CS545: Machine Learning Assignment- 2 (Neural Network)

Submitted By: Palak Goel

Dated: April 28, 2017

Experiment 1: Vary number of hidden units

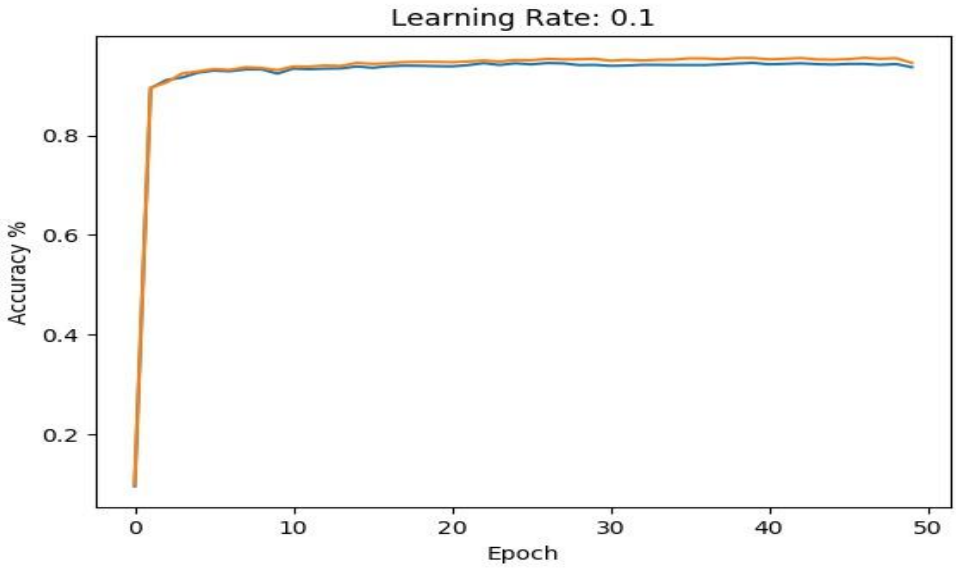
Description:

In this experiment, I have tested the neural network accuracy by changing the number of hidden units (20, 50, and 100). Inputs are 785(including bias) and outputs are 10. The learning rate and momentum is kept constant for all the different hidden unit values which are 0.1 and 0.9 respectively. Data set used is mnist data for the hand written digit recognition and have scaled the data by dividing by 255 on the scale of 0, 1.The accuracies on the training and test data are plotted for each epoch. Training dataset = 60000 and test dataset = 10000.

Note: Orange Line represents 'Training Accuracy'

Blue Line represents 'Testing Accuracy'.

