Cera Oh
ML
Winter 2021
1/25/2022

Programming 1 Report

For this assignment, we were analyzing MNIST data set, which contains 60,000 training data samples and 10,000 test data samples. The purpose is to recognize handwritten numbers 0 through 9 using a neural network with one hidden layer. Initial weights for output and hidden layer neurons were random numbers between -0.05 and 0.05. Data was preprocessed to be between 0 and 1 by dividing the input values of data samples by 255. The number of neurons in the hidden layer varied for experiment 1. For experiments 2 and 3, the number of neurons in the hidden layer was kept at n = 100. Default momentum was at 0.9 and the learning rate was 0.1. Momentum varied for experiment 2. For experiment 3, the training data numbers varied.

Each neuron in the hidden layer was connected to 785 input nodes, including a bias node. 784 inputs each represented a pixel of 28x28 pixel handwriting sample. Each of the hidden layer neurons and a bias node connected to each of the 10 output neurons. Each output neuron represented one of the possible number predictions, 0 through 9. For each writing sample data, the output neuron with the highest value was chosen as neuron that fired the prediction, and the prediction was checked against the correct label. The Sigmoid activation function was used for calculation. If the prediction was correct, next sample was given. If the prediction was wrong, the weights of neural networks were updated using backward propagation equations given in the lecture. The neural network was trained for 50 epochs, and accuracy per epoch for training data and test data were recorded. Confusion matrix for testing data was also recorded. Each experiment will be explained below.

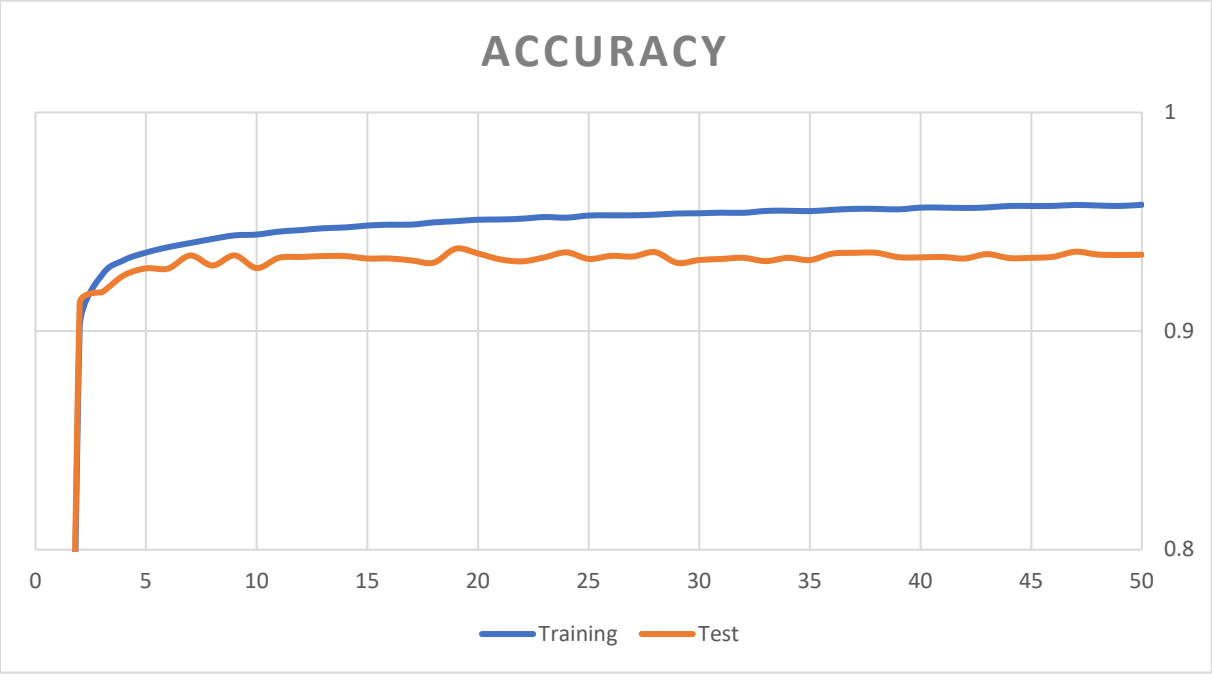
Experiment 1

The purpose of experiment 1 is to vary the number of neurons in the hidden layer to see if this had any effect on accuracy. The varying number of hidden layer neurons were n=20, 50, and 100. Momentum was kept constant at 0.9 and learning rate was kept at 0.1 for all three cases. All other details remain the same throughout the experiment.

Cera Oh
ML
Winter 2021
1/25/2022

Data

Accuracy per Epoch Graph for n = 20



Comment: Epoch 0 had initial accuracy of 0.10218333333333333. After 1 epoch, training accuracy jumped to 0.9015666666666666. Final accuracy for training was 0.9576833333333333 and for testing was 0.9341. While training accuracy stably increased with minimal oscillation, oscillation can be seen for testing accuracy.

