Kaggle user: Milin D Desai
CSC 578 HW 6
Milin Desai

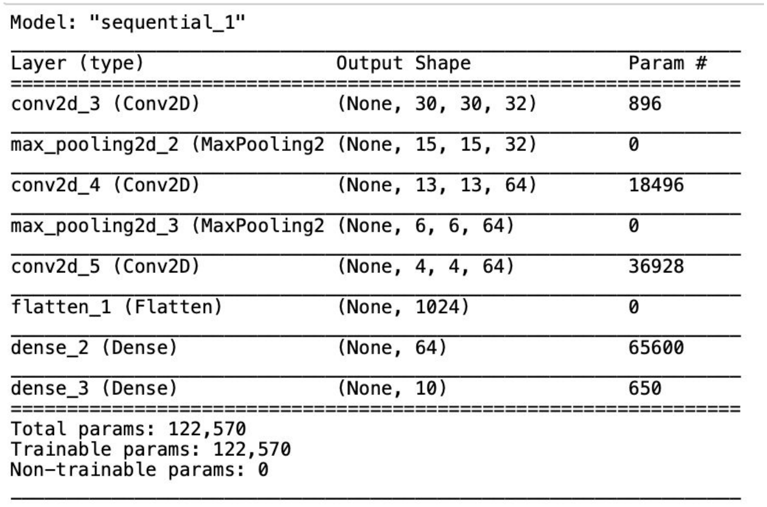
Journey

Note : For a I have started executing code after installing tensor flow version two so I don't need to uncomment anything from the code.

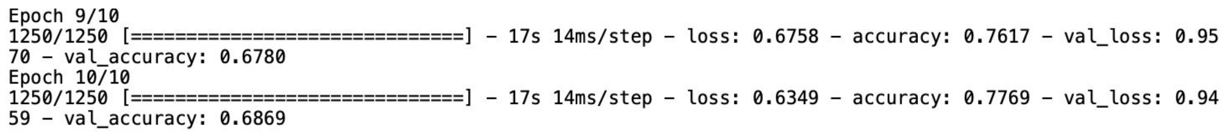
Twisting Hyperparameters:

Model 0:

As below you can see the summary of the model zero which was provided by the professor.



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 13, 13, 64)        18496
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 4, 4, 64)          36928
_________________________________________________________________
flatten_1 (Flatten)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                65600
_________________________________________________________________
dense_3 (Dense)              (None, 10)                650
=================================================================
Total params: 122,570
Trainable params: 122,570
Non-trainable params: 0
_________________________________________________________________
```

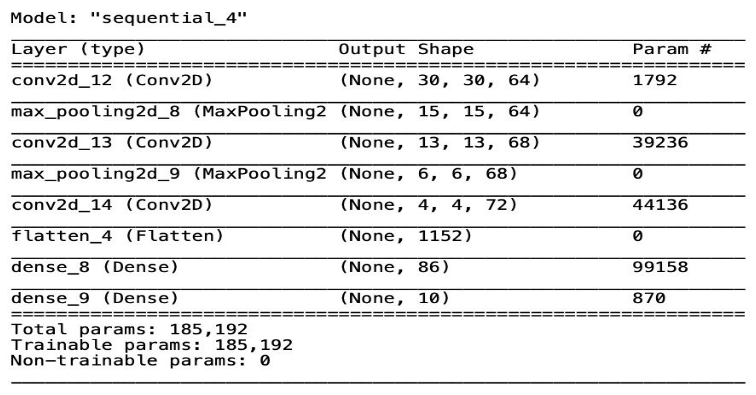


