Programming #1

Kirtan Patel **PSU ID: 973582669**

Neural Network

MNIST Dataset is being used to implement the Neural Network structure in this Programming assignment. The Neural Network used has 785 input units, one hidden layer with 'n' units and 10 output units, each corresponding to classes (0-9) and the network is fully connected. The bias values of both input and hidden layer is considered as '1'. The weights of each connection are randomly chosen between the intervals (-0.05, 0.05).

Here Back-Propagation with stochastic Gradient descent is used to train the network, with Learning rate 0.1 and momentum 0.9.

Experiment 1: Vary the hidden nodes

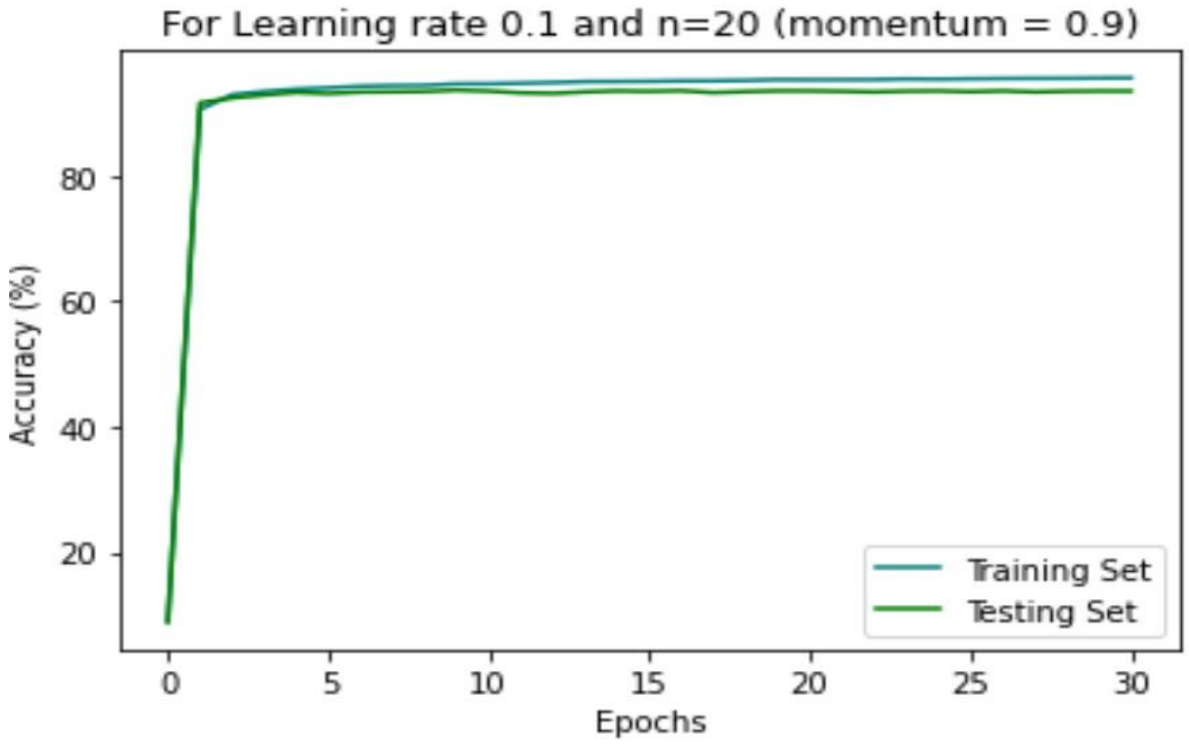
Case 1: hidden nodes= 20, Learning rate=0.1, Momentum=0.9