- Hidden units = 20



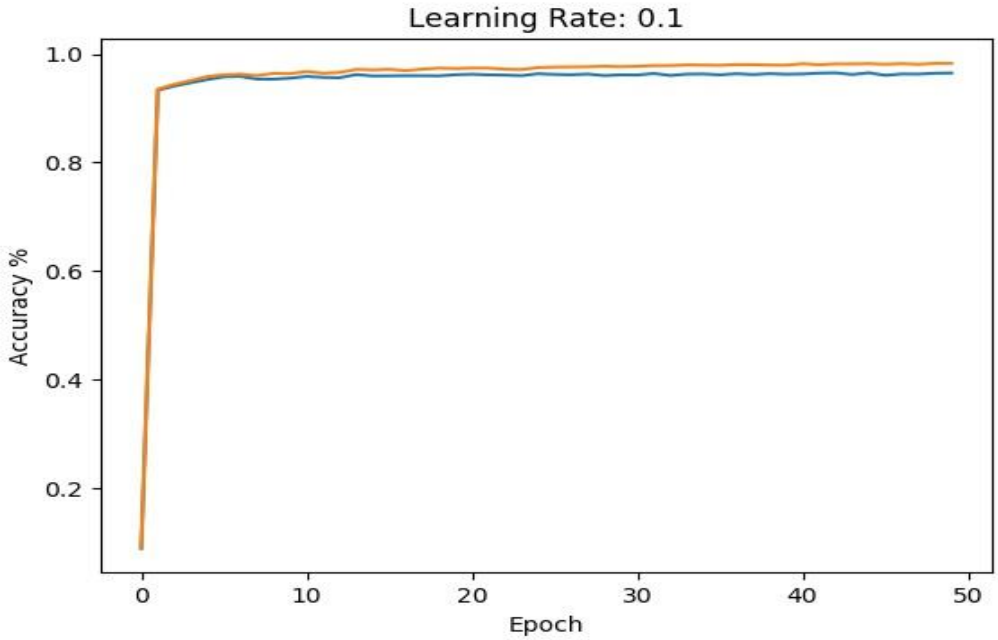
1. Hidden units = 20

Confusion Matrix:

```
[[ 961    0    0    2    0    2   11    2    2    0]
 [   0 1109    2    5    1    2    1    1   14    0]
 [  14    4  964    5    8    2    8   16   10    1]
 [   0    0   22  935    1   22    1    9   17    3]
 [   1    2    5    0  915    0   10    6    9   34]
 [  12    1    5   21    4  813   12    4   12    8]
 [  17    4    8    2    1    6  913    2    5    0]
 [   1    4   24   11    3    0    0  962    8   15]
 [   7    3    8   13    4    6    9    5  915    4]
 [   9    7    0   12   14    6    1   17   16  927]]
```

Note: Orange Line represents 'Training Accuracy'
Blue Line represents 'Testing Accuracy'.

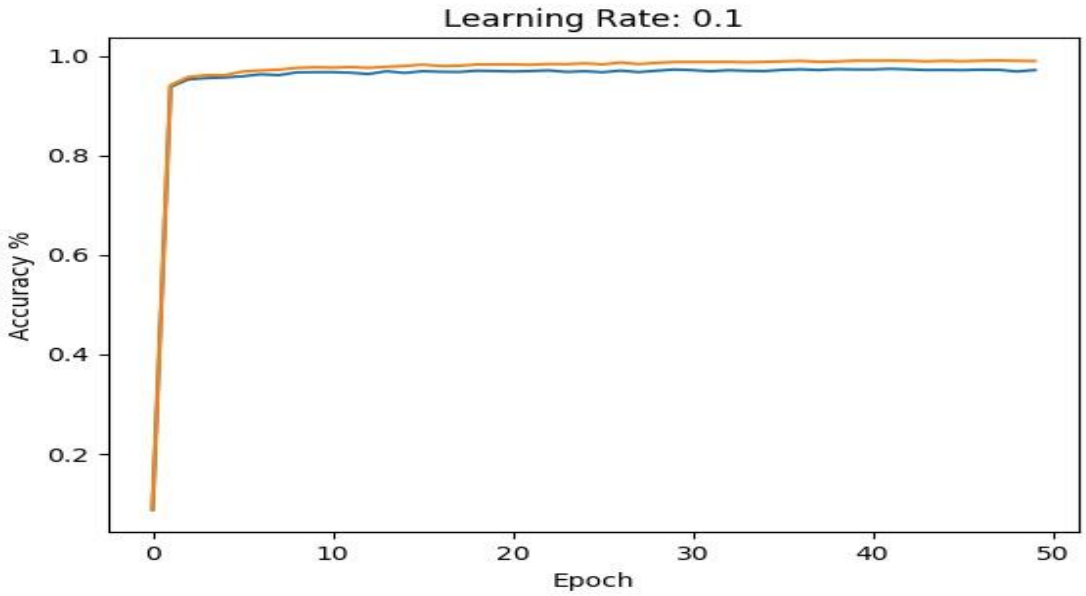
- Hidden units = 50




```
2. Hidden units = 50
Confusion Matrix:
[[ 969    1    2    1    1    2    2    1    1    0]
 [   0 1119    3    2    1    2    3    1    4    0]
 [   9    4  977   11    4    2    4    9   11    1]
 [   1    0   11  967    0   12    0    8    5    6]
 [   2    1    3    0  951    1    6    5    1   12]
 [   4    0    2   13    1  859    6    1    2    4]
 [   5    3    4    1    5    9  926    1    4    0]
 [   4    4    9    4    2    1    0  999    0    5]
 [  12    2    3   11    5   10    5    7  914    5]
 [   8    5    1    7   15   13    1    7    4  948]]
```