```
Epoch 9/10
1250/1250 [==============================] - 17s 14ms/step - loss: 0.6758 - accuracy: 0.7617 - val_loss: 0.95
70 - val_accuracy: 0.6780
Epoch 10/10
1250/1250 [==============================] - 17s 14ms/step - loss: 0.6349 - accuracy: 0.7769 - val_loss: 0.94
59 - val_accuracy: 0.6869
```

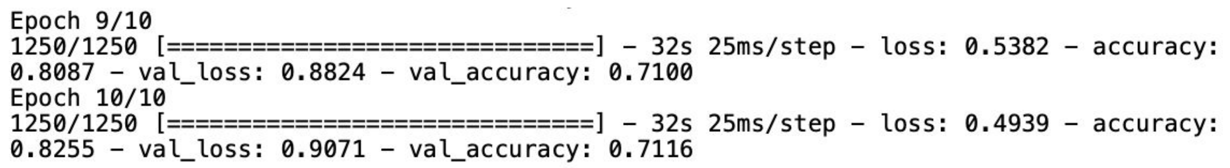
Model 0 is the initial model, I haven't changed anything in the code with that I am getting accuracy of 77% with loss of 63.49% on training set, where in the testing set accuracy of 68% and loss of 94%

Model 1

For Model 1 I have updated the number of filters hidden layers, I haven't touch the input and output layer's number of filter summary of the model you can see as below:



```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_12 (Conv2D)           (None, 30, 30, 64)        1792
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 15, 15, 64)        0
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 13, 13, 68)        39236
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 6, 6, 68)          0
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 4, 4, 72)          44136
_________________________________________________________________
flatten_4 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_8 (Dense)              (None, 86)                99158
_________________________________________________________________
dense_9 (Dense)              (None, 10)                870
=================================================================
Total params: 185,192
Trainable params: 185,192
Non-trainable params: 0
_________________________________________________________________
```

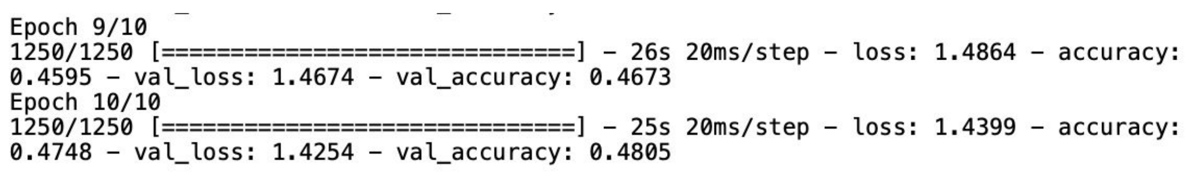


```
Epoch 9/10
1250/1250 [==============================] - 32s 25ms/step - loss: 0.5382 - accuracy:
0.8087 - val_loss: 0.8824 - val_accuracy: 0.7100
Epoch 10/10
1250/1250 [==============================] - 32s 25ms/step - loss: 0.4939 - accuracy:
0.8255 - val_loss: 0.9071 - val_accuracy: 0.7116
```

From the above results you can see this model perform better than the previous model as you can see it achieved accuracy of 82% on training and 71% on the test set with respective loss with the 49% and 90%. Time per second is also not too high so this model is efficient.

Model 2

For Model 2 I have updated the activation form "softmax" in the hidden layer in place of the using "relu" activation form, I haven't touch the input and output layer's number of filter summary of the model you can see as below:

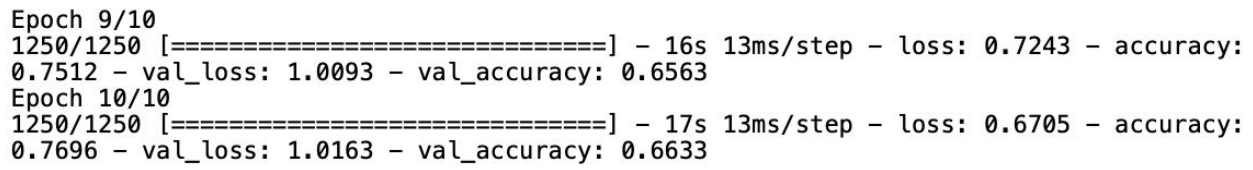


