Submitted By

- Ruthvic Punyamurtula (Problems - 3,5)
- Sai Charan Kothapalli (problems - 1,2,4)

Resources

- youtube demo - 1,2,4
- youtube demo - 3,5
- Source Code

Introduction

This is lab assignment 3 of CS5590 - python/Deep Learning class. This lab specifically consists of several deep learning tasks based on regression, CNN and LSTM.

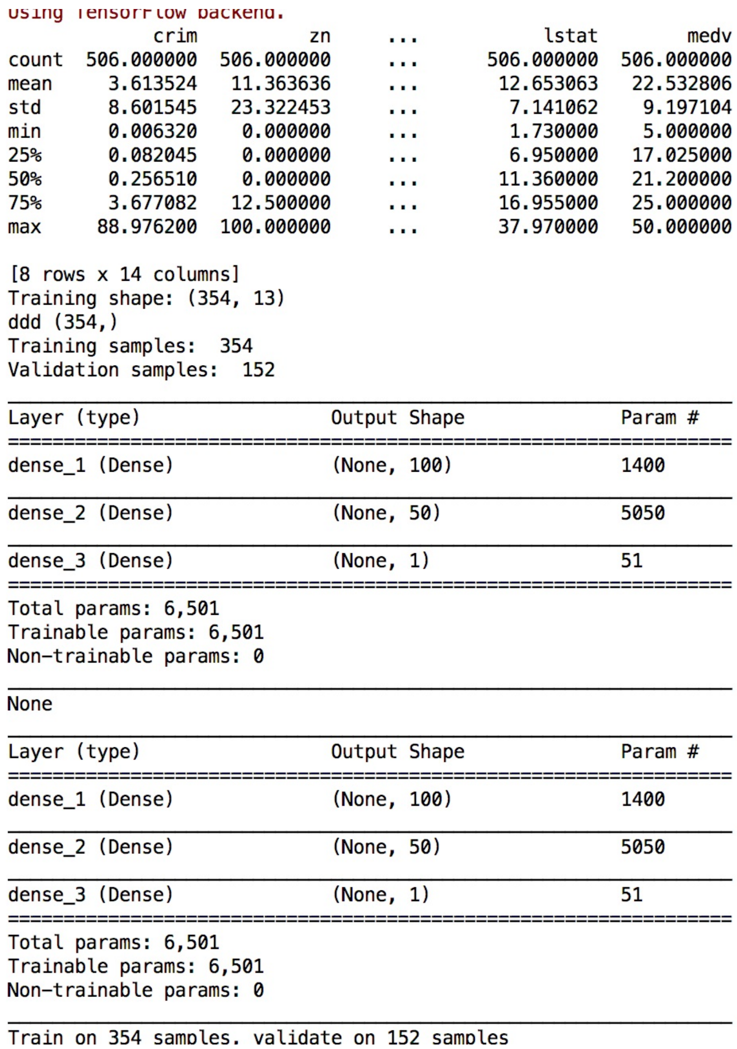
Objective

In this assignment, we used Kaggle datasets & implemented

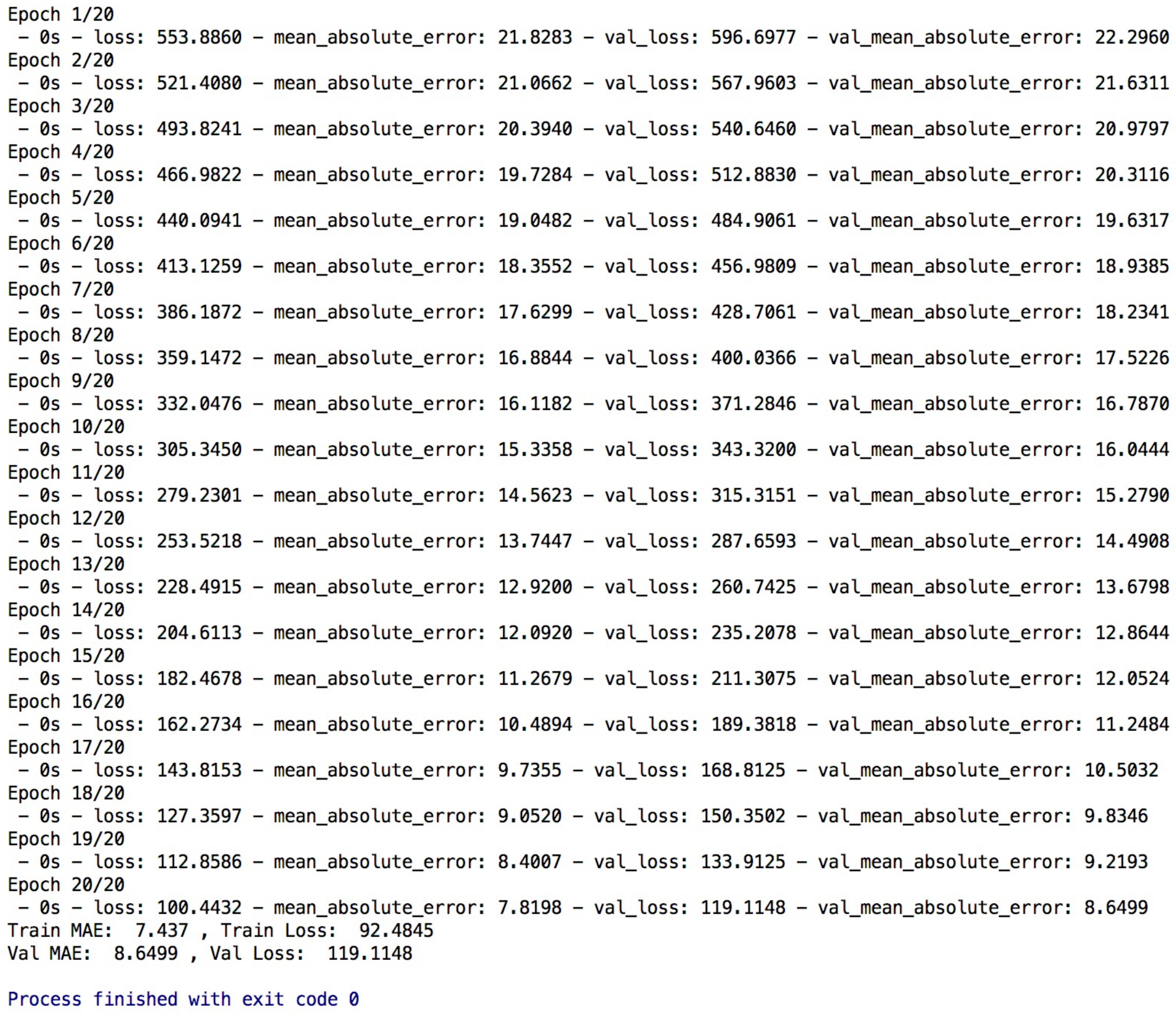
- Linear Regression
- Logistics Regression
- Image Classification with CNN
- Text Classification with CNN
- Text classification with LSTM
- Compare Text Classification with CNN & LSTM