Confusion matrix for testing and training set

```
for testing
confusion matrix for epoch  30
[[5751    0   29   13    8   20   28    1   62   11]
 [   2 6515   56   52    8   19    2   10   64   14]
 [  19   13 5628   42   64   10   36   56   73   17]
 [   6    2   96 5732   10   59   10   37  141   38]
 [   6   17   26    3 5530    4   46    7   38  165]
 [  25    4   31   77   14 5125   65    5   54   21]
 [  31    6   29    1   23   54 5732    0   42    0]
 [   2   13   42   12   34    4    3 6036   25   94]
 [   9   40   21   34   11   39   18    3 5642   34]
 [  18    7   12   57   87   35   11   34   60 5628]]
```

```
for training
confusion matrix for epoch  30
[[ 954    0    7    0    0    6    7    1    5    0]
 [   0 1106    1    4    0    0    4    1   17    2]
 [   0    2  960    3    8    1    6   10   37    5]
 [   2    0   13  905    0   39    4    5   36    6]
 [   1    3    7    1  887    0   10    1   13   59]
 [   7    1    8   19    2  796   16    7   24   12]
 [   6    3    5    0    4   11  910    0   19    0]
 [   0    4   19    1    3    2    2  952   15   30]
 [   7    6    4    5    5    5    8    2  921   11]
 [   3    5    0    4    9   12    1    3   19  953]]
```

Accuracy Plot for case 1



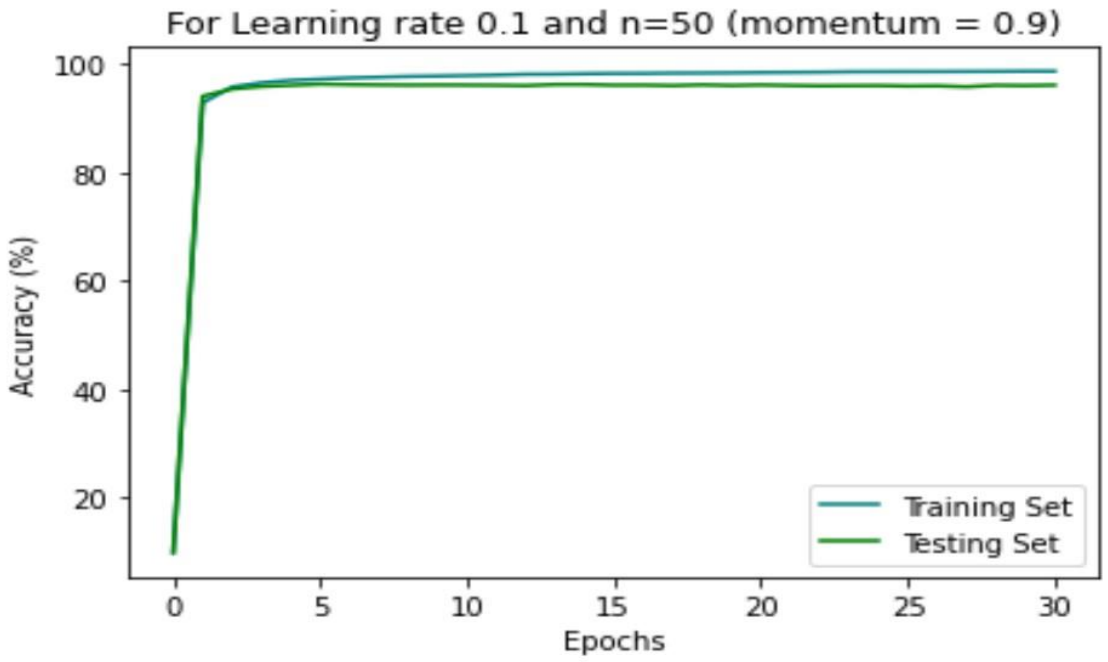
Case 2: Hidden layers=50, Learning rate=0.1, Momentum=0.9

Testing and training confusion matrix

```
for testing
confusion matrix for epoch  30
[[5844    0    6    2    2    1   17    1   43    7]
 [   2 6674   25    8    7    0    1    4   15    6]
 [   8    5 5884   11    8    0    5   17   16    4]
 [   2    2   18 6017    2   23    2    9   43   13]
 [   3    9   13    1 5741    1   15    4   10   45]
 [   9    3    5   27    1 5340   16    1   11    8]
 [  15    6    4    1   11    2 5859    0   20    0]
 [   2   10   18    4    9    0    0 6198    2   22]
 [   8   10    7    6    0    8    7    6 5791    8]
 [  18    3    2   29   18    7    2   16   30 5824]]
```

```
for training
confusion matrix for epoch  30
[[ 967    0    0    0    1    1    2    3    5    1]
 [   0 1122    3    1    1    3    1    1    3    0]
 [   7    3  984    5    4    0    4    8   15    2]
 [   2    0    3  969    0   12    0    5   12    7]
 [   1    0    2    0  937    0    7    1    5   29]
 [   4    2    1   13    0  841    9    5   11    6]
 [   7    3    4    1    2    7  927    0    7    0]
 [   2    3   16    0    7    0    0  969    3   28]
 [  11    1    3    3    6    5    6    3  926   10]
 [   5    6    0   13   11    2    0    2    7  963]]
```