```
Epoch 9/10
1250/1250 [==============================] - 26s 20ms/step - loss: 1.4864 - accuracy:
0.4595 - val_loss: 1.4674 - val_accuracy: 0.4673
Epoch 10/10
1250/1250 [==============================] - 25s 20ms/step - loss: 1.4399 - accuracy:
0.4748 - val_loss: 1.4254 - val_accuracy: 0.4805
```

From the above results you can see that the model performed really poorly on both the test and training set. Also, loss on both sets is high so I won't consider using this model.

Model 3:

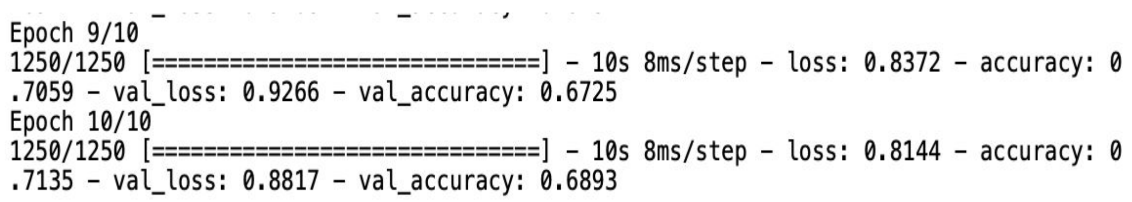
In model 3, I have updated the activation form to " tanh" so performance improved but not as the other model so I won't use this model.



```
Epoch 9/10
1250/1250 [==============================] - 16s 13ms/step - loss: 0.7243 - accuracy:
0.7512 - val_loss: 1.0093 - val_accuracy: 0.6563
Epoch 10/10
1250/1250 [==============================] - 17s 13ms/step - loss: 0.6705 - accuracy:
0.7696 - val_loss: 1.0163 - val_accuracy: 0.6633
```

Model 4

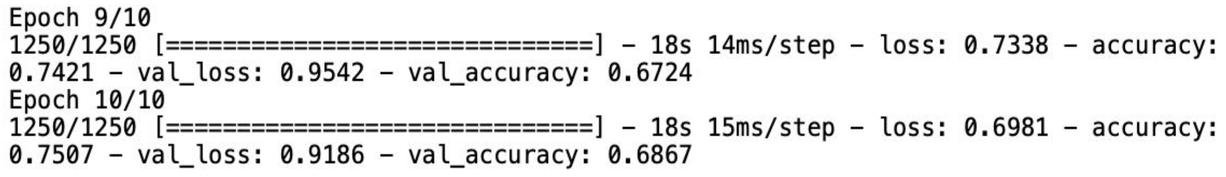
In the Model 4 I have interchange the size of filters and run for the model, accuracy is 71% and 68% respectively for the training and testing set and losses are 0.81 and 0.68 which better than the previous model but I will still won't consider it as best model as accuracy is lower than the second model.



```
Epoch 9/10
1250/1250 [==============================] - 10s 8ms/step - loss: 0.8372 - accuracy: 0
.7059 - val_loss: 0.9266 - val_accuracy: 0.6725
Epoch 10/10
1250/1250 [==============================] - 10s 8ms/step - loss: 0.8144 - accuracy: 0
.7135 - val_loss: 0.8817 - val_accuracy: 0.6893
```

Model 5

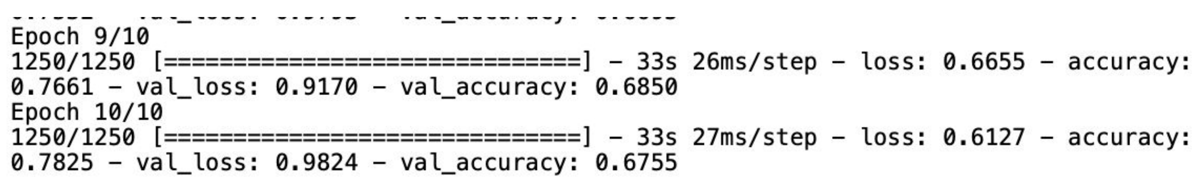
Accuracy changed on the training set with 75% and computation time decreased by almost 10ms when I added another convolution layer with 32 to filter with size (2 *2). Also, this model performed better then our best model, losses are also less then the previous model.



