Multilayer Perceptron Using the CIFAR-10 Dataset

Syed Haroon Perveez

1. Introduction:

A multilayer perceptron is the basic kind of a deep neural network an here we have altered various parameters of the CIFAR-10 dataset to come up with best model by providing different observations and provided further suggestions on how the accuracy of the model can be increased.

Observations:

Here I have noted down 3 observations that I observed as a result of various permutations and combinations done on the parameters of the base code of the MLP.

Trial Number	Train Accuracy(%)	Train Loss	Validation Accuracy(%)	Validation Loss
1	56.46	1.232	53.50	1.3191
2	54.72	1.2490	51.84	1.3649
3	54.49	1.2583	51.35	1.3918

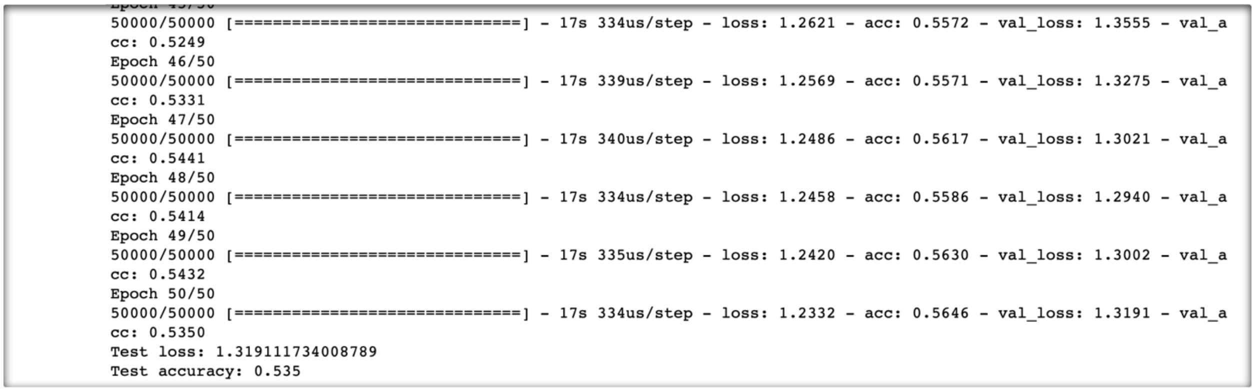
Observation 1:

Number of epochs: 50
Batch size: 128
Network Configuration

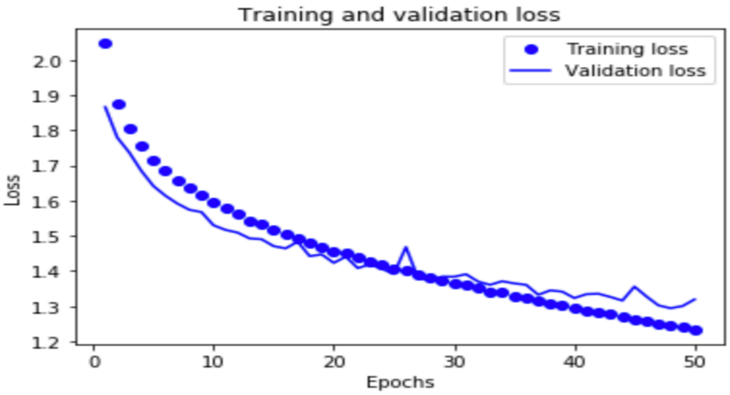
a. Number of neurons in layer: Layer 1 – 1024, Layer 2 - 512
b. Number of layers: 2