Approaches/Methods

1.Linear Regression

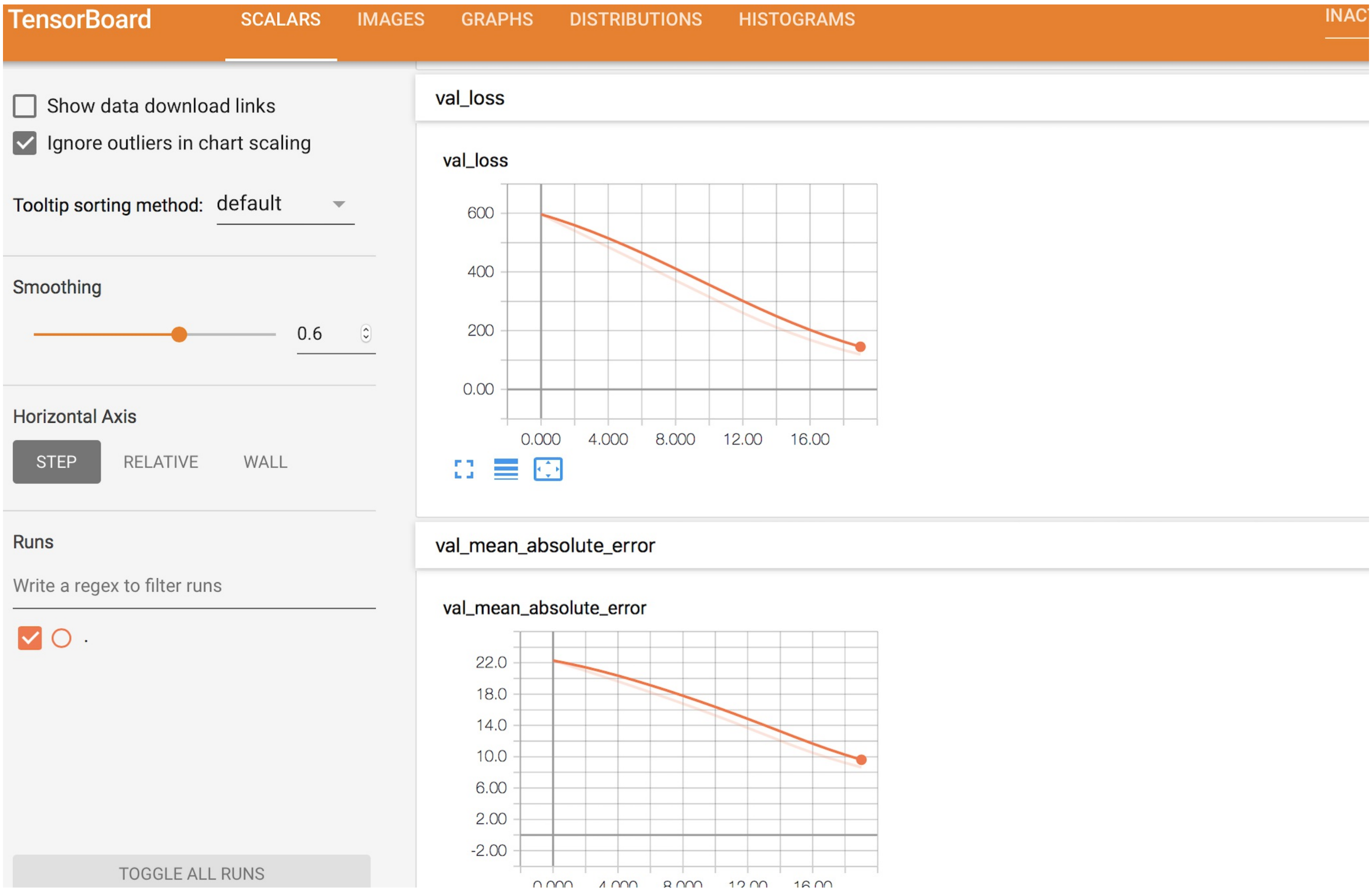


```
Using TensorFlow backend.
              crim          zn     ...         lstat        medv
count  506.000000  506.000000     ...    506.000000  506.000000
mean     3.613524   11.363636     ...     12.653063   22.532806
std      8.601545   23.322453     ...      7.141062    9.197104
min      0.006320    0.000000     ...      1.730000    5.000000
25%      0.082045    0.000000     ...      6.950000   17.025000
50%      0.256510    0.000000     ...     11.360000   21.200000
75%      3.677082   12.500000     ...     16.955000   25.000000
max     88.976200  100.000000     ...     37.970000   50.000000

[8 rows x 14 columns]
Training shape: (354, 13)
ddd (354,)
Training samples:  354
Validation samples:  152
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 100)               1400
_________________________________________________________________
dense_2 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 51
=================================================================
Total params: 6,501
Trainable params: 6,501
Non-trainable params: 0
_________________________________________________________________
None
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 100)               1400
_________________________________________________________________
dense_2 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 51
=================================================================
Total params: 6,501
Trainable params: 6,501
Non-trainable params: 0
_________________________________________________________________
Train on 354 samples, validate on 152 samples
```



```
Epoch 1/20
 - 0s - loss: 553.8860 - mean_absolute_error: 21.8283 - val_loss: 596.6977 - val_mean_absolute_error: 22.2960
Epoch 2/20
 - 0s - loss: 521.4080 - mean_absolute_error: 21.0662 - val_loss: 567.9603 - val_mean_absolute_error: 21.6311
Epoch 3/20
 - 0s - loss: 493.8241 - mean_absolute_error: 20.3940 - val_loss: 540.6460 - val_mean_absolute_error: 20.9797
Epoch 4/20
 - 0s - loss: 466.9822 - mean_absolute_error: 19.7284 - val_loss: 512.8830 - val_mean_absolute_error: 20.3116
Epoch 5/20
 - 0s - loss: 440.0941 - mean_absolute_error: 19.0482 - val_loss: 484.9061 - val_mean_absolute_error: 19.6317
Epoch 6/20
 - 0s - loss: 413.1259 - mean_absolute_error: 18.3552 - val_loss: 456.9809 - val_mean_absolute_error: 18.9385
Epoch 7/20
 - 0s - loss: 386.1872 - mean_absolute_error: 17.6299 - val_loss: 428.7061 - val_mean_absolute_error: 18.2341
Epoch 8/20
 - 0s - loss: 359.1472 - mean_absolute_error: 16.8844 - val_loss: 400.0366 - val_mean_absolute_error: 17.5226
Epoch 9/20
 - 0s - loss: 332.0476 - mean_absolute_error: 16.1182 - val_loss: 371.2846 - val_mean_absolute_error: 16.7870
Epoch 10/20
 - 0s - loss: 305.3450 - mean_absolute_error: 15.3358 - val_loss: 343.3200 - val_mean_absolute_error: 16.0444
Epoch 11/20
 - 0s - loss: 279.2301 - mean_absolute_error: 14.5623 - val_loss: 315.3151 - val_mean_absolute_error: 15.2790
Epoch 12/20
 - 0s - loss: 253.5218 - mean_absolute_error: 13.7447 - val_loss: 287.6593 - val_mean_absolute_error: 14.4908
Epoch 13/20
 - 0s - loss: 228.4915 - mean_absolute_error: 12.9200 - val_loss: 260.7425 - val_mean_absolute_error: 13.6798
Epoch 14/20
 - 0s - loss: 204.6113 - mean_absolute_error: 12.0920 - val_loss: 235.2078 - val_mean_absolute_error: 12.8644
Epoch 15/20
 - 0s - loss: 182.4678 - mean_absolute_error: 11.2679 - val_loss: 211.3075 - val_mean_absolute_error: 12.0524
Epoch 16/20
 - 0s - loss: 162.2734 - mean_absolute_error: 10.4894 - val_loss: 189.3818 - val_mean_absolute_error: 11.2484
Epoch 17/20
 - 0s - loss: 143.8153 - mean_absolute_error: 9.7355 - val_loss: 168.8125 - val_mean_absolute_error: 10.5032
Epoch 18/20
 - 0s - loss: 127.3597 - mean_absolute_error: 9.0520 - val_loss: 150.3502 - val_mean_absolute_error: 9.8346
Epoch 19/20
 - 0s - loss: 112.8586 - mean_absolute_error: 8.4007 - val_loss: 133.9125 - val_mean_absolute_error: 9.2193
Epoch 20/20
 - 0s - loss: 100.4432 - mean_absolute_error: 7.8198 - val_loss: 119.1148 - val_mean_absolute_error: 8.6499
Train MAE:  7.437 , Train Loss:  92.4845
Val MAE:  8.6499 , Val Loss:  119.1148

Process finished with exit code 0
```