Note: Orange Line represents 'Training Accuracy'
Blue Line represents 'Testing Accuracy'.

- Hidden units = 100




```
3. Hidden units = 100
Confusion Matrix:
[[ 974    0    0    2    1    0    1    1    1    0]
 [   0 1117    3    1    0    1    4    1    8    0]
 [   6    4  995    3    1    2    5   10    5    1]
 [   0    0    9  977    0   14    0    5    1    4]
 [   2    0    4    0  949    1    5    0    2   19]
 [   4    0    0    4    0  872    4    2    4    2]
 [   9    4    0    0    1    7  935    0    2    0]
 [   1    3    8    4    1    0    1 1000    0   10]
 [  11    1    2    8    3    7    2    6  930    4]
 [   2    4    1    6    5    6    2    6    9  968]]
```

Question-1: How does the number of hidden units affect the final accuracy on the test data?
Answer: As we can seen from results, the accuracy on the test data increases with the number of hidden layers.

Question-2: How does it affect the number of epochs needed for training to converge?
Answer: The number of epochs needed for convergence decreased as the number of layers is increased.

Question-3: Is there evidence that any of your networks has over fit to the training data? If so, what is that evidence?
Answer: Yes, network does over fit to the training data. It can be verified as the training accuracy is higher than testing accuracy.

Question-4: How do your results compare to the results obtained by your perceptron in HW 1?

Answer: Accuracy rates are higher in neural network as compared to perceptrons. In perceptron, test accuracy was around 83% whereas using neural network, it is around 94%.

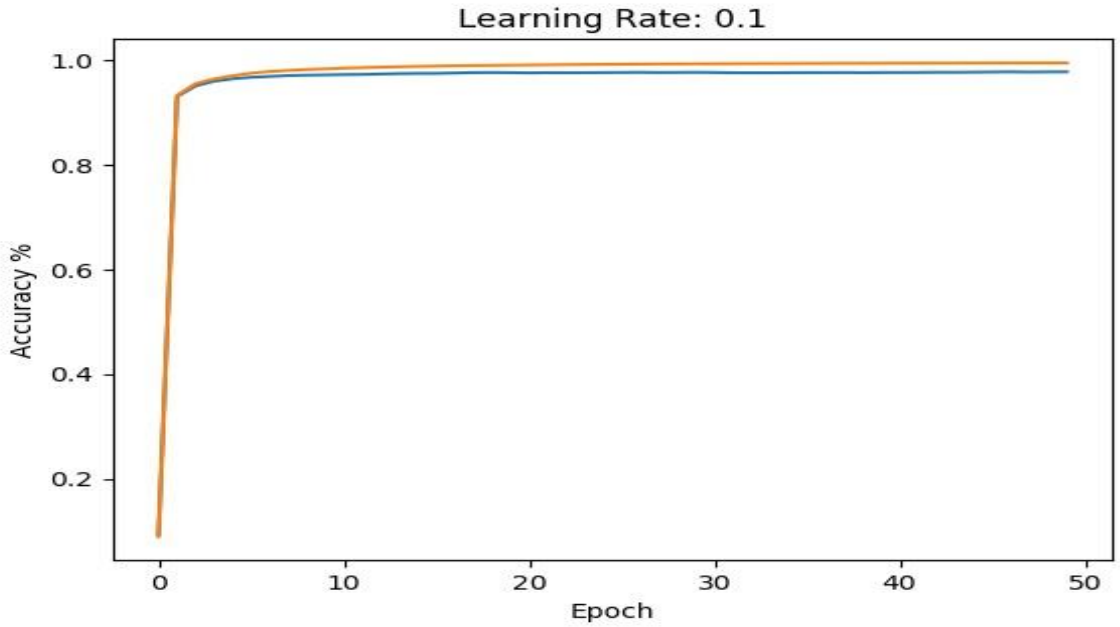
Experiment 2: Vary the momentum value

Description:

In this experiment, I have tested the neural network accuracy by changing the number of momentum values (0, 0.25 and 0.5). Inputs are 785(including bias) and outputs are 10. The learning rate and hidden units are kept constant for all the different momentum values which are 0.1 and 100 respectively. Data set used is mnist data for the hand written digit recognition and have scaled the data by dividing by 255 on the scale of 0, 1. The accuracies on the training and test data are plotted for each epoch. Training dataset = 60000 and test dataset = 10000.

Note: Orange Line represents 'Training Accuracy'
Blue Line represents 'Testing Accuracy'.

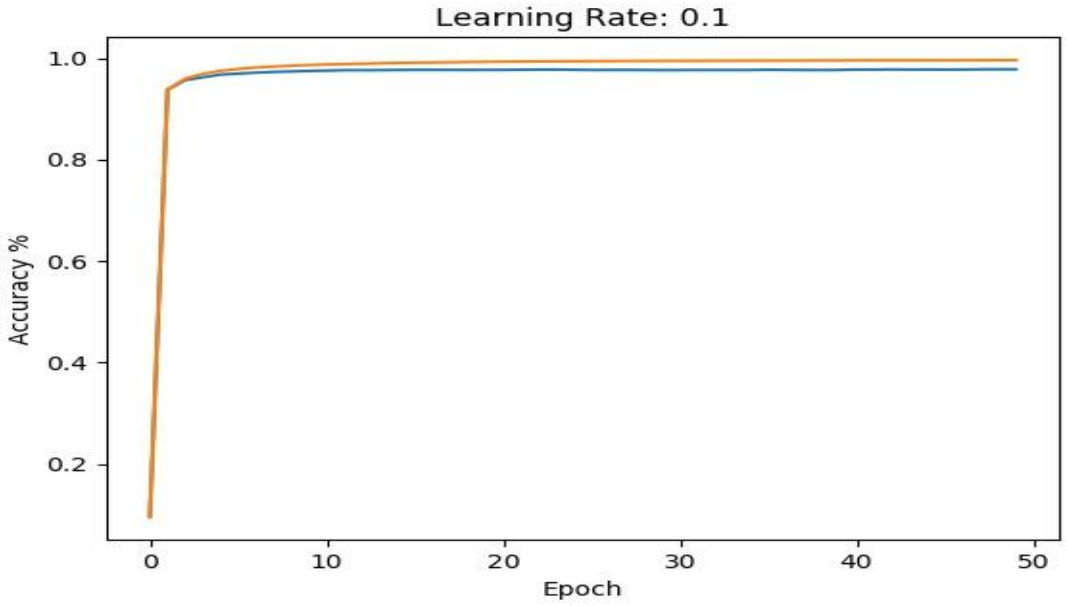
- Momentum value = 0



```
1. Momentum = 0
Confusion Matrix:
[[ 972    0    0    1    0    2    2    1    1    1]
 [   0 1123    2    3    0    0    2    1    4    0]
 [   4    2 1011    1    1    0    3    7    2    1]
 [   1    0    6  983    0    9    0    3    4    4]
 [   2    0    2    1  958    0    4    2    0   13]
 [   3    0    1    6    2  869    5    1    3    2]
 [   5    3    0    0    3    4  939    0    4    0]
 [   2    3   11    2    2    0    0 1000    2    6]
 [   3    1    3    4    5    2    3    2  948    3]
 [   4    2    1    3    9    3    1    5    1  980]]
```

Note: Orange Line represents 'Training Accuracy'
Blue Line represents 'Testing Accuracy'.