Accuracy Plot for case 2

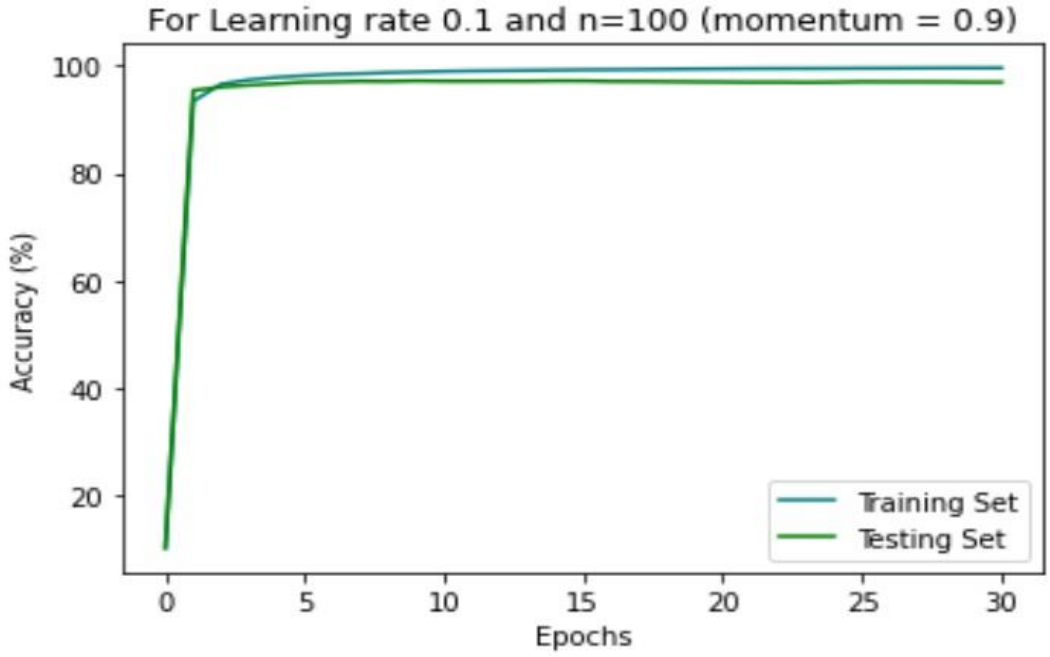


Case 3: Hidden Layer=100, Learning rate=0.1, Momentum=0.9

Testing and Training Confusion Matrix

```
for testing
confusion matrix for epoch  30
[[5899    0    2    0    0    0    4    1   16    1]
 [   1 6725    5    1    2    0    2    1    5    0]
 [   6    1 5934    3    3    0    1    2    7    1]
 [   1    0   10 6084    0    1    0    6   19   10]
 [   2    0    6    0 5815    0    2    1    6   10]
 [   0    1    5    7    0 5388    8    0    9    3]
 [   6    1    1    0    1    2 5899    0    8    0]
 [   0    4    6    3    2    0    0 6233    2   15]
 [   3    2    1    1    0    2    2    1 5835    4]
 [   8    0    2    6    2    3    0    4    9 5915]]
```

```
for training
confusion matrix for epoch  30
[[ 970    0    0    0    0    1    4    1    4    0]
 [   1 1124    2    1    0    1    2    2    2    0]
 [   3    2  998    7    1    1    4    8    7    1]
 [   0    2    7  972    0   12    0    5    9    3]
 [   1    0    6    1  945    0    3    1    2   23]
 [   6    0    2    6    0  853    7    2   10    6]
 [   6    4    1    0    0    9  932    0    6    0]
 [   1    4   14    0    2    0    0  990    5   12]
 [   4    2    2    3    5    3    3    2  945    5]
 [   5    6    1    8    6    2    0    6   15  960]]
```



Report

In this experiment we have variable hidden rates (n). The network is trained and the weights are updated and the accuracy is calculated for training and test data, after each epoch and plotted on an accuracy graph after 30 epochs.