Tensorboard



☐ Show data download links

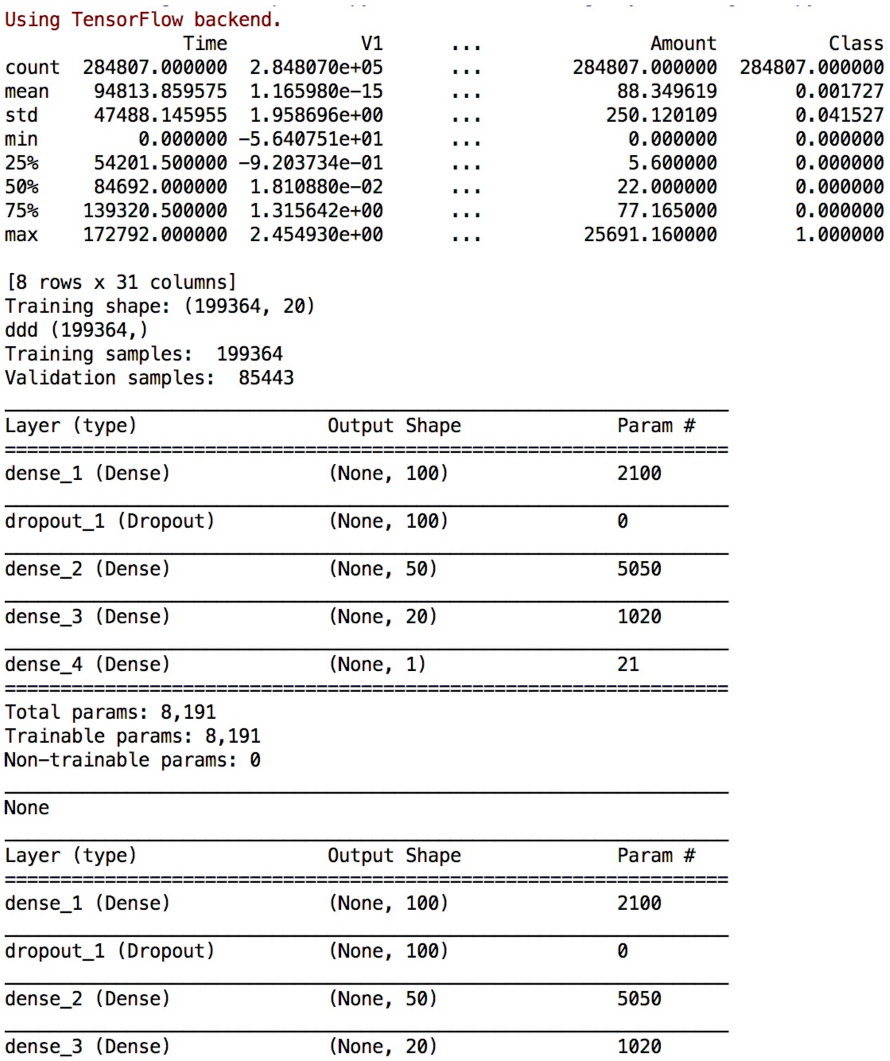
☑ Ignore outliers in chart scaling

Tooltip sorting method: default

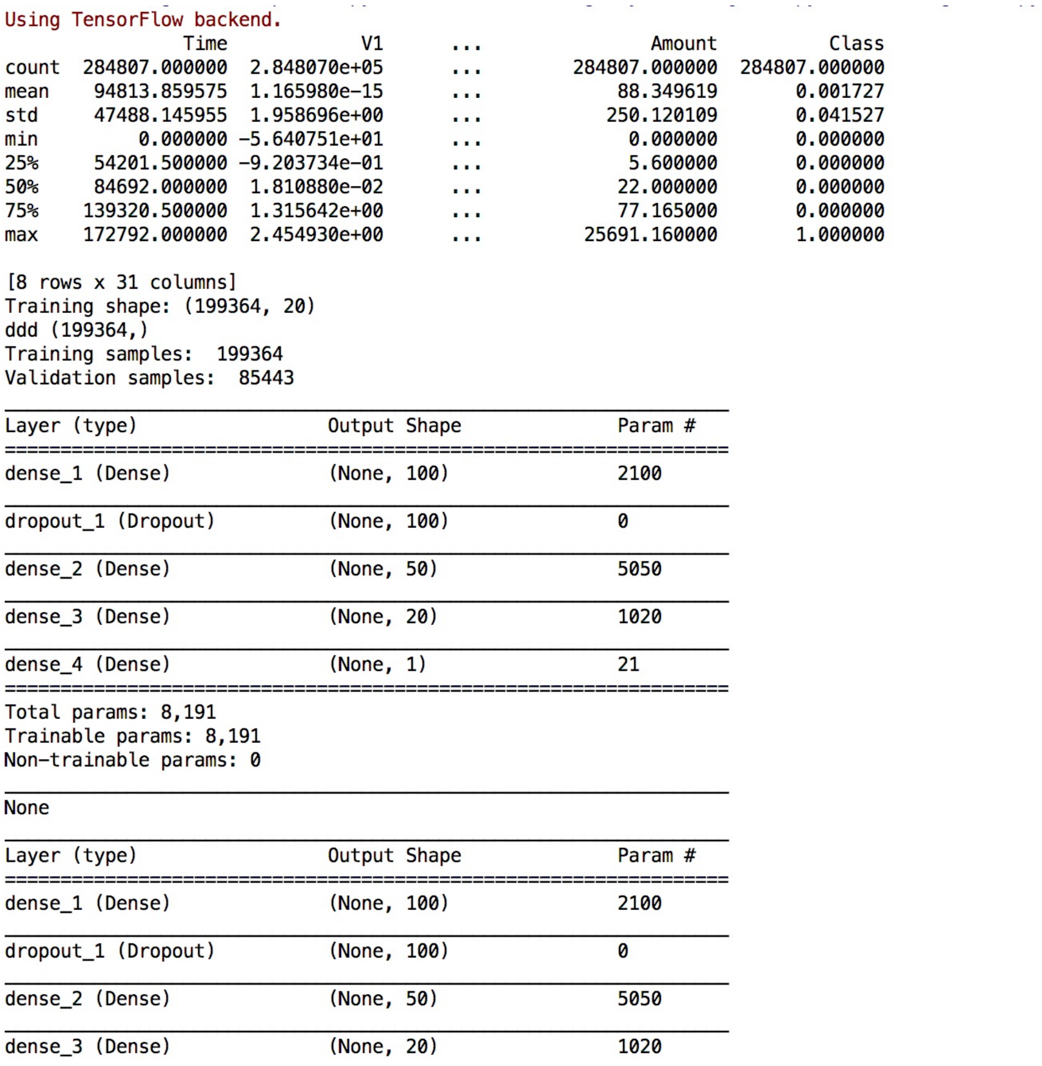
Smoothing 0.6

Horizontal Axis: STEP

2. Logistic Regression

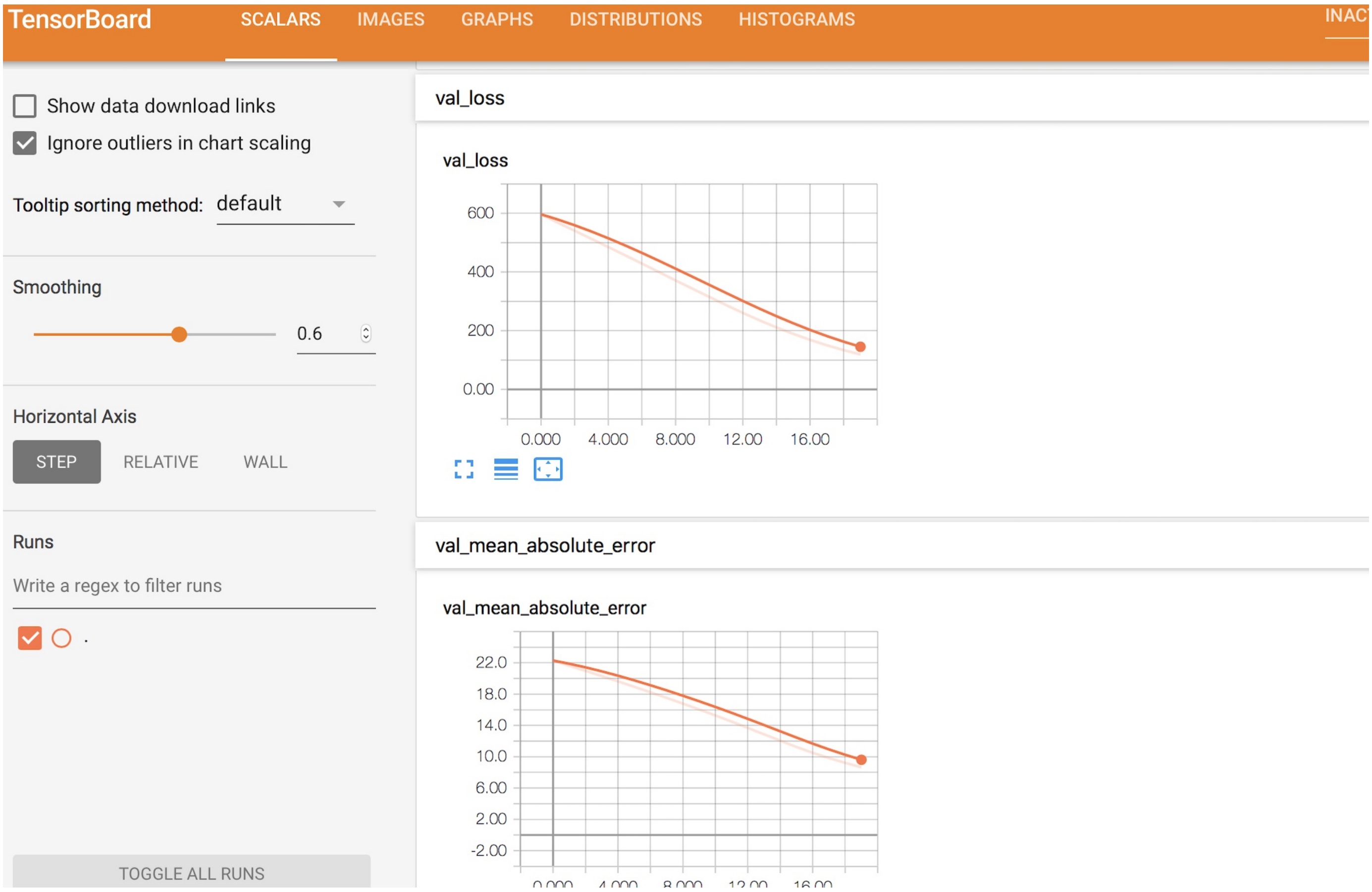


```
Using TensorFlow backend.
                Time            V1      ...            Amount          Class
count  284807.000000  2.848070e+05      ...     284807.000000  284807.000000
mean    94813.859575  1.165980e-15      ...         88.349619       0.001727
std     47488.145955  1.958696e+00      ...        250.120109       0.041527
min         0.000000 -5.640751e+01      ...          0.000000       0.000000
25%     54201.500000 -9.203734e-01      ...          5.600000       0.000000
50%     84692.000000  1.810880e-02      ...         22.000000       0.000000
75%    139320.500000  1.315642e+00      ...         77.165000       0.000000
max    172792.000000  2.454930e+00      ...      25691.160000       1.000000

[8 rows x 31 columns]
Training shape: (199364, 20)
ddd (199364,)
Training samples:  199364
Validation samples:  85443
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 100)               2100
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_2 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_3 (Dense)              (None, 20)                1020
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 21
=================================================================
Total params: 8,191
Trainable params: 8,191
Non-trainable params: 0
_________________________________________________________________
None
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 100)               2100
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_2 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_3 (Dense)              (None, 20)                1020
```



```
Using TensorFlow backend.
                Time            V1      ...           Amount          Class
count  284807.000000  2.848070e+05      ...    284807.000000  284807.000000
mean    94813.859575  1.165980e-15      ...        88.349619       0.001727
std     47488.145955  1.958696e+00      ...       250.120109       0.041527
min         0.000000 -5.640751e+01      ...         0.000000       0.000000
25%     54201.500000 -9.203734e-01      ...         5.600000       0.000000
50%     84692.000000  1.810880e-02      ...        22.000000       0.000000
75%    139320.500000  1.315642e+00      ...        77.165000       0.000000
max    172792.000000  2.454930e+00      ...     25691.160000       1.000000

[8 rows x 31 columns]
Training shape: (199364, 20)
ddd (199364,)
Training samples:  199364
Validation samples:  85443
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_1 (Dense)              (None, 100)               2100      
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 50)                5050      
_________________________________________________________________
dense_3 (Dense)              (None, 20)                1020      
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 21        
=================================================================
Total params: 8,191
Trainable params: 8,191
Non-trainable params: 0
_________________________________________________________________
None
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_1 (Dense)              (None, 100)               2100      
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 50)                5050      
_________________________________________________________________
dense_3 (Dense)              (None, 20)                1020      
```