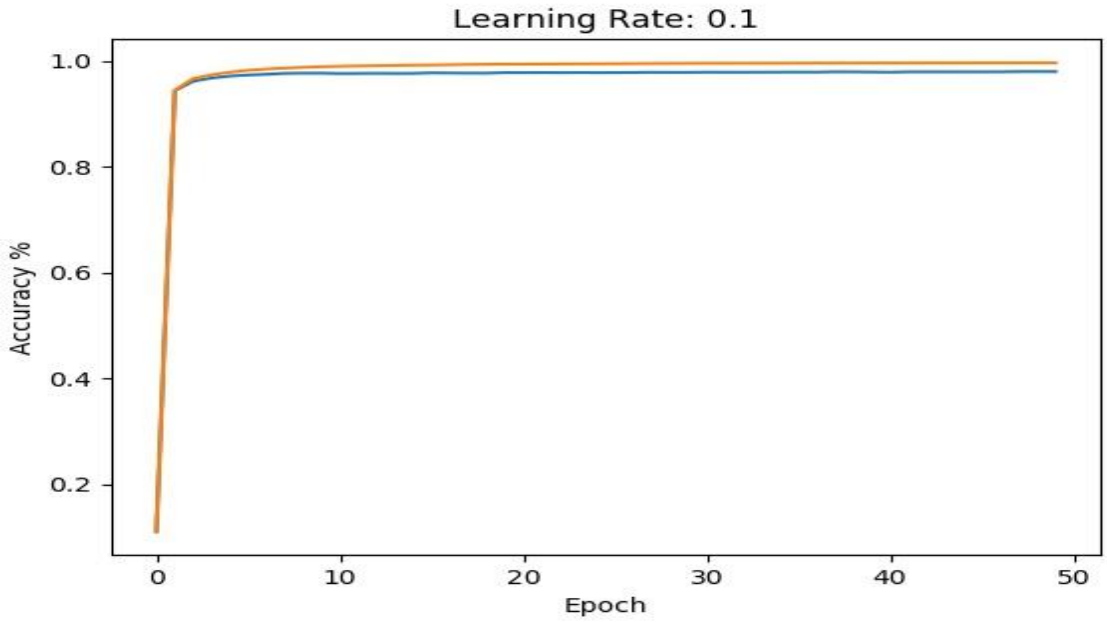
- Momentum value = 0.25



```
2. Momentum = 0.25
Confusion Matrix:
[[ 968    0    0    1    0    1    5    2    2    1]
 [   0 1125    2    1    0    0    2    1    4    0]
 [   3    1 1008    4    2    0    1    8    5    0]
 [   3    0    4  987    0    5    1    2    5    3]
 [   2    0    3    0  956    1    3    4    0   13]
 [   4    2    0    7    0  871    4    1    1    2]
 [   6    3    2    0    2    4  939    0    2    0]
 [   0    3    9    2    2    0    0 1002    2    8]
 [   7    1    5    4    4    4    0    6  941    2]
 [   3    3    0    5    9    4    1    4    1  979]]
```

Note: Orange Line represents 'Training Accuracy'
Blue Line represents 'Testing Accuracy'.

- Momentum value = 0.50



```
3. Momentum = 0.5
Confusion Matrix:
[[ 972    0    0    1    1    2    1    1    2    0]
 [   0 1124    5    0    0    0    2    1    3    0]
 [   8    3 1008    3    1    0    1    6    2    0]
 [   0    0    3  994    0    5    0    3    4    1]
 [   2    0    4    1  956    1    2    1    2   13]
 [   3    0    0    6    1  875    3    0    3    1]
 [   6    2    2    1    1    7  938    0    1    0]
 [   1    2    9    3    1    1    0 1002    2    7]
 [   4    0    3    3    4    5    1    2  949    3]
 [   2    3    1    6    9    4    1    6    1  976]]
```

Question-1: How does the momentum value affect the final accuracy on the test data?
Answer: There is not much difference between test accuracies for the different values of momentum.
When momentum = 0, test accuracy =97.8
When momentum = 0.25, test accuracy =97.7
When momentum = 0.5, test accuracy =97.9 (Highest)

Question-2: How does it affect the number of epochs needed for training to converge?
Answer: The one with higher momentum converged earlier

Question-3: Again, is there evidence that any of your networks has over fit to the training data? If so, what is that evidence?
Answer: Yes, network does over fit to the training data. It can be verified as the training accuracy is higher than testing accuracy.

