training and validation graph after fine tuning:



PART B:

Base model: InceptionV3 model.

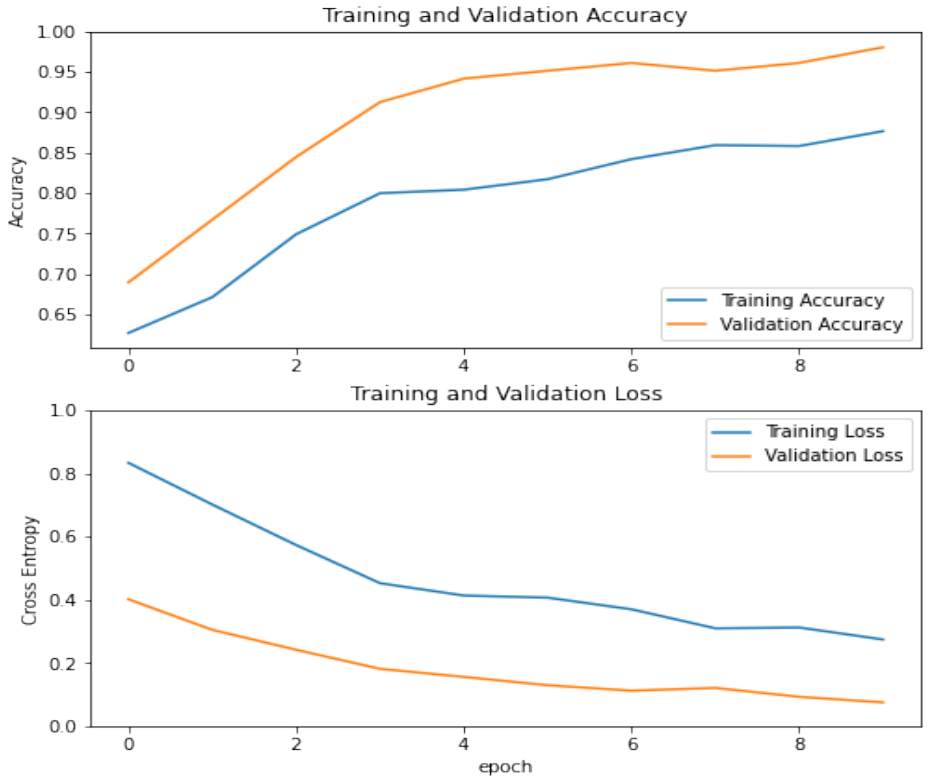
The model architecture for transfer learning is as follows:

Input layer of shape (224,224,3) , data_augmentation layer , pre_processing layer, basemodel, global average layer, Dropout of 0.4 drop out. And dense layer with predictions.

The validation accuracy of the model is given as follows.

Pre tuning accuracy	Post fine tuning accuracy
0.9806	1.0000

Training and validation graph for first 10 epochs



Training and validation graph after fine tuning