Tensorboard



☐ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis STEP

Runs ☑ ○ .

val_loss

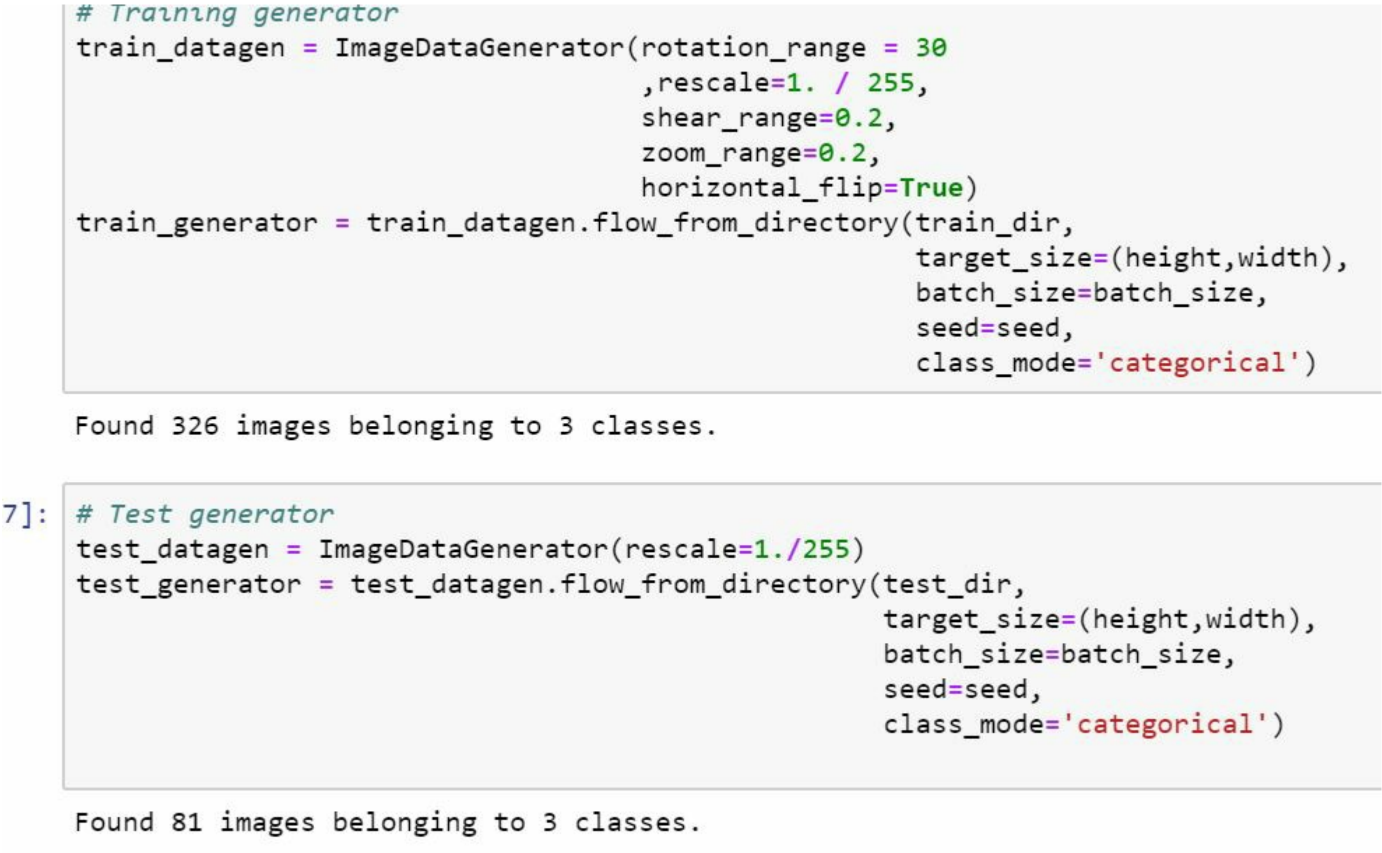
val_mean_absolute_error

3. Image Classification with CNN

The dataset used is - https://www.kaggle.com/slothkong/10-monkey-species

The initial data had 10 classes/species of monkeys. Due to laptop performance issues, we have reduced the dataset to 3 species of monkeys and loaded the data into /content of google colab - with runtime as GPU.

The code snippet describing the data size is given below



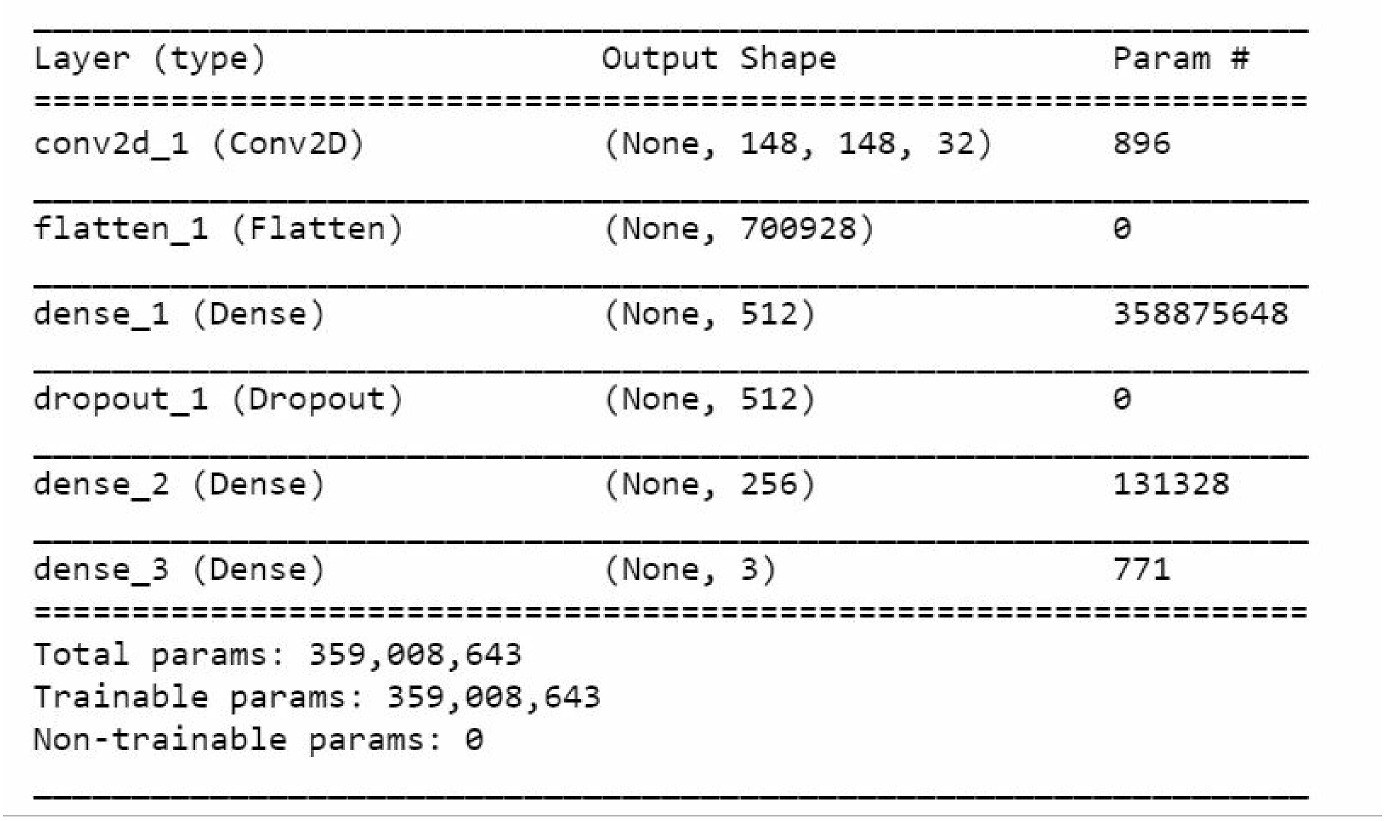
```
# Training generator
train_datagen = ImageDataGenerator(rotation_range = 30
                                   ,rescale=1. / 255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
train_generator = train_datagen.flow_from_directory(train_dir,
                                                    target_size=(height,width),
                                                    batch_size=batch_size,
                                                    seed=seed,
                                                    class_mode='categorical')
```

```
Found 326 images belonging to 3 classes.
```

```
7]: # Test generator
    test_datagen = ImageDataGenerator(rescale=1./255)
    test_generator = test_datagen.flow_from_directory(test_dir,
                                                      target_size=(height,width),
                                                      batch_size=batch_size,
                                                      seed=seed,
                                                      class_mode='categorical')
```