Activation function: ReLU function
Dropout rates: 0.3

Here we notice the model is over-fitting as there is too much of validation loss. As the training loss decreases the validation loss increases with the number of epochs.



```
Epoch 45/50
50000/50000 [==============================] - 17s 334us/step - loss: 1.2621 - acc: 0.5572 - val_loss: 1.3555 - val_a
cc: 0.5249
Epoch 46/50
50000/50000 [==============================] - 17s 339us/step - loss: 1.2569 - acc: 0.5571 - val_loss: 1.3275 - val_a
cc: 0.5331
Epoch 47/50
50000/50000 [==============================] - 17s 340us/step - loss: 1.2486 - acc: 0.5617 - val_loss: 1.3021 - val_a
cc: 0.5441
Epoch 48/50
50000/50000 [==============================] - 17s 334us/step - loss: 1.2458 - acc: 0.5586 - val_loss: 1.2940 - val_a
cc: 0.5414
Epoch 49/50
50000/50000 [==============================] - 17s 335us/step - loss: 1.2420 - acc: 0.5630 - val_loss: 1.3002 - val_a
cc: 0.5432
Epoch 50/50
50000/50000 [==============================] - 17s 334us/step - loss: 1.2332 - acc: 0.5646 - val_loss: 1.3191 - val_a
cc: 0.5350
Test loss: 1.319111734008789
Test accuracy: 0.535
```



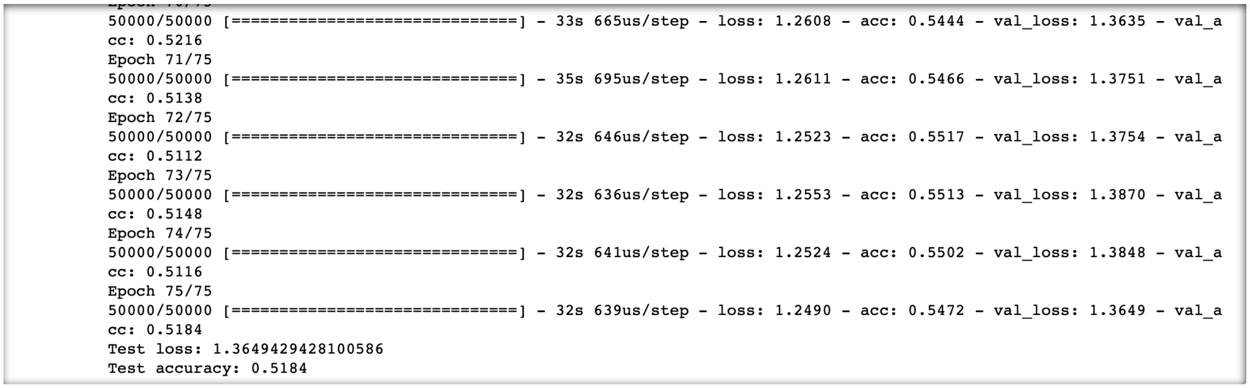
Observation 2:

Number of epochs: 75
Batch size: 128
Network Configuration

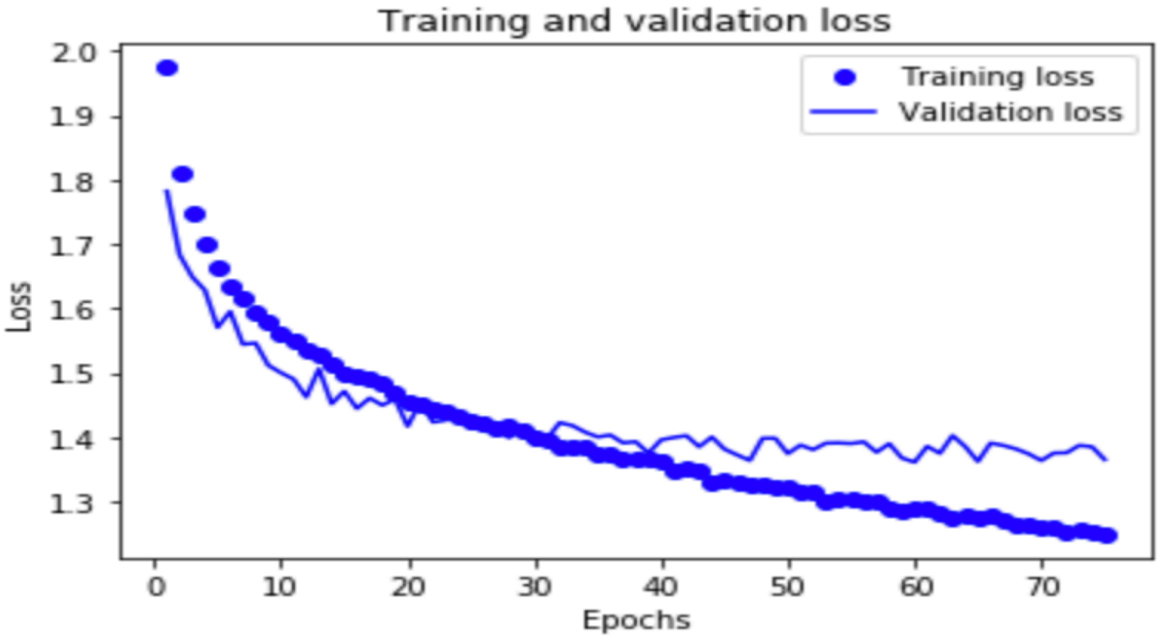
a. Number of neurons in layer: Layer 1 – 1024, Layer 2 –512, Layer 3 - 512
b. Number of layers: 3