1. How does the number of hidden units effect the final accuracy on the test data?

   With the increase in number of hidden units, we can see that the accuracy of testing data increases. When n=20 accuracy was 93.44% whereas with n=100 accuracy was 96.89%. The difference is of around 3.5%.

2. How does it affect the number of epochs needed for training to converge?

   Higher the number of hidden layers lesser will be the epochs needed for convergence. For 50 hidden units it took 5 epochs to reach 96% accuracy for training and it took little more than 30 epochs to reach 96% accuracy for n=20.

3. Is there any evidence that any of your networks has overfit to the training data? If so, what is the evidence?

   There is overfitting in network when n=100. After 25 epochs the model performed very well in training but not during testing phase. We can see that from deviation from graph.

4. How do your results compare to the results obtained by your perceptron in HW1?

   In HW1, single perceptron with learning rate 0.1, after 60 epochs accuracy was 90% in training and 88% in testing.

   Now with just n=20 with learning rate 0.1, after 30 epochs we have 96% accuracy for training set and 93% accuracy for testing set.

Experiment 2: Vary the momentum

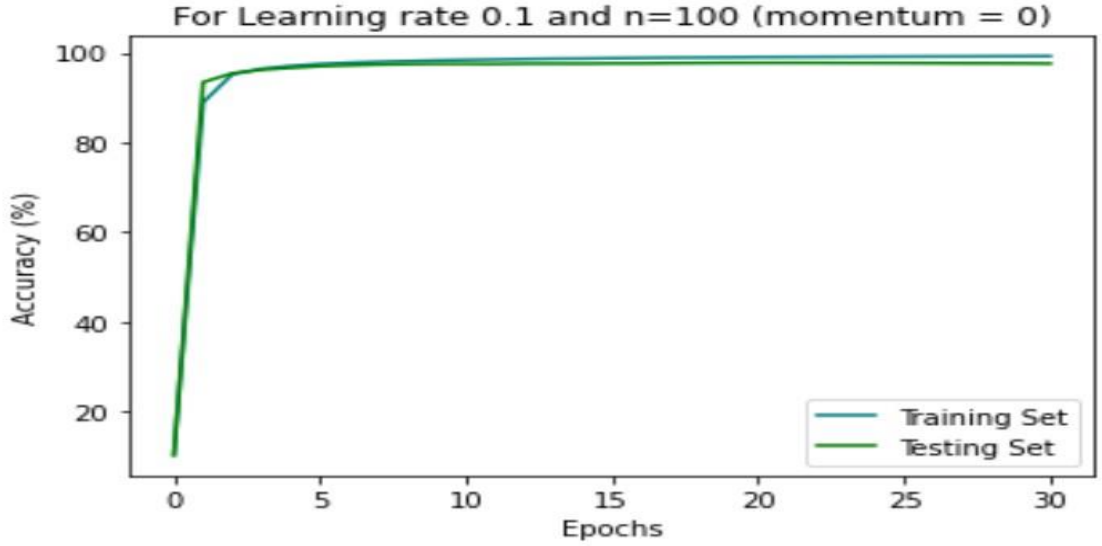
Case 1: n=100, learning rate=0.1, momentum=0

Training and Testing Confusion Matrix

```
for training
confusion matrix for epoch  30
[[ 972    1    2    1    0    1    0    1    2    0]
 [   0 1125    2    2    0    1    3    0    2    0]
 [   9    1 1001    6    1    0    2    7    4    1]
 [   1    0    2  989    1    5    1    3    2    6]
 [   1    0    0    1  959    0    6    0    2   13]
 [   3    0    0    7    1  862    7    2    5    5]
 [   7    3    0    0    1    6  936    0    5    0]
 [   1    2   11    4    4    0    0  988    2   16]
 [   3    1    3    3    4    3    3    3  945    6]
 [   3    5    0    5    6    1    2    3    2  982]]
```

```
for testing
confusion matrix for epoch  30
[[5899    1    3    1    0    0    6    0   10    3]
 [   1 6697   10    2    6    1    2   10   10    3]
 [  12    3 5920    1    1    1    1   11    6    2]
 [   2    1   13 6068    0    5    1   18   13   10]
 [   2    7    2    0 5800    0    8    2    3   18]
 [  10    2    6   13    8 5357   11    0    7    7]
 [   8    6    1    0    1    3 5891    0    8    0]
 [   4    7   15    1    5    1    0 6214    1   17]
 [   5    4    9    3    0    5    5    1 5814    5]
 [  11    3    1    2   12    5    2   19    7 5887]]
```