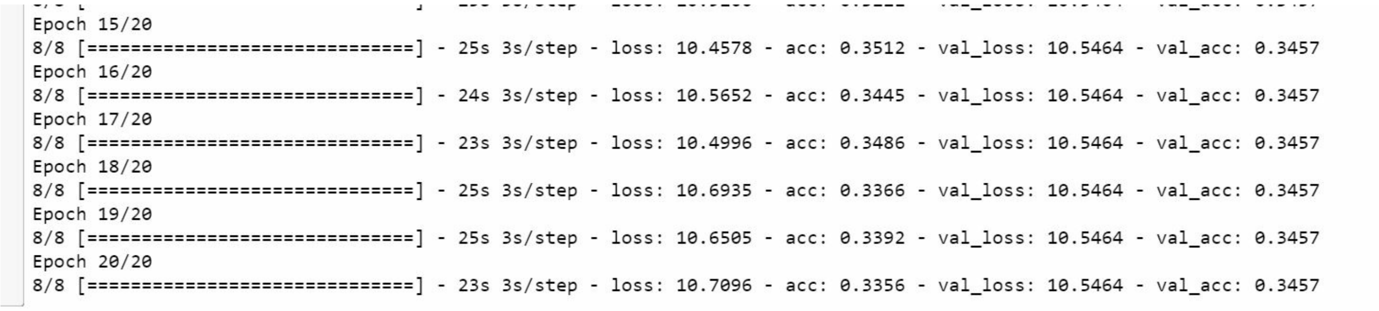
```
Found 81 images belonging to 3 classes.
```

The model used for classification is



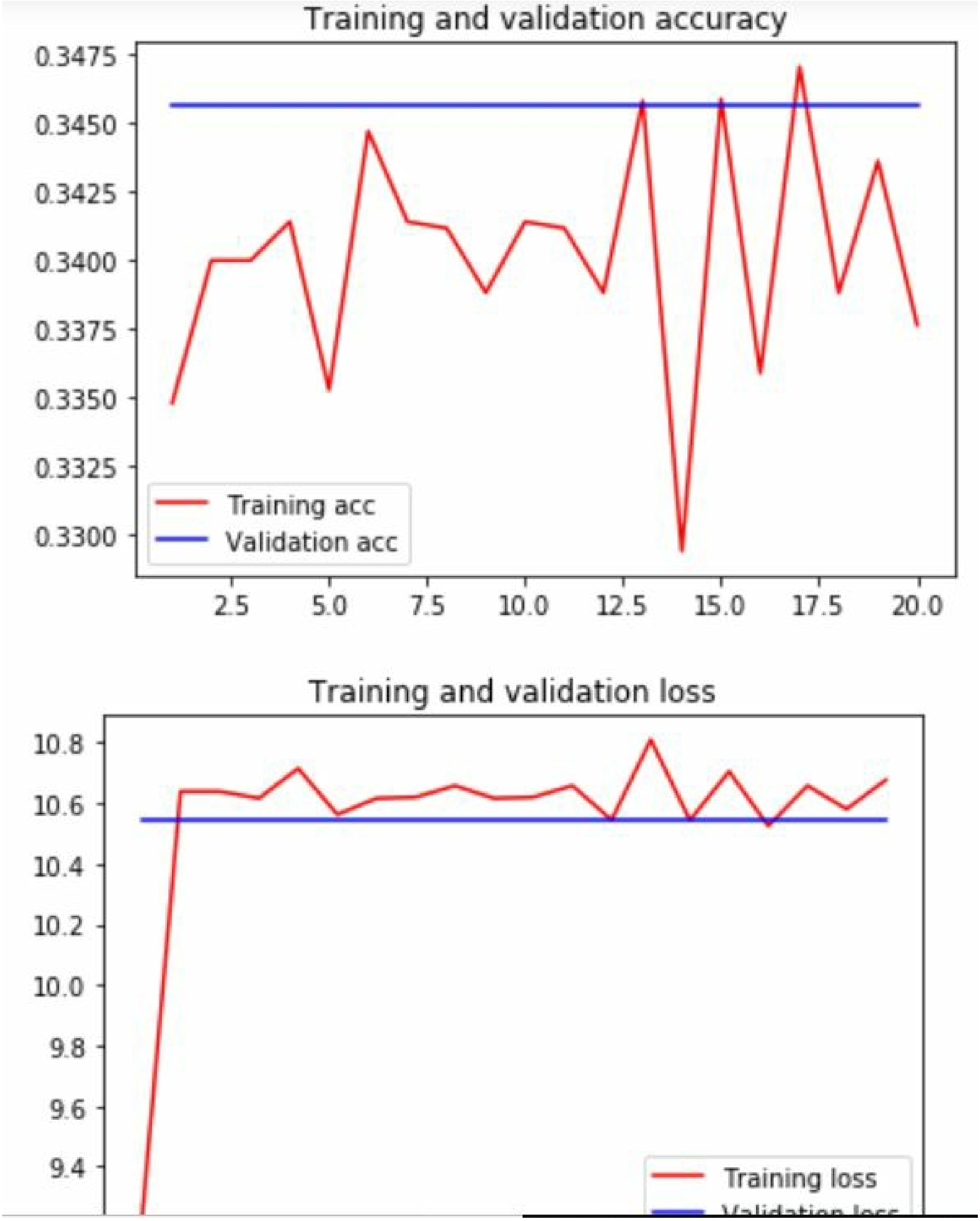
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 148, 148, 32)      896
_________________________________________________________________
flatten_1 (Flatten)          (None, 700928)            0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               358875648
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 256)               131328
_________________________________________________________________
dense_3 (Dense)              (None, 3)                 771
=================================================================
Total params: 359,008,643
Trainable params: 359,008,643
Non-trainable params: 0
_________________________________________________________________
```

Because we had small data & training on 20 epochs the accuracy obtained is low (about 35%)

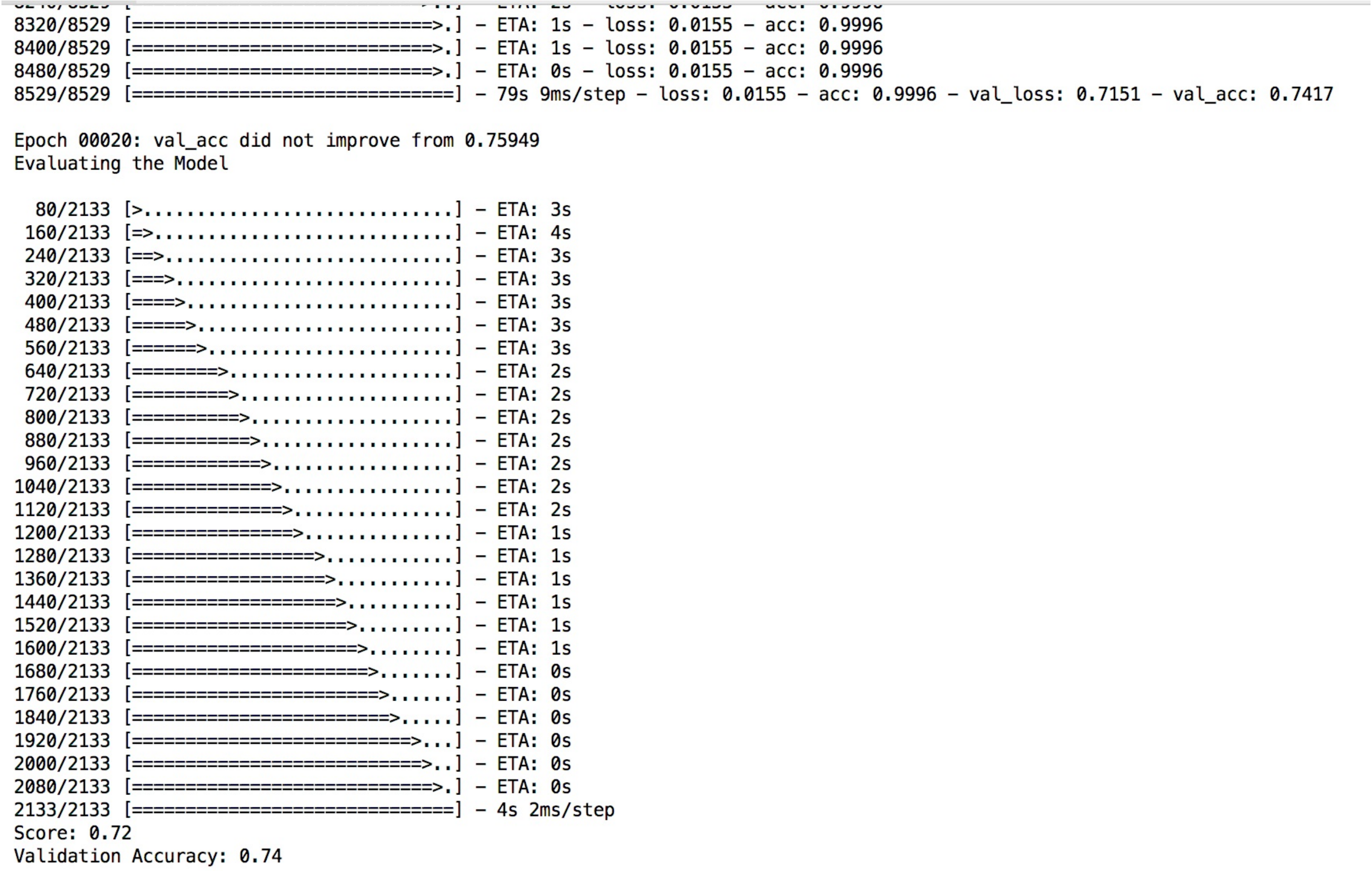