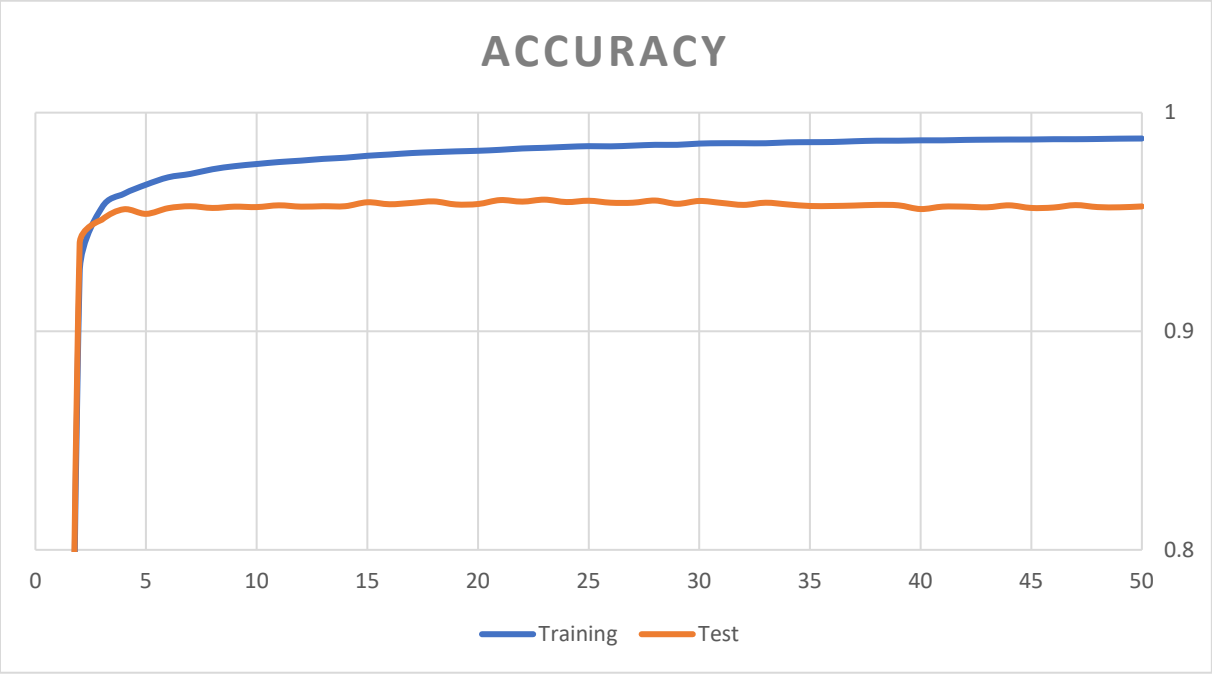
Confusion Matrix after 50 epochs of test for n = 20 (in 10^2)

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	474.96	0.05	0.44	0.60	0.92	1.21	2.70	1.60	6.30	1.22
1	0	555.21	7.80	2.96	0.1	1.02	1.85	0.37	4.97	0.24
2	1.12	1.94	464.02	8.43	4.53	0.67	4.72	4.13	23.84	2.60
3	1.52	0.02	5.14	464.17	0.06	13.38	1.13	2.72	13.11	3.75
4	0.47	0.67	2.59	0.16	448.94	0	7.79	0.67	2.22	27.49
5	2.67	0.54	0.66	11.00	1.57	400.68	5.59	3.38	12.90	7.01
6	5.68	1.47	0.50	0.2	2.91	7.68	450.25	0.25	10.00	0.06
7	0.28	3.31	9.08	4.75	2.13	0.43	0.36	476.27	5.24	12.15
8	3.83	3.33	1.94	5.30	4.00	3.50	5.14	0.88	452.76	6.32
9	1.82	2.61	0.41	4.09	6.07	1.44	0.14	2.76	9.20	475.96

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused the most per label.

Cera Oh
ML
Winter 2021
1/25/2022

Accuracy per Epoch Graph for n = 50



Comment: Epoch 0 had initial accuracy of 0.11435. After 1 epoch, training accuracy jumped to 0.9269833333333334. Final accuracy for training was 0.98835 and for testing was 0.9565. While training accuracy stably increased, slight oscillation can be seen for testing accuracy.