Accuracy Plot for Experiment 2 Case 1



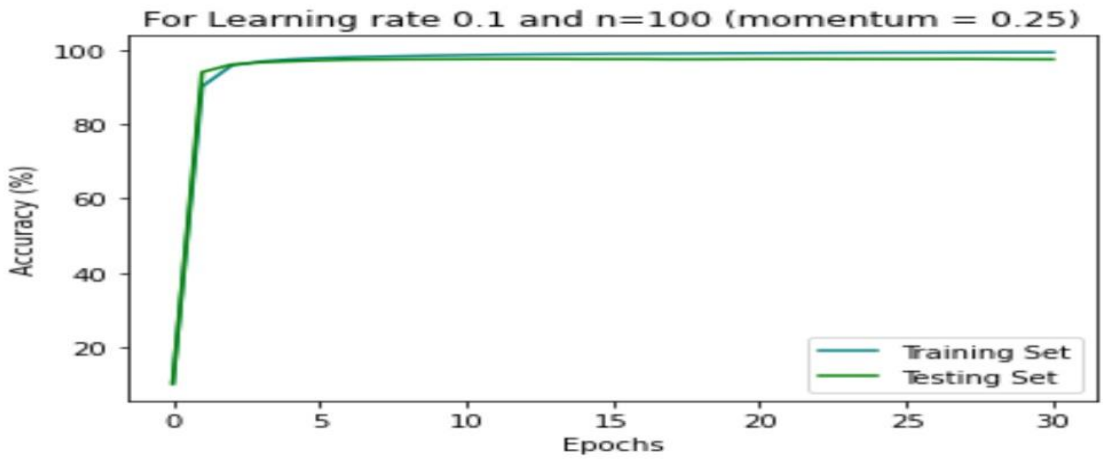
Case 2: n=100, learning rate=0.1, momentum=0.25

Training and testing confusion matrix

```
for training
confusion matrix for epoch  30
[[ 970    1    0    0    0    2    4    1    2    0]
 [   0 1127    1    2    0    1    2    1    1    0]
 [   4    2  997    7    4    1    2    9    5    1]
 [   0    0    6  985    0    9    0    3    3    4]
 [   1    0    0    0  958    0    5    1    1   16]
 [   5    1    0    6    0  860    9    2    3    6]
 [   5    3    0    1    1    4  941    0    3    0]
 [   3    4   12    5    1    0    1  991    3    8]
 [   3    1    2    5    6    2    4    4  944    3]
 [   6    6    0    7    8    1    0    4    6  971]]
```

```
for testing
confusion matrix for epoch  30
[[5902    0    4    1    0    0    6    1    6    3]
 [   1 6702   17    3    2    0    1    9    5    2]
 [   8    4 5925    1    3    0    1    9    4    3]
 [   0    2    9 6072    0    7    1   16   12   12]
 [   3    6    5    0 5809    0    4    1    3   11]
 [   6    1    1    7    3 5384   10    0    5    4]
 [  12    4    2    0    2    7 5884    0    7    0]
 [   2    6   12    3    3    1    0 6225    2   11]
 [   6    4    7    4    2    3    5    1 5813    6]
 [   8    3    3    7   11    4    1   17   14 5881]]
for training
```

