**Homework-3**

CSCE 633: Machine Learning

Manav Gurumoorthy

830000011

Introduction:

In this HW, we looked at the use of Artificial Neural Networks to perform image recognition.  
The dataset consists of labelled images of humans depicting different facial expression. The recgonition task was to accurately discern the facial expression being depicted in the image.

The dataset was consisted of several 48X48 grayscale images. i.e. each image was considered as a vector of length 2304, with each element representing the intensity of that pixel.

We look at two main approaches here

1. The use of a Feed Forward Network
2. Use of a Convolutional Neural Network.

While these are the two broad approaches in this study, each method had their own nuances and required careful manipulation of hyperparameters.

Data Exploration:

In this part, we explored the data that we were going to model. The first step was to plot the images from the data that we had. This was done using inbuilt functions in python.

(one such example) more in the code section.

We also looked at how many exhibits from each of the labels were present in the training dataset.

Angry: 3995

Disgust: 436

Fear: 4097

Happy: 7215

Sad: 4830

Surprise: 3171

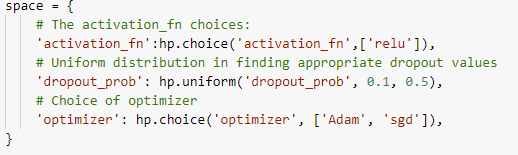
Neutral: 4965

\_\_\_\_\_\_\_

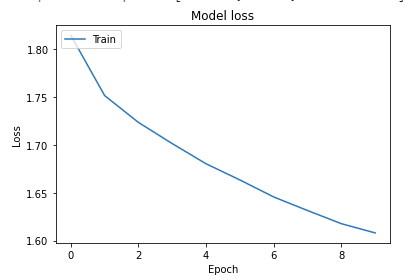
Total: 28709

Feedforward Neural Network:

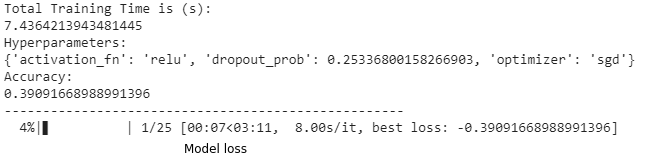
In this section we construct a FNN using the keras library and test the performance of the model with the validation data by playing around with hyperparameter values. I liked the hyperopt library, so I used this to find optimal hyperparameter values. My search space consisted of the following parameters:



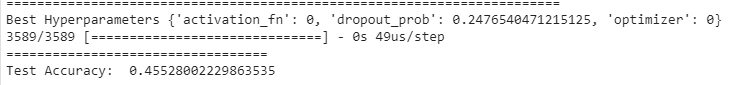
We also plotted the loss as a function of the number of iterations (epochs?). One such example is show below. The rest are shown as a part of the code section.



Some stats that are printed with every change in hyperparameters: (This was done using Prafulla’s code)



The best hyperparams were then used and the model was tested on the test set.



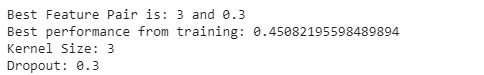
In an earlier test I also tried 3 different activation functions namely ReLu, Sigmoid and tanh, but since the other 2 gave me extremely poor results I defaulted to just using ReLu.

Convolution Neural Networks:

Like the FNN case, we create a CNN model using the keras library APIs. Similar to the FNN case, we try to optimise the model using the validation set and different combinations of hyperparameters to optimise the model.

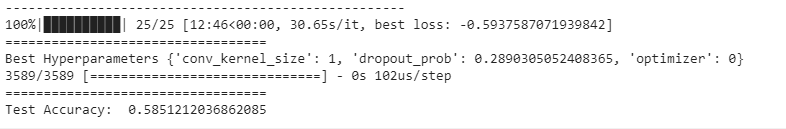
Firstly I used a simple for loop which used the following params and trained 10 different models, the best hyperparameter pair from this was used on the test set.