```
Epoch 9/10
1250/1250 [==============================] - 18s 14ms/step - loss: 0.7338 - accuracy:
0.7421 - val_loss: 0.9542 - val_accuracy: 0.6724
Epoch 10/10
1250/1250 [==============================] - 18s 15ms/step - loss: 0.6981 - accuracy:
0.7507 - val_loss: 0.9186 - val_accuracy: 0.6867
```

Model 6

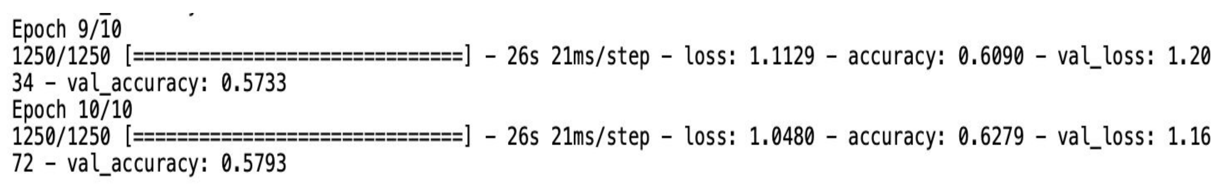
In this I have updated model 5 mainly activation form and number of filters and its size accuracy improved by 3% on the training set but still failed to be more efficient model because of computational time and losses.



```
Epoch 9/10
1250/1250 [==============================] - 33s 26ms/step - loss: 0.6655 - accuracy:
0.7661 - val_loss: 0.9170 - val_accuracy: 0.6850
Epoch 10/10
1250/1250 [==============================] - 33s 27ms/step - loss: 0.6127 - accuracy:
0.7825 - val_loss: 0.9824 - val_accuracy: 0.6755
```

Model 7

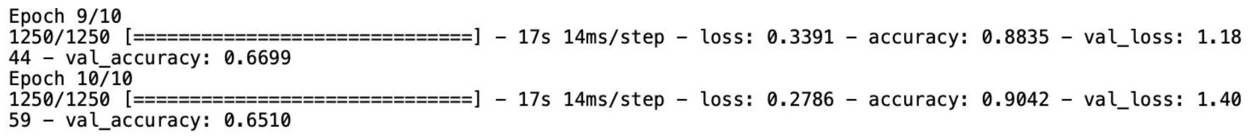
For better accuracy I’ve tried to remove the convolution layer from the model and then run the code Which worked really well. I've got accuracy of 90% which is the best accuracy I’ve got so far from all models on the training model. But testing set’s accuracy tells us overfitting in the model so this model is too easy to work with so we have to try a couple of hyperparameters.