Accuracy Plot for Case 2



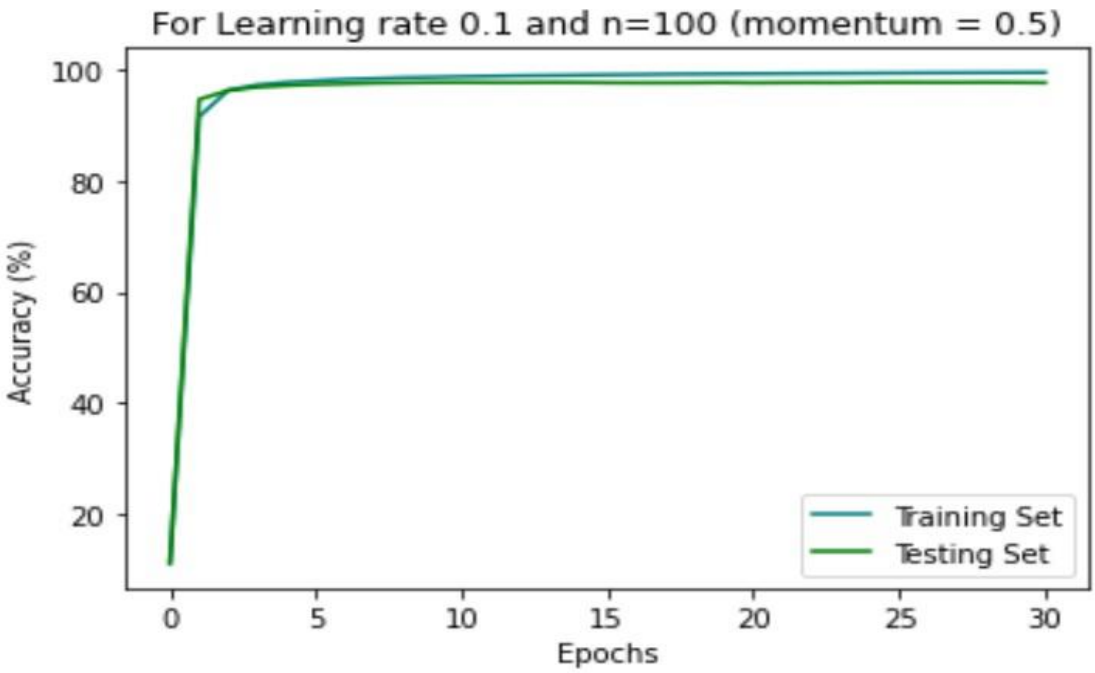
Case 3: n=100, learning rate=0.1, momentum=0.5

Training and testing confusion matrix

```
for training
confusion matrix for epoch  30
[[ 974    2    0    0    0    0    2    1    1    0]
 [   0 1122    2    4    1    1    2    1    2    0]
 [   3    2 1008    5    2    0    0    5    6    1]
 [   1    0    6  985    0    6    0    5    3    4]
 [   1    0    0    0  962    0    5    0    2   12]
 [   4    1    0    8    0  856    8    1    6    8]
 [   5    3    0    1    1    5  940    0    3    0]
 [   1    2   12    4    1    0    0  997    4    7]
 [   4    1    3    1    4    5    4    4  943    5]
 [   2    6    0    7    9    1    1    5    3  975]]
```

```
for testing
confusion matrix for epoch  30
[[5901    1    3    1    1    1    4    0    9    2]
 [   2 6704   12    2    4    2    2    4    7    3]
 [   5    3 5933    2    2    0    0    5    6    2]
 [   2    0    3 6103    0    2    0    9    7    5]
 [   2    6    2    0 5814    0    3    2    2   11]
 [   4    1    2    9    3 5375    8    2    9    8]
 [  11    3    1    0    1    3 5892    0    7    0]
 [   3    6    9    1    0    0    0 6234    1   11]
 [   0    5    4    1    0    3    2    1 5831    4]
 [   8    4    0    6   10    6    0   15    8 5892]]
```

Accuracy Plot for Case 3:



Report

Here the number of hidden layers is fixed n=100 and momentum value is varied to 0, 0.25, 0.50 and 0.9. The model and accuracy is plotted after 30 epochs. Here in this experiment we can observe that the accuracy can be achieved much faster with higher momentum.

1) How does the momentum value affect the final accuracy?

   As the momentum increases the accuracy for testing is reached faster. That means it needs less epochs to reach the accuracy. When the momentum is 0 the final accuracy is 99.2% after 30 epochs. But when momentum was 0.5 it reached the same accuracy much faster i.e. just after 16 epochs.

2) How does it affect the number of epochs needed for training to converge?