```
Epoch 15/20
8/8 [==============================] - 25s 3s/step - loss: 10.4578 - acc: 0.3512 - val_loss: 10.5464 - val_acc: 0.3457
Epoch 16/20
8/8 [==============================] - 24s 3s/step - loss: 10.5652 - acc: 0.3445 - val_loss: 10.5464 - val_acc: 0.3457
Epoch 17/20
8/8 [==============================] - 23s 3s/step - loss: 10.4996 - acc: 0.3486 - val_loss: 10.5464 - val_acc: 0.3457
Epoch 18/20
8/8 [==============================] - 25s 3s/step - loss: 10.6935 - acc: 0.3366 - val_loss: 10.5464 - val_acc: 0.3457
Epoch 19/20
8/8 [==============================] - 25s 3s/step - loss: 10.6505 - acc: 0.3392 - val_loss: 10.5464 - val_acc: 0.3457
Epoch 20/20
8/8 [==============================] - 23s 3s/step - loss: 10.7096 - acc: 0.3356 - val_loss: 10.5464 - val_acc: 0.3457
```

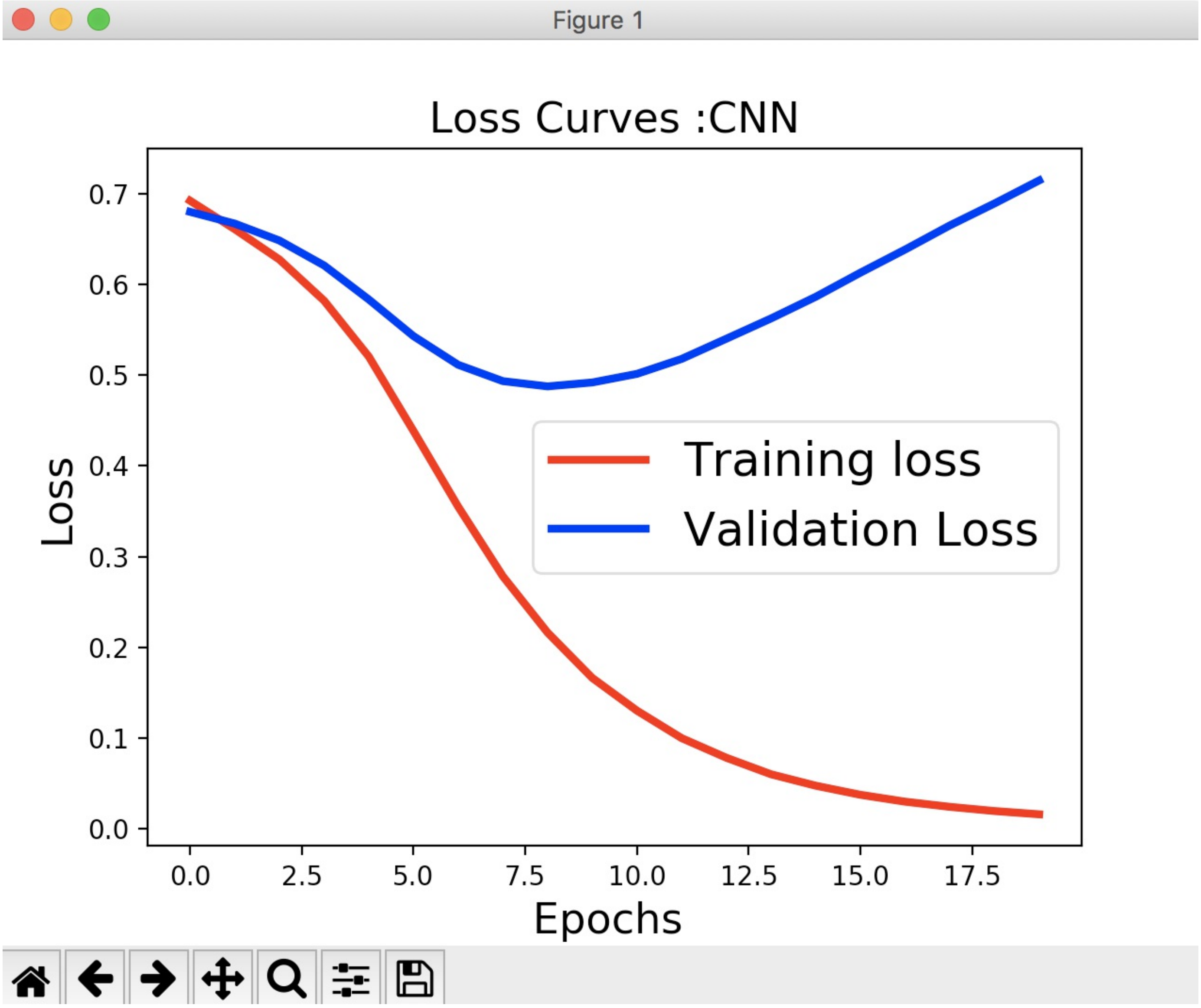
The plots showing accuracy & loss for train & validation data is shown below



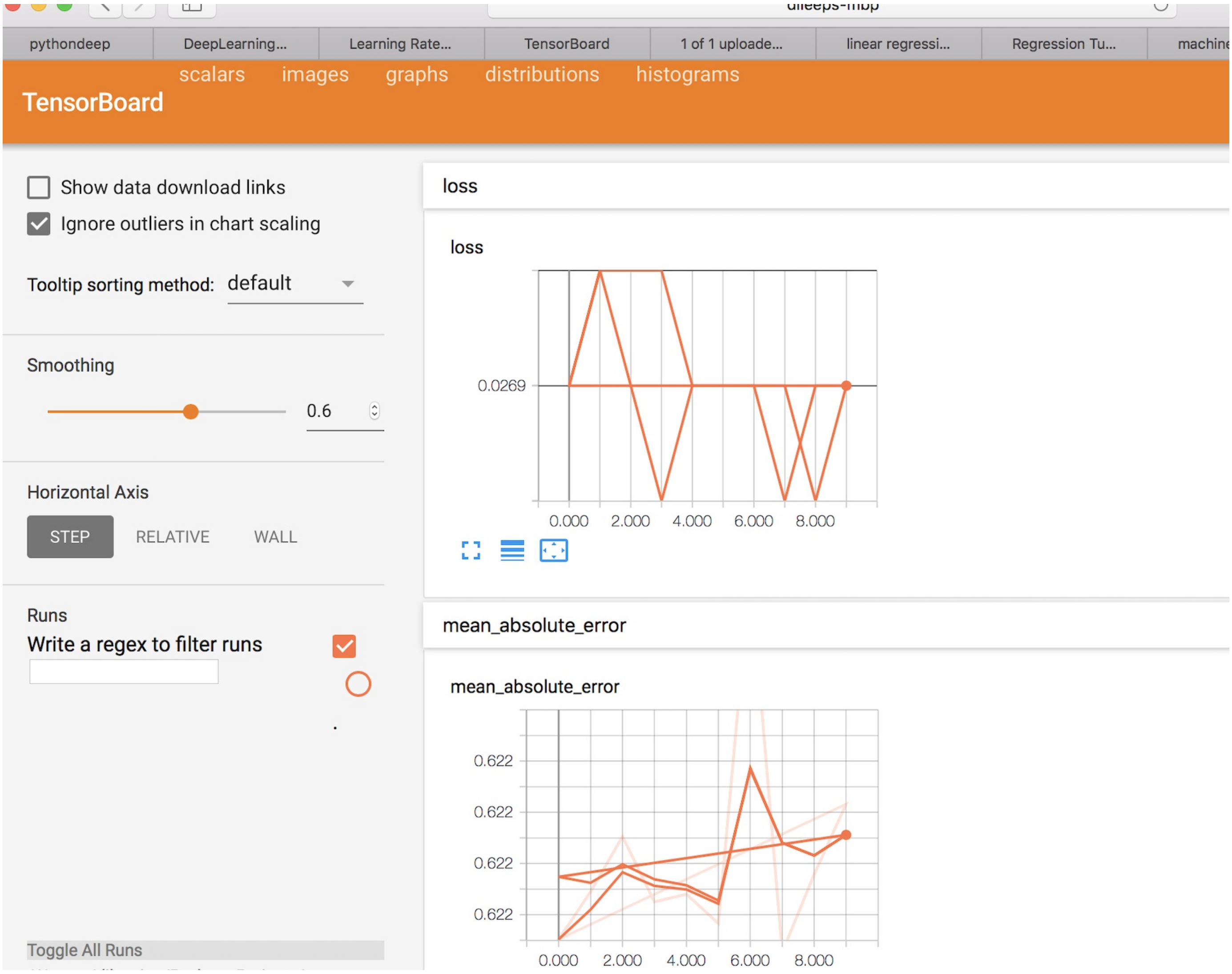
4. Text Classification with CNN