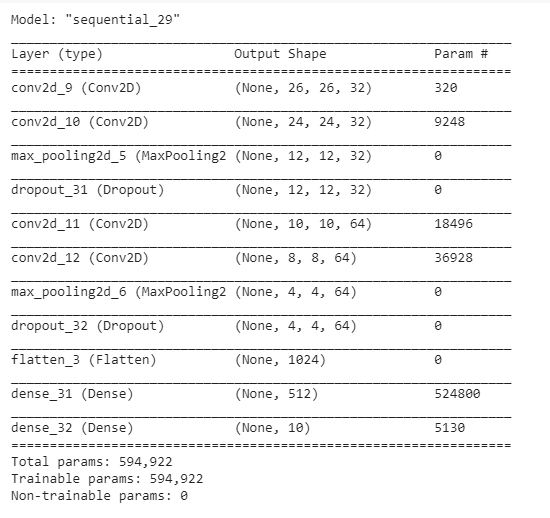


Unfortunately, this was not yielding very good results because I was only using a small set of values. So once again I used HyperOpt to find a better set of hyperparameters.

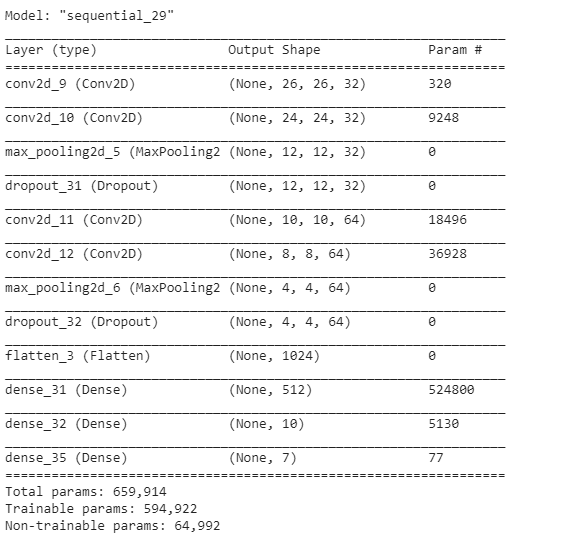


Fine Tuning:

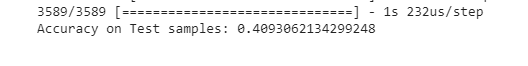
I was a little unclear about this part so I have done 2 things.   
The first thing is to train a model on the MNIST dataset and modify the FER dataset suitably to work with the trained model. Doing this got an accuracy of 40%

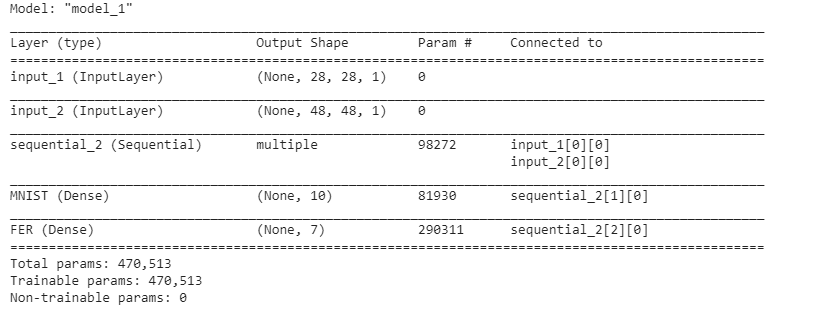


Then adding another layer for the outputs of FER,



The final model accuracy:



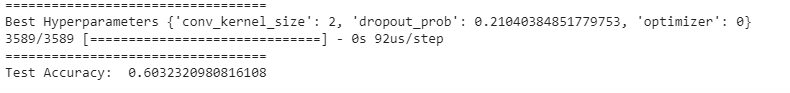
The second thing was to create a combined model of FER and MNIST. 

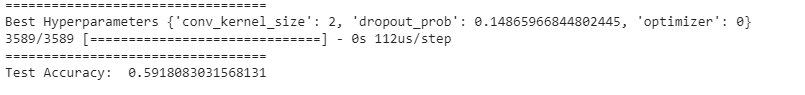
This yielded the following results,

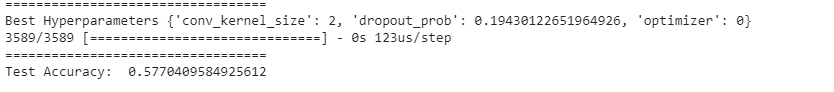


Data Augmentation:

In this section we look at different types of data augmentation using the ImageDataGenerator from keras. I tried Standardisation, ZCA Whitening and Random Rotation, the results are shown below in the same order.







Feature Design:

In this section I tried implementing HOG on the images because it seemed like it would give good results for the task we had intended. Unfortunately, I guess either there is an error in my implementation or HOG is not a method that is suitable for this sort of task. But I attribute it to the former rather than the latter.



APPENDIX A: COLAB Workbook