   The convergence happens much faster with smaller number of epochs when the momentum is high.

3) Again, is there evidence that any of your network has overfit to the training data? If so, what is the evidence?

   Yes, we can see overfit case, we can observe that when momentum=0.5 training dataset works very well after after 4th epoch with constant increase in accuracy but it doesn't work very well in testing dataset where accuracy doesn't change.

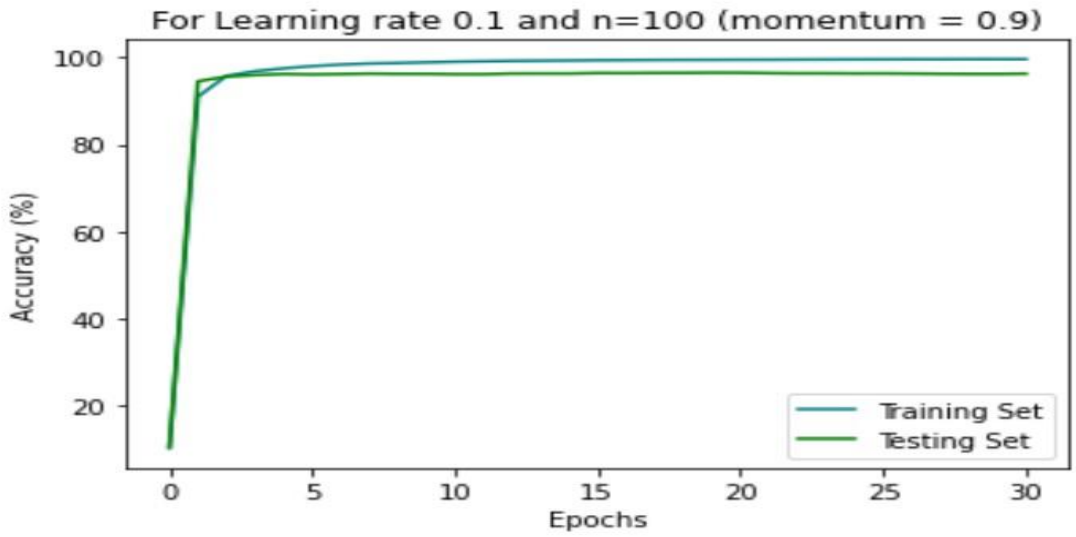
Experiment 3: Training dataset changed to 30k and 15K

Training and testing confusion matrix

```
for training
confusion matrix for epoch  30
[[ 966    0    2    1    1    0    3    1    6    0]
 [   0 1120    3    5    0    1    1    1    4    0]
 [   2    2  981    8    5    0    3    6   22    3]
 [   1    1    5  983    0    2    0    4   13    1]
 [   1    1    0    0  959    0    6    1    3   11]
 [   3    1    1   27    3  829   13    2   12    1]
 [   5    3    0    0    4    7  932    0    7    0]
 [   0    5   13    3    4    0    0  986    6   11]
 [   6    1    3    3    4    4    3    2  943    5]
 [   2    8    1   18   20    2    1    6   19  932]]
```

```
for testing
confusion matrix for epoch  30
[[2990    0    1    0    0    0    0    0    9    2]
 [   0 3359    9    1    0    1    1    0    5    0]
 [   0    0 2998    1    0    0    0    0    2    1]
 [   0    0    4 3033    0    1    0    4    8    3]
 [   0    1    2    0 2906    0    0    0    1    2]
 [   2    0    1    1    1 2689    4    0    2    0]
 [   0    1    0    0    0    0 2952    0    0    0]
 [   0    1    8    0    0    0    0 3137    0    5]
 [   0    0    0    0    0    0    2    0 2926    0]
 [   0    0    0    6    1    0    0    3    5 2908]]
```