```
8320/8529 [============================>.] - ETA: 1s - loss: 0.0155 - acc: 0.9996
8400/8529 [============================>.] - ETA: 1s - loss: 0.0155 - acc: 0.9996
8480/8529 [============================>.] - ETA: 0s - loss: 0.0155 - acc: 0.9996
8529/8529 [==============================] - 79s 9ms/step - loss: 0.0155 - acc: 0.9996 - val_loss: 0.7151 - val_acc: 0.7417

Epoch 00020: val_acc did not improve from 0.75949
Evaluating the Model

  80/2133 [>.............................] - ETA: 3s
 160/2133 [=>............................] - ETA: 4s
 240/2133 [==>...........................] - ETA: 3s
 320/2133 [===>..........................] - ETA: 3s
 400/2133 [====>.........................] - ETA: 3s
 480/2133 [=====>........................] - ETA: 3s
 560/2133 [======>.......................] - ETA: 3s
 640/2133 [========>.....................] - ETA: 2s
 720/2133 [=========>....................] - ETA: 2s
 800/2133 [==========>...................] - ETA: 2s
 880/2133 [===========>..................] - ETA: 2s
 960/2133 [============>.................] - ETA: 2s
1040/2133 [=============>................] - ETA: 2s
1120/2133 [==============>...............] - ETA: 2s
1200/2133 [===============>..............] - ETA: 1s
1280/2133 [=================>............] - ETA: 1s
1360/2133 [==================>...........] - ETA: 1s
1440/2133 [===================>..........] - ETA: 1s
1520/2133 [====================>.........] - ETA: 1s
1600/2133 [=====================>........] - ETA: 1s
1680/2133 [======================>.......] - ETA: 0s
1760/2133 [=======================>......] - ETA: 0s
1840/2133 [========================>.....] - ETA: 0s
1920/2133 [==========================>...] - ETA: 0s
2000/2133 [===========================>..] - ETA: 0s
2080/2133 [============================>.] - ETA: 0s
2133/2133 [==============================] - 4s 2ms/step
Score: 0.72
Validation Accuracy: 0.74
```



Tensorboard



☐ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0.6

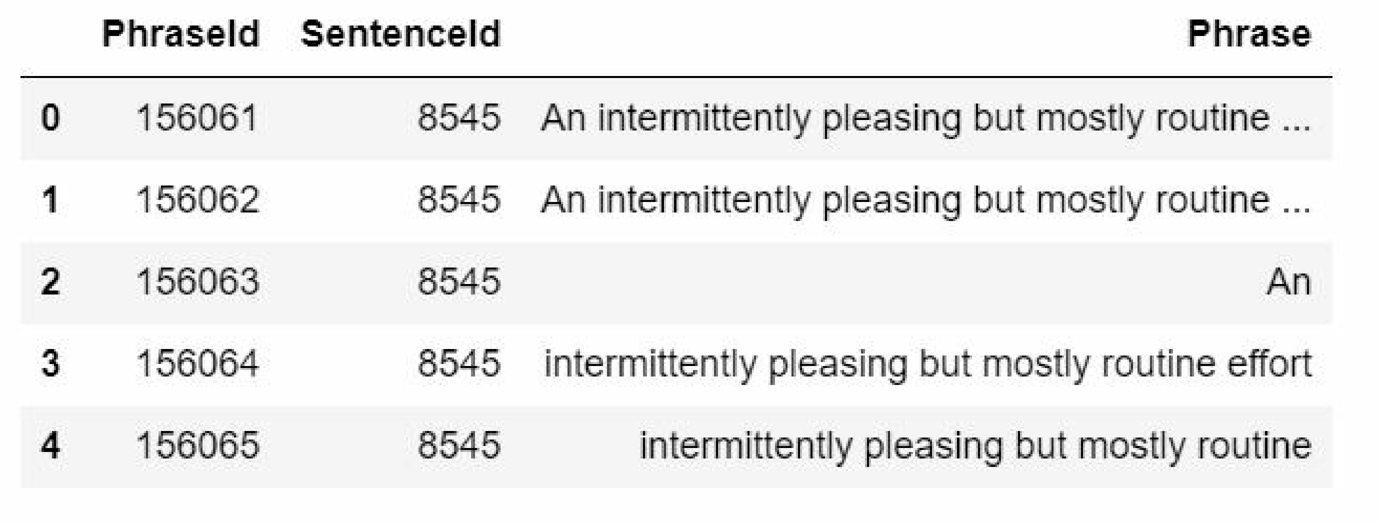
Horizontal Axis: STEP

5. Text Classification with LSTM

The dataset used is - https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews/data

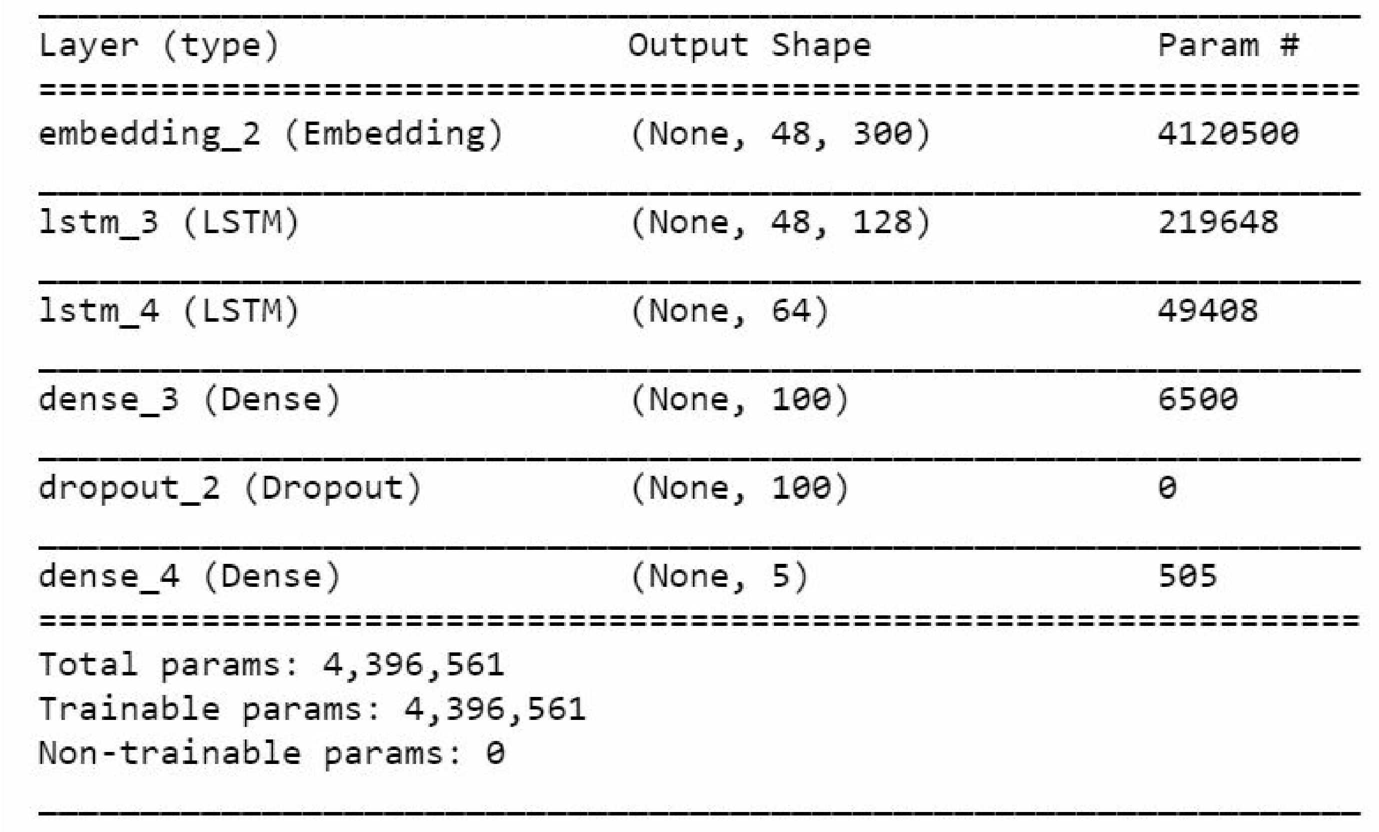
As it is a text data, we used the original dataset and uploaded the content to google colab with runtime as GPU.

The code snippet describing the data is given below



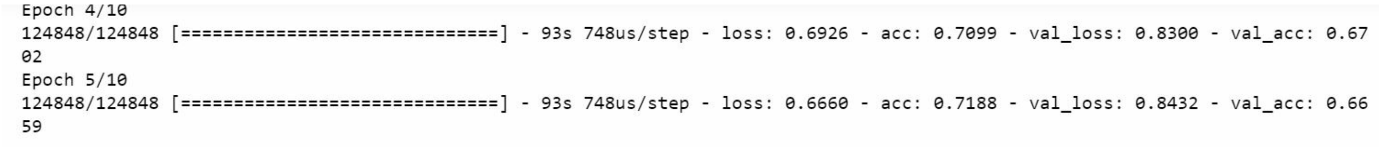
	PhraseId	SentenceId	Phrase
0	156061	8545	An intermittently pleasing but mostly routine ...