Activation function: ReLU function
Dropout rates: 0.2

Here notice that the accuracy increases but with still there is overfitting



```
50000/50000 [==============================] - 33s 665us/step - loss: 1.2608 - acc: 0.5444 - val_loss: 1.3635 - val_a
cc: 0.5216
Epoch 71/75
50000/50000 [==============================] - 35s 695us/step - loss: 1.2611 - acc: 0.5466 - val_loss: 1.3751 - val_a
cc: 0.5138
Epoch 72/75
50000/50000 [==============================] - 32s 646us/step - loss: 1.2523 - acc: 0.5517 - val_loss: 1.3754 - val_a
cc: 0.5112
Epoch 73/75
50000/50000 [==============================] - 32s 636us/step - loss: 1.2553 - acc: 0.5513 - val_loss: 1.3870 - val_a
cc: 0.5148
Epoch 74/75
50000/50000 [==============================] - 32s 641us/step - loss: 1.2524 - acc: 0.5502 - val_loss: 1.3848 - val_a
cc: 0.5116
Epoch 75/75
50000/50000 [==============================] - 32s 639us/step - loss: 1.2490 - acc: 0.5472 - val_loss: 1.3649 - val_a
cc: 0.5184
Test loss: 1.3649429428100586
Test accuracy: 0.5184
```



Observation 3:

Number of epochs: 30

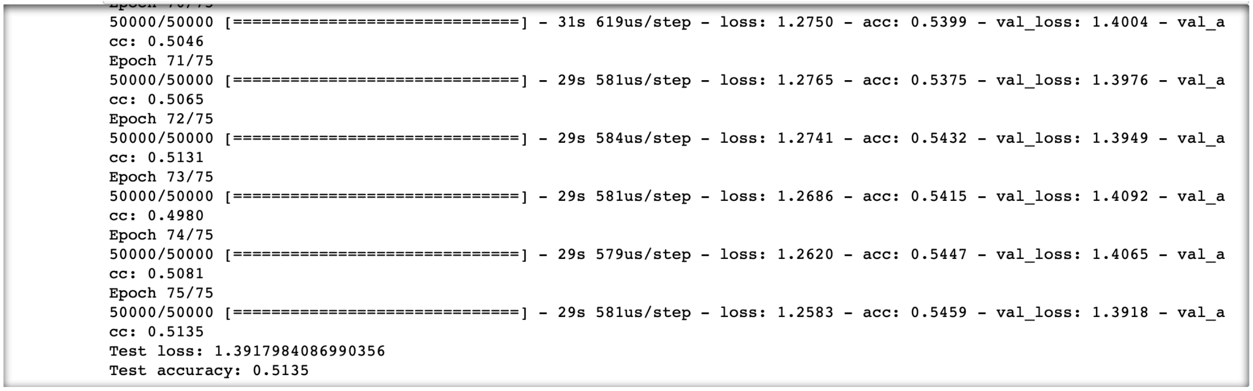
Batch size: 512

Network Configuration

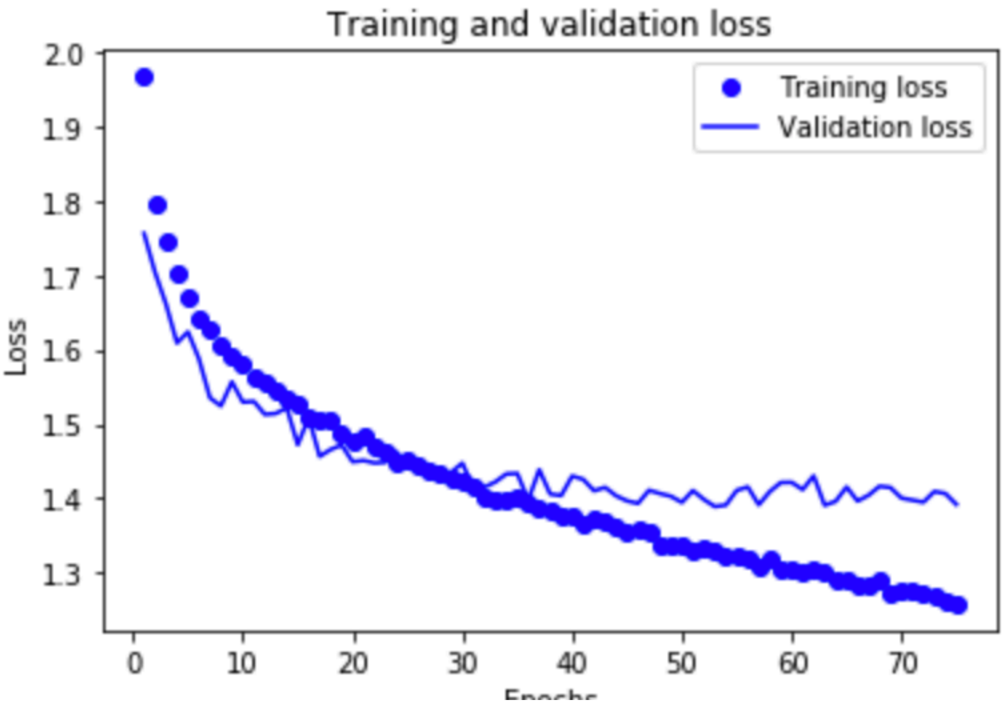
a. Number of neurons in layer: Layer 1 – 1024, Layer 2 –1024, Layer 3 – 512
b. Number of layers: 3

Activation function: ReLU function

Dropout rates: 0.2 first two layers and 0.1 for last layer



```
50000/50000 [==============================] - 31s 619us/step - loss: 1.2750 - acc: 0.5399 - val_loss: 1.4004 - val_a
cc: 0.5046
Epoch 71/75
50000/50000 [==============================] - 29s 581us/step - loss: 1.2765 - acc: 0.5375 - val_loss: 1.3976 - val_a
cc: 0.5065
Epoch 72/75
50000/50000 [==============================] - 29s 584us/step - loss: 1.2741 - acc: 0.5432 - val_loss: 1.3949 - val_a
cc: 0.5131
Epoch 73/75
50000/50000 [==============================] - 29s 581us/step - loss: 1.2686 - acc: 0.5415 - val_loss: 1.4092 - val_a
cc: 0.4980
Epoch 74/75
50000/50000 [==============================] - 29s 579us/step - loss: 1.2620 - acc: 0.5447 - val_loss: 1.4065 - val_a
cc: 0.5081
Epoch 75/75
50000/50000 [==============================] - 29s 581us/step - loss: 1.2583 - acc: 0.5459 - val_loss: 1.3918 - val_a
cc: 0.5135
Test loss: 1.3917984086990356
Test accuracy: 0.5135
```



Here in this observation we can see three characteristics. Along the graph the validation is less than the training loss, after a few epochs it becomes equal to the training loss and after another few epochs, we observe over fitting in the model.

Conclusion:

After trying different combinations of parameters, I came up with the conclusion that increasing the density of the hidden layers, that is, increasing the number of neurons in the hidden layers increases the accuracy of the training data but it starts to overfit after a particular number of epochs. (approx. 45 epochs)

The best Model according to me is 2nd one which has training accuracy of 54.7(cutoff at 40 epochs) and validation accuracy at 52.9