COMPARISON OF THE PERFORMANCE OF TWO MULTI-LAYER PERCEPTRONS IN MNIST DATASET

Submitted by,

Dheeraj B
S5 CSE
Roll no: 27

SOURCE CODE

The Source code used to train the model can be found here (Github repository link)

I trained the MNIST dataset in two models. The model summary of the two models is as shown in the images below

Model 1

```
Model: "sequential_10"

 Layer (type)                Output Shape              Param #
=================================================================
 flatten_10 (Flatten)        (None, 784)               0

 dense_22 (Dense)            (None, 50)                39250

 dense_23 (Dense)            (None, 10)                510


=================================================================
Total params: 39,760
Trainable params: 39,760
Non-trainable params: 0
```

Model 2

```
Model: "sequential_11"

 Layer (type)                Output Shape              Param #
=================================================================
 flatten_11 (Flatten)        (None, 784)               0

 dense_24 (Dense)            (None, 100)               78500

 dense_25 (Dense)            (None, 100)               10100

 dense_26 (Dense)            (None, 10)                1010


=================================================================
Total params: 89,610
Trainable params: 89,610
Non-trainable params: 0
```

A comparison on both the model training is tabulated below:

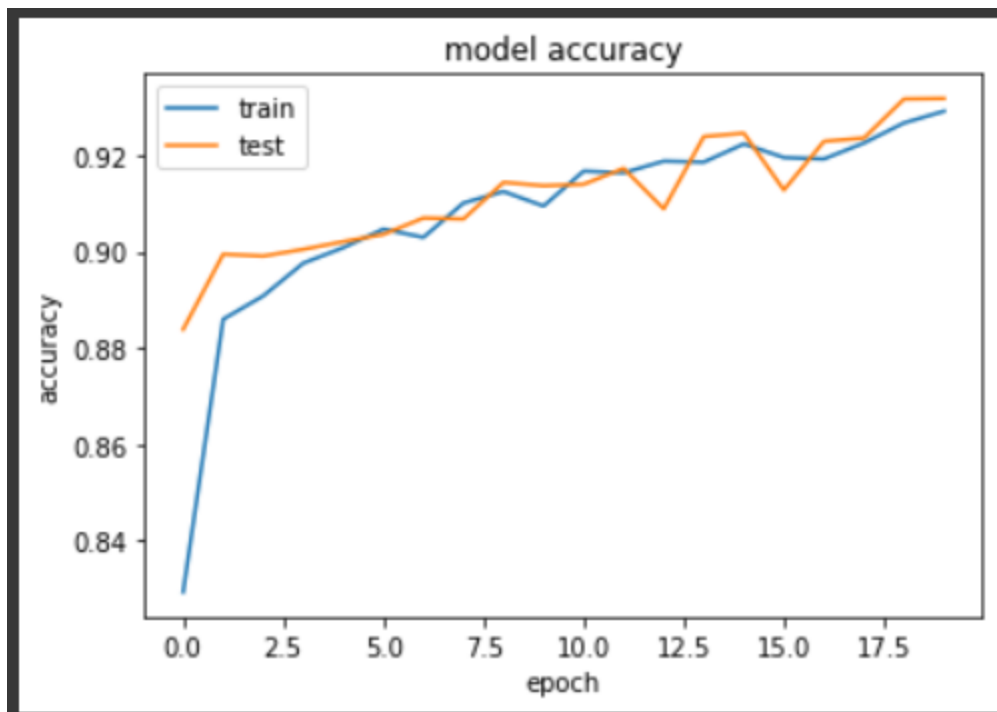| Criteria	Model 1	Model 2
# of Hidden layers	3	6
# of parameters	39760	89610
Total # of neurons	60	210

Both the models were trained with same Loss function (cross_entropy), same optimization function(Adam) and the same number of epochs(ten).

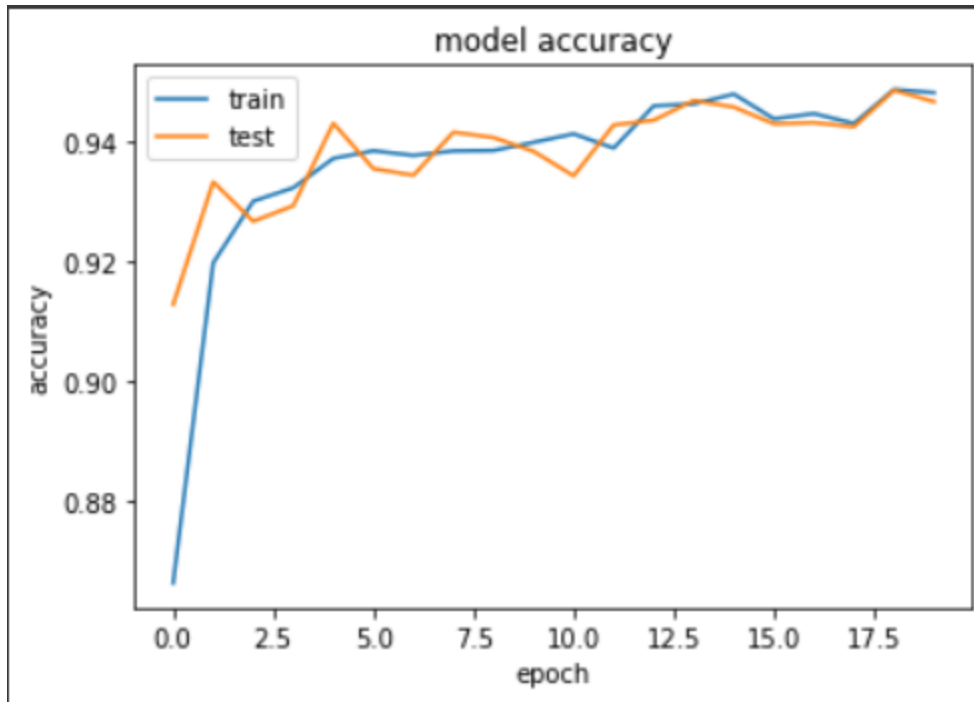Validation Accuracy of Model 1 = 93.18%
Validation Accuracy of Model 2 =94.66%

Accuracy vs Epoch Plots

Model 1

Model 2

Inference

Clearly the second model is a more complex model compared to the first one as it contains more hidden layers, more neurons, and hence more trainable parameters. From the plots, we can infer that the second model is getting overfitted towards the end epochs. Thus Model 1 shows slightly less accuracy than Model 2. The second model is getting overfitted because it is even learning the noise in the training samples and is losing its generalization ability.

It is also observed that there was a significant difference in the time, required to train these models.
Model 1 took around 6 seconds for each epoch, whereas Model 2 took around 9 seconds for the same. This can be explained by the more amount of trainable parameters in Model 2 as compared to Model 1.