```
Epoch 9/10
1250/1250 [==============================] - 26s 21ms/step - loss: 1.1129 - accuracy: 0.6090 - val_loss: 1.20
34 - val_accuracy: 0.5733
Epoch 10/10
1250/1250 [==============================] - 26s 21ms/step - loss: 1.0480 - accuracy: 0.6279 - val_loss: 1.16
72 - val_accuracy: 0.5793
```

Model 8

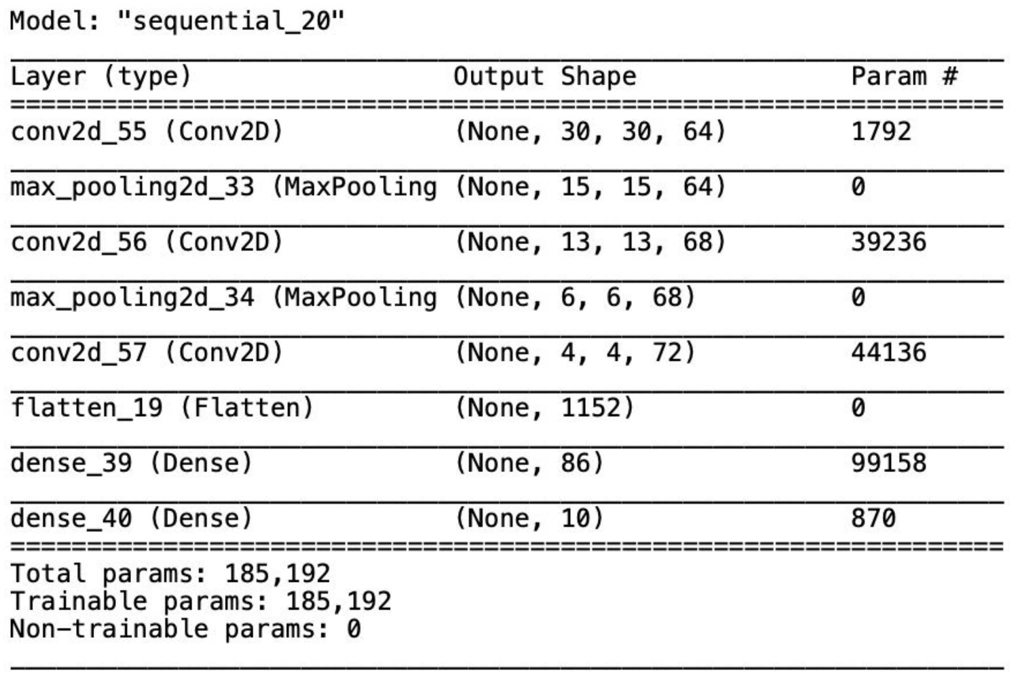
Again I’ve updated the model 7 by changing the activation form to overcome the overfitting problem but by doing so I also reduced the accuracy of the model although computational time increased compared to previous model but normal as compared to other model.



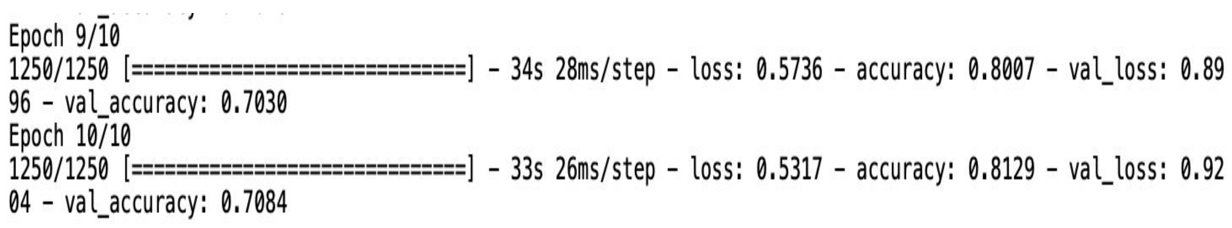
```
Epoch 9/10
1250/1250 [==============================] - 17s 14ms/step - loss: 0.3391 - accuracy: 0.8835 - val_loss: 1.18
44 - val_accuracy: 0.6699
Epoch 10/10
1250/1250 [==============================] - 17s 14ms/step - loss: 0.2786 - accuracy: 0.9042 - val_loss: 1.40
59 - val_accuracy: 0.6510
```

Best Model:

In this model, I updated the number of filters from the initial code. Which resulted in the highest accuracy among all models. There was also no evidence of cover fitting in training and testing set accuracy. In addition, computational time was also normal compared to the other models.

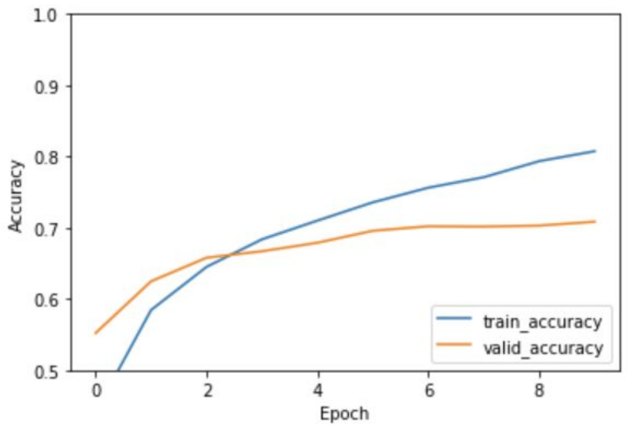