1	156062	8545	An intermittently pleasing but mostly routine ...
2	156063	8545	An
3	156064	8545	intermittently pleasing but mostly routine effort
4	156065	8545	intermittently pleasing but mostly routine

The LSTM model used for Text classification is



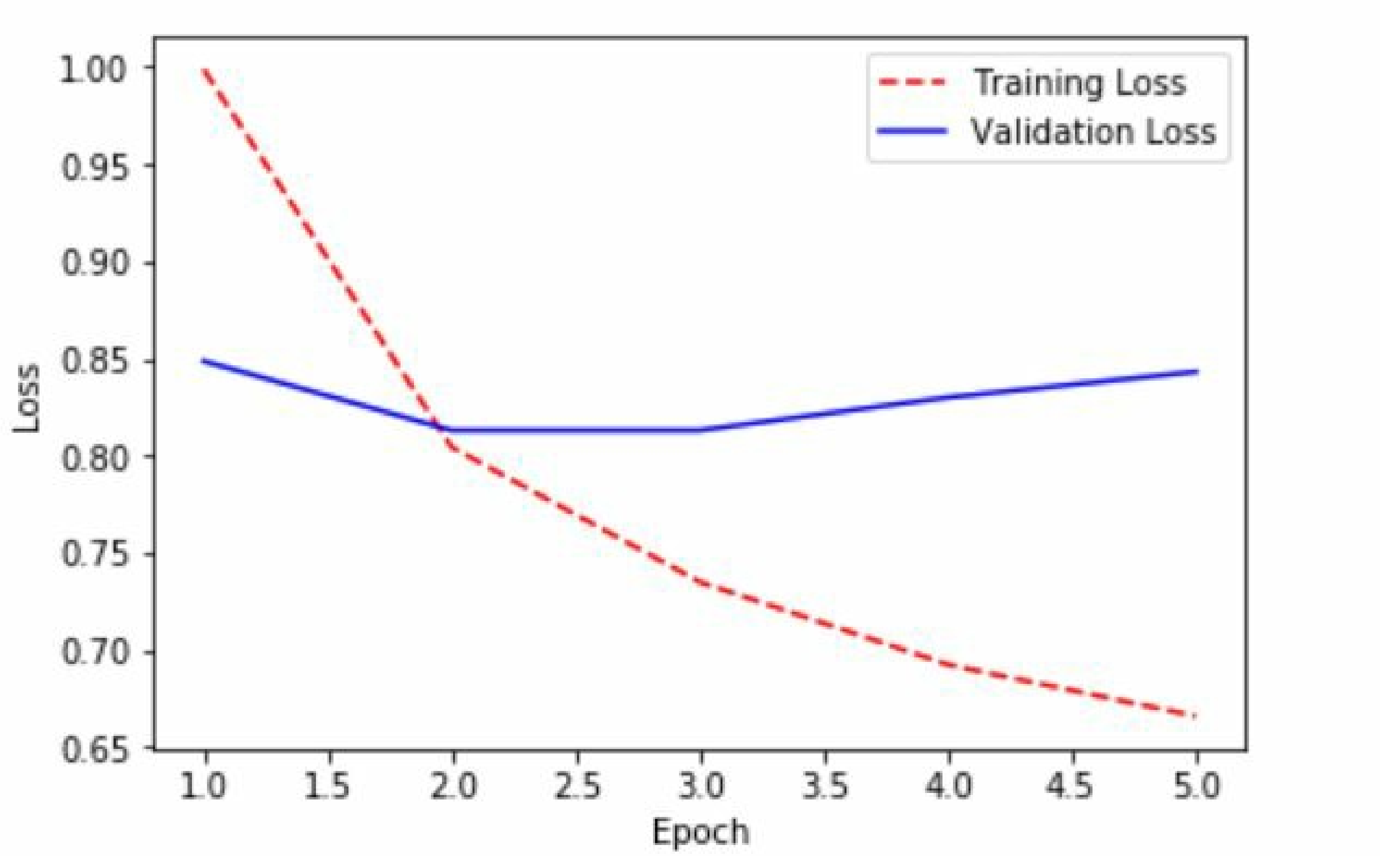
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_2 (Embedding)      (None, 48, 300)           4120500   
_________________________________________________________________
lstm_3 (LSTM)                (None, 48, 128)           219648    
_________________________________________________________________
lstm_4 (LSTM)                (None, 64)                49408     
_________________________________________________________________
dense_3 (Dense)              (None, 100)               6500      
_________________________________________________________________
dropout_2 (Dropout)          (None, 100)               0         
_________________________________________________________________
dense_4 (Dense)              (None, 5)                 505       
=================================================================
Total params: 4,396,561
Trainable params: 4,396,561
Non-trainable params: 0
_________________________________________________________________
```

We trained the model for 5 epochs, with a batch size of 256 for faster training and got an accuracy of 66%



```
Epoch 4/10
124848/124848 [==============================] - 93s 748us/step - loss: 0.6926 - acc: 0.7099 - val_loss: 0.8300 - val_acc: 0.67
02
Epoch 5/10
124848/124848 [==============================] - 93s 748us/step - loss: 0.6660 - acc: 0.7188 - val_loss: 0.8432 - val_acc: 0.66
59
```

The plots showing accuracy for train & validation data is shown below



6. Comparisons on CNN and LSTM models

We can clearly see that by comparing the accuracy and the tensorboard results

CNN text classification has obtained an accuracy of 71.47%

CNN image classification has obtained an accuracy of 31.34%

LSTM has obtained a validation accuracy of 61.29%

CNN text classification though is run on 5, 10, 15 epochs has produced different learning rates with a better accuracy for each epoch with an improving accuracy each time it is feed more data.

LSTM is a slow learner for which it gives varying accuracies for each time the text is given as input, where it learns for each continuous text data feed.

We can conclude by these accuracies that CNN outperforms LSTM for text classification.