# ASSIGNMENT 1

## CIFAR-10 IMAGE CLASSIFICATION USING KERAS

### Problem 1:

Using Keras, build a MLP to classify the CIFAR-10 dataset. Note that each record is of size 1*3072. Starting with the MNIST example code, build a MLP to classify the data into the 10 classes.

Model 1:

Used Keras to build a MLP model for Image classification on CIFAR-10 dataset. Each record is of size 1*3072(shape = 32,32,3). MLP image classifier classifies the data into the 10 classes.

Model 2:

Used Keras to build a Convolutional Neural Network for Image classification on CIFAR-10 dataset. Each record is of size 1*3072(shape = 32,32,3). CNN image classifier classifies the data into the 10 classes.

DATASET:

CIFAR-10 dataset contains 60000 color images of size 32x32 in 10 classes, with 6000 images per class. The size of the training dataset is 50000 and 10000 test images.

The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.

http://www.cs.utoronto.ca/~kriz/cifar.html

Modify the following parameters and discuss the effect of changing parameters on loss and Accuracy:
The table below shows the parameters of the best models.

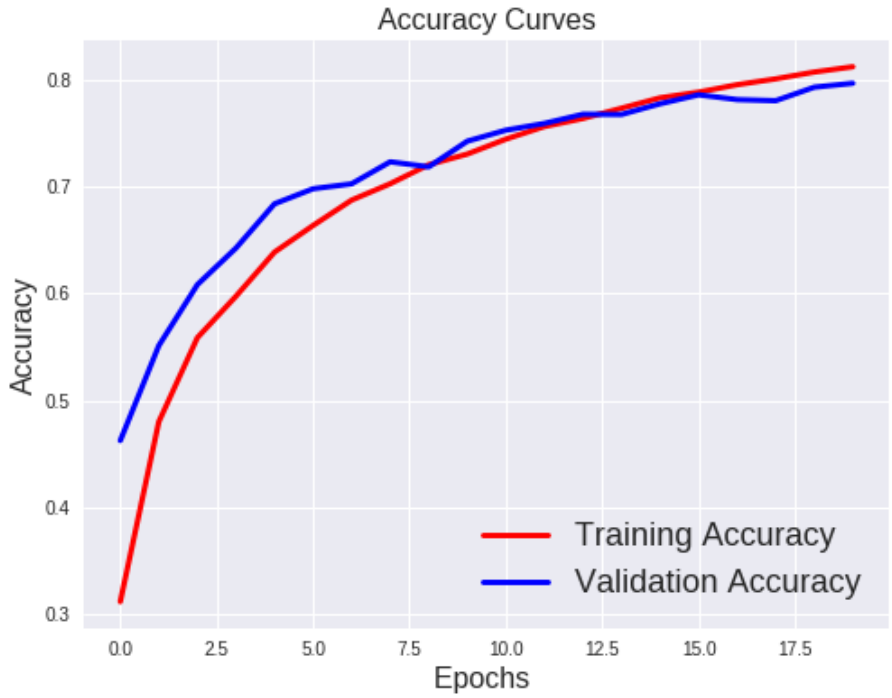
| | No of epochs | Batch size | Number of neurons | Number of layers | Learning rate | Activation functions | Dropout rates |
|---|---|---|---|---|---|---|---|
| MLP | 20 | 256 | 3786 | 4 | 0.0003 | Relu + Softmax | 0.25,0.4, 0.5 |
| CNN | 20 | 256 | - | 10 | 0.002 | Relu + Softmax | 0.25, 0.25,0.3 |