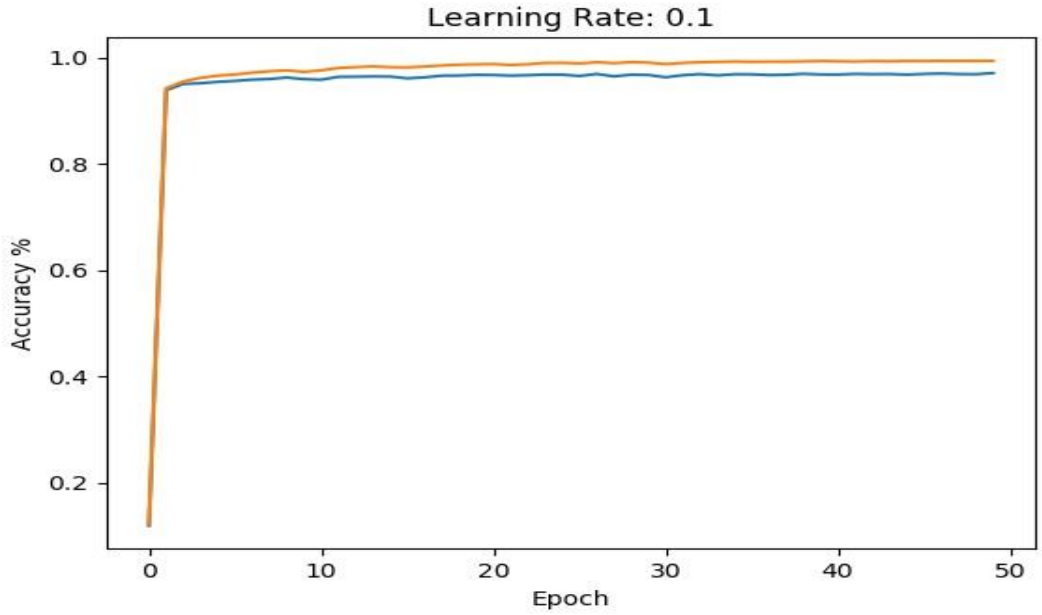
Experiment 3: Vary the number of training examples

Description:

In this experiment, I have tested the neural network accuracy by changing the size of training dataset(one quarter, one half). Inputs are 785(including bias) and outputs are 10. The learning rate, momentum and hidden units are kept constant, for the different training dataset size, which are 0.1, 0.9 and 100 respectively. Data set used is mnist data for the hand written digit recognition and have scaled the data by dividing by 255 on the scale of 0, 1. The accuracies on the training and test data are plotted for each epoch. Test dataset = 10000.

Note: Orange Line represents 'Training Accuracy'
Blue Line represents 'Testing Accuracy'.