Accuracy Plot for Experiment 3



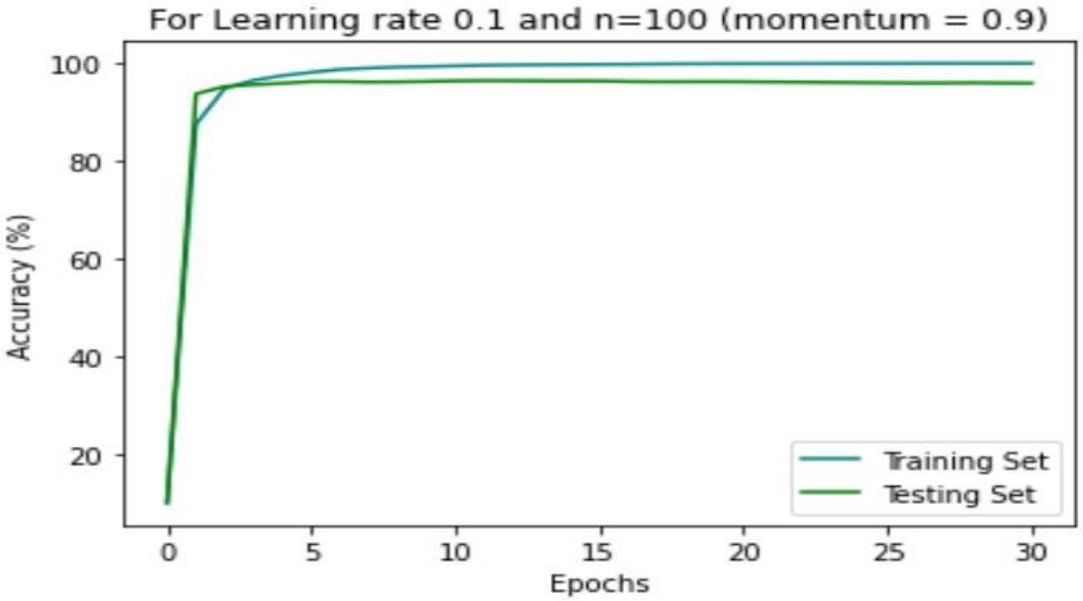
Case 2: Training dataset changed to 15k

Training and testing confusion matrix

```
for training
confusion matrix for epoch  30
[[ 967    0    2    0    1    0    4    0    4    2]
 [   1 1115    4    2    1    1    4    0    7    0]
 [   4    2  981   13    2    2    5   10   11    2]
 [   0    0    9  960    0   12    2    6   14    7]
 [   1    1    5    0  938    0    4    0    2   31]
 [   7    0    1   14    2  827    5    5   21   10]
 [  12    3    0    1    5   11  915    0   10    1]
 [   2    3    8    5    3    1    0  991    6    9]
 [   8    0    2    7    2    6    1    7  934    7]
 [   3    5    0    5   13    1    0    9   23  950]]
```

```
for testing
confusion matrix for epoch  30
[[1405    0    0    0    0    0    0    0    2    0]
 [   0 1697    0    1    2    0    0    1    1    0]
 [   0    0 1506    0    0    0    0    0    0    0]
 [   0    0    0 1525    0    0    0    0    3    1]
 [   0    0    0    0 1501    0    1    0    0    1]
 [   0    0    0    0    0 1361    0    0    1    0]
 [   0    0    0    0    0    1 1468    0    0    0]
 [   1    2    0    0    0    0    0 1543    0    1]
 [   0    0    0    0    0    0    0    0 1430    0]
 [   0    0    1    0    2    0    0    0    1 1541]]
```

Accuracy plot



Report

In this experiment, we take different training quantity (30k and 15k), by keeping the hidden layers=100, learning rate=0.1 and momentum=0.9.

In this experiment we observe that we need to give huge number of data for training to get more accuracy on testing set.

1) How does the size of the training data affect the final accuracy on the test data?

   As the size of training data was reduced the accuracy of testing data reduced. When complete data was used for training the final accuracy after 30 epochs was 99.5% but when the training data was halved the accuracy reduced to 97.5% and when the training data was reduced to 15k, the accuracy dropped to 96%.

2) How does it affect the number of epochs needed for training to converge?

   More the data, lesser the epochs needed for training to converge.
   As the training data of the model is high, it needs less epochs to reach maximum testing accuracy. In Experiment 1 when we sent full data it took only 2 epochs, but when data was halved it took 5 epochs and took 8 epochs when we sent quarter of data.

3) Is there any evidence that any of your networks has overfit to the training data? If so what is the evidence?

   Yes, we see overfit in case 1 where we have given 50% data, after 12 epochs we can see deviation between training and testing data from the graph