## Model Fitting

```
Epoch 1/20
50000/50000 [==============================] - 16s 317us/step - loss:
1.8315 - acc: 0.3118 - val_loss: 1.4680 - val_acc: 0.4625
Epoch 2/20
50000/50000 [==============================] - 14s 270us/step - loss:
1.4189 - acc: 0.4801 - val_loss: 1.2693 - val_acc: 0.5514
Epoch 3/20
50000/50000 [==============================] - 14s 271us/step - loss:
1.2399 - acc: 0.5588 - val_loss: 1.1206 - val_acc: 0.6084
Epoch 4/20
50000/50000 [==============================] - 13s 269us/step - loss:
1.1268 - acc: 0.5975 - val_loss: 1.0056 - val_acc: 0.6426
Epoch 5/20
50000/50000 [==============================] - 14s 272us/step - loss:
1.0283 - acc: 0.6388 - val_loss: 0.9145 - val_acc: 0.6840
Epoch 6/20
50000/50000 [==============================] - 14s 271us/step - loss:
0.9556 - acc: 0.6637 - val_loss: 0.8693 - val_acc: 0.6981
Epoch 7/20
50000/50000 [==============================] - 13s 270us/step - loss:
0.8975 - acc: 0.6875 - val_loss: 0.8597 - val_acc: 0.7027
Epoch 8/20
50000/50000 [==============================] - 14s 271us/step - loss:
0.8464 - acc: 0.7028 - val_loss: 0.7984 - val_acc: 0.7233
Epoch 9/20
50000/50000 [==============================] - 14s 271us/step - loss:
0.8034 - acc: 0.7206 - val_loss: 0.8054 - val_acc: 0.7186
Epoch 10/20
50000/50000 [==============================] - 14s 271us/step - loss:
0.7688 - acc: 0.7305 - val_loss: 0.7393 - val_acc: 0.7426
Epoch 11/20
50000/50000 [==============================] - 14s 271us/step - loss:
0.7364 - acc: 0.7445 - val_loss: 0.7141 - val_acc: 0.7529
Epoch 12/20
50000/50000 [==============================] - 14s 271us/step - loss:
0.6979 - acc: 0.7563 - val_loss: 0.6903 - val_acc: 0.7592
Epoch 13/20
50000/50000 [==============================] - 13s 269us/step - loss:
0.6771 - acc: 0.7638 - val_loss: 0.6732 - val_acc: 0.7678
Epoch 14/20
50000/50000 [==============================] - 14s 271us/step - loss:
0.6505 - acc: 0.7735 - val_loss: 0.6761 - val_acc: 0.7676
Epoch 15/20
50000/50000 [==============================] - 14s 272us/step - loss:
0.6219 - acc: 0.7832 - val_loss: 0.6472 - val_acc: 0.7776
Epoch 16/20
50000/50000 [==============================] - 14s 270us/step - loss:
0.6033 - acc: 0.7885 - val_loss: 0.6297 - val_acc: 0.7860
Epoch 17/20
50000/50000 [==============================] - 14s 271us/step - loss:
0.5849 - acc: 0.7955 - val_loss: 0.6398 - val_acc: 0.7815
Epoch 18/20
```

```
50000/50000 [==============================] - 14s 272us/step - loss:
0.5678 - acc: 0.8009 - val_loss: 0.6418 - val_acc: 0.7805
Epoch 19/20
50000/50000 [==============================] - 13s 270us/step - loss:
0.5500 - acc: 0.8073 - val_loss: 0.5961 - val_acc: 0.7931
Epoch 20/20
50000/50000 [==============================] - 14s 275us/step - loss:
0.5351 - acc: 0.8122 - val_loss: 0.5944 - val_acc: 0.7968
```

Graphical representation of Accuracy:




Graphical Representation of Accuracy:


## Results:

1) Provide a recommendation for the best model you would recommend for classification. Which model (with parameter values) would you choose and why?

A->The best model I would recommend for classification is the convolutional neural network. The accuracy after validation is ~78% and evaluation accuracy are 79%

The best parameters that helped improve the model are:

epochs = 20

batch_size = 256

learning rate = 0.002

optimizer = adam

metrics = accuracy

activation function – relu + softmax

dropout rate – 0.25/.25/.3

• Comment on how good your model is? Does it overfit/underfit data? What could you do to improve the model?

A-> All the MLP models could not produce accurate results. All the models were overfitting, giving an average 45% accuracy. The model starts overfitting just after the $5^{th}$ epoch. However,

when the image classifier was built using the convolutional NN the accuracy improved and loss decreased significantly. After training the model for 20 epochs the validation score is ~78% and ~79% for evaluation. The model started overfitting after 30 epochs.

I think the model could improve and produce more accurate results if data augmentation is performed. Due to the less amount of data, the model could not improve accuracy and reduce the loss. Also model starts to remember the data after a point which reduces the validation score because it performs poorly on the out of sample data.