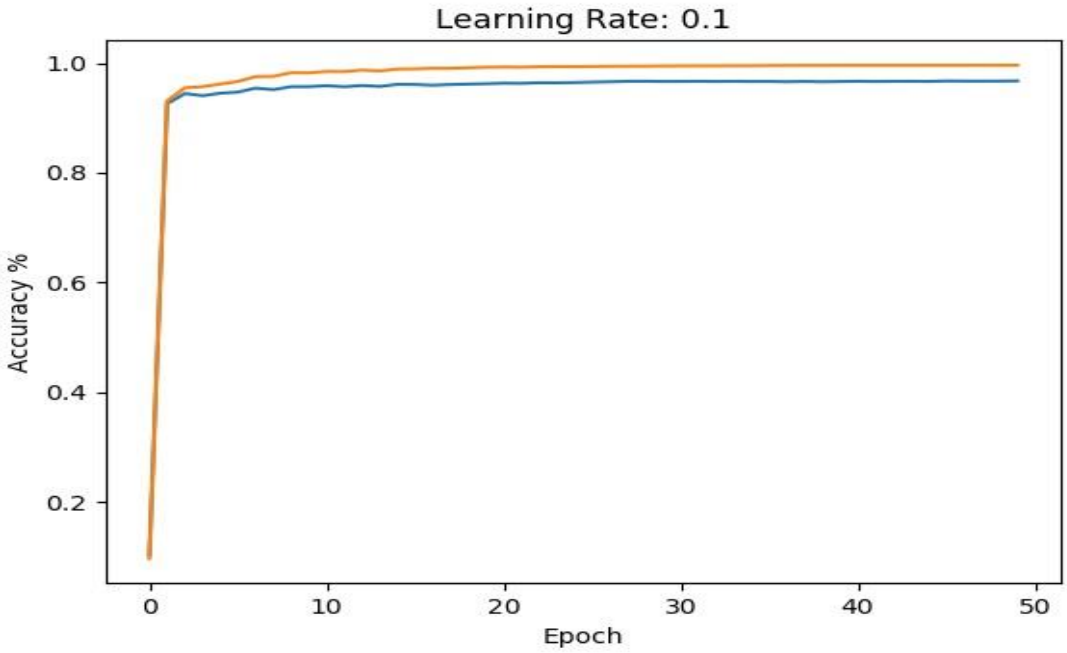
- Training Set Size = 30000 (One Half)




```
1. Training set = 15000
Confusion Matrix:
[[ 966    0    3    1    0    0    4    2    3    1]
 [   1 1123    2    3    1    1    2    0    2    0]
 [   2    1 1000    7    6    2    2    7    5    0]
 [   1    0    7  985    0    1    1    5    5    5]
 [   2    0    4    0  952    0    5    1    1   17]
 [   2    2    0   14    1  855    7    1    3    7]
 [   7    3    1    0    3    3  937    0    3    1]
 [   3    6   11   10    1    0    0  990    1    6]
 [   8    2    6    5    4    5    7    4  926    7]
 [   2    4    1    8   12    2    1    7    2  970]]
```

Note: Orange Line represents 'Training Accuracy'
Blue Line represents 'Testing Accuracy'.

- Training Set Size = 15000(One Quarter)




```
2. Training set = 30000
Confusion Matrix:
[[ 969    1    1    2    0    0    2    3    2    0]
 [   0 1124    1    4    0    2    1    1    2    0]
 [   7    2  995    4    4    0    5    7    6    2]
 [   1    0    4  986    2    4    0    2    7    4]
 [   2    2    2    0  949    0    7    1    2   17]
 [   5    1    2   10    1  859    9    2    2    1]
 [   9    3    1    1    5    6  929    1    3    0]
 [   2   13   15    2    0    0    0  977    3   16]
 [   6    2    5   11    8    4    9    5  919    5]
 [   3    5    0    4   10    6    2    8    5  966]]
```

Question-1: How does the size of the training data affect the final accuracy on the test data?
Answer: The difference between test accuracies was not much.
When training set =30000, test accuracy = 97.04%
When training set =15000, test accuracy = 96.84%
If training dataset will be too small, network will not able to learn efficiently i.e. under fitting. If dataset will be too large, there will be over fitting.

Question-2: How does it affect the number of epochs needed for training to converge?
Answer: The network with the one quarter converges earlier as compared to the one half.

Question-3: Again, is there evidence that any of your networks has over fit to the training data? If so, what is that evidence?
Answer: Yes, network does over fit to the training data. It can be verified as the training accuracy is higher than testing accuracy.