```
Model: "sequential_20"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_55 (Conv2D)           (None, 30, 30, 64)        1792
_________________________________________________________________
max_pooling2d_33 (MaxPooling (None, 15, 15, 64)        0
_________________________________________________________________
conv2d_56 (Conv2D)           (None, 13, 13, 68)        39236
_________________________________________________________________
max_pooling2d_34 (MaxPooling (None, 6, 6, 68)          0
_________________________________________________________________
conv2d_57 (Conv2D)           (None, 4, 4, 72)          44136
_________________________________________________________________
flatten_19 (Flatten)         (None, 1152)              0
_________________________________________________________________
dense_39 (Dense)             (None, 86)                99158
_________________________________________________________________
dense_40 (Dense)             (None, 10)                870
=================================================================
Total params: 185,192
Trainable params: 185,192
Non-trainable params: 0
_________________________________________________________________
```



```
Epoch 9/10
1250/1250 [==============================] - 34s 28ms/step - loss: 0.5736 - accuracy: 0.8007 - val_loss: 0.89
96 - val_accuracy: 0.7030
Epoch 10/10
1250/1250 [==============================] - 33s 26ms/step - loss: 0.5317 - accuracy: 0.8129 - val_loss: 0.92
04 - val_accuracy: 0.7084
```

```
313/313 - 1s - loss: 0.9204 - accuracy: 0.7084
valid_accuracy=0.7084000110626221, valid_loss=0.9203763604164124
```



```
array([[9.16872596e-05, 1.21282181e-04, 4.29793909e-05, ...,
         4.83334179e-06, 9.58305970e-03, 4.54883062e-04],
        [3.27145593e-04, 8.70263088e-04, 2.60794697e-09, ...,
         1.16433096e-11, 9.98798370e-01, 3.48283857e-06],
        [7.60395303e-02, 4.12312537e-01, 1.73454185e-03, ...,
         2.79331085e-04, 4.02229100e-01, 9.72837359e-02],
        ...,
        [2.97923270e-06, 7.16236059e-07, 2.15551630e-03, ...,
         4.46977885e-03, 4.82129217e-05, 2.20777883e-06],
        [8.53645429e-03, 1.11738004e-01, 5.14256628e-03, ...,
         1.33331283e-04, 1.52999455e-05, 5.30263933e-04],
        [7.29238181e-08, 2.06557368e-10, 4.68017703e-07, ...,
         9.99579251e-01, 7.88085038e-13, 6.82521417e-09]], dtype=float32)
```

Conclusions: From the above model I tried playing with different parameters, I've updated the number of filters, size of the filter, activation layers added and removed the conclusion layers in the initial code. As I added the different activation form it impacted the model in different ways. E.g. using relu accuracy reduced in the training where using tanh accuracy increased. When I removed the layer my model became too fit so overfitting issues were there. Changing the number of filters, worked best for making it an optimized model. So that was also my best model with increased filters in the hidden layer.

Reflections:

First of all this was fun homework specially making it open for class and making a leadership board it makes it more interesting. It was challenging with the right amount of toughness and more importantly by changing different parameters we practically understand how it affects the model and how it helps to make an efficient model.