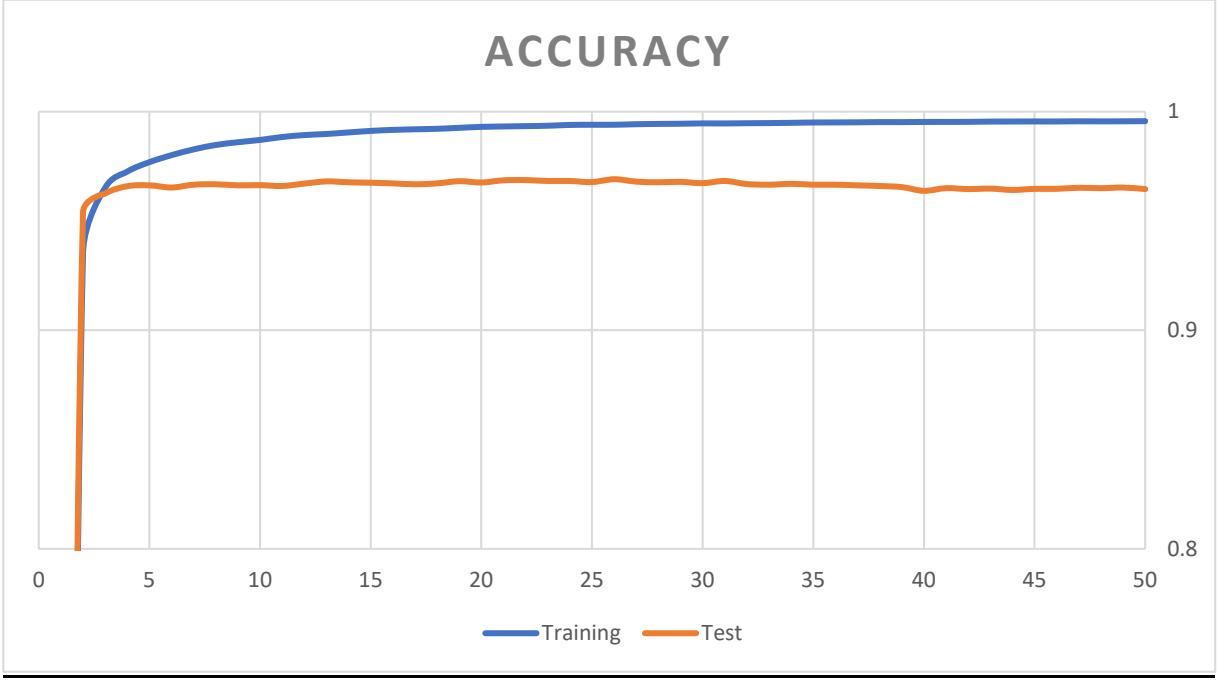
Confusion Matrix after 50 epochs of test for n = 50 (in 10^2)

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	482.11	0.38	0.02	0.57	0.04	2.77	1.36	0.63	1.98	0.14
1	0	560.40	0.93	2.11	0.21	0.53	1.02	0.13	2.15	0.02
2	4.26	1.40	485.12	5.15	1.63	0.76	2.47	3.49	9.57	2.15
3	0.45	0.24	2.06	481.35	0.67	7.49	0.04	1.94	5.00	5.76
4	0.9	0.33	1.13	0.27	468.40	0.27	4.58	0.93	1.75	12.44
5	2.80	0.57	0.39	5.30	0.09	419.35	3.53	0.74	8.01	5.22
6	4.00	1.51	0.34	0.02	0.99	4.49	462.23	0	5.16	0.26
7	0.15	1.76	8.03	0.80	1.85	0.04	0.04	486.64	3.46	11.23
8	2.97	0.94	2.34	2.44	2.54	4.00	3.08	1.23	461.09	6.37
9	3.38	3.02	0.07	2.22	5.28	2.16	0.44	2.23	6.28	479.42

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused the most.

Cera Oh
ML
Winter 2021
1/25/2022

Accuracy per Epoch Graph for n = 100



Comment: Epoch 0 had initial accuracy of 0.1044. After 1 epoch, training accuracy jumped to 0.93505. Final accuracy for training was 0.99565 and for testing was 0.9652. While training accuracy stably increased, minor oscillation can be seen for testing accuracy.

Confusion Matrix after 50 epochs of test for n = 100 (in 10^2)

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	484.43	0.39	0.38	1.23	0.03	0.65	0.67	0.60	1.47	0.15
1	0.39	560.83	1.62	0.52	0	0.76	1.15	0.49	1.74	0
2	1.08	2.03	494.26	3.77	1.11	0.41	1.01	2.74	7.56	2.03
3	1.00	0.05	1.58	489.67	0.01	3.40	0.08	2.37	4.86	1.98
4	0.53	0.67	1.41	0	472.03	0	3.00	0.01	0.78	12.57
5	2.11	0.89	0.01	4.67	0.50	426.10	3.06	0.92	5.40	2.34
6	2.20	2.34	0.01	0.09	0.86	5.44	464.09	0	3.55	0.42
7	0.90	3.01	6.18	1.11	2.09	0.04	0.18	490.49	1.82	8.18
8	3.62	0.68	0.32	0.68	1.84	2.85	2.42	1.67	467.73	5.19
9	2.22	2.60	0.17	4.61	4.74	1.33	0.54	1.30	5.43	481.56

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused the most.

Cera Oh
ML
Winter 2021
1/25/2022

Discussion

Training accuracy increased as the number of neurons in the hidden layer increased. Test accuracy increased from n = 20 to n = 50 but remained similar for n = 50 to n = 100. 8 and 9 were the most wrong guesses the neural network made.

1) How does the number of hidden units affect final accuracy on test data?

As number of hidden units (neurons) increased, so did the training accuracy. At n = 20, the final training accuracy was 0.9576833333333333, at n = 50 it was 0.98835, and at n = 100 it was 0.99565. However, the test accuracies did not increase as much as training accuracies did. At n = 20 the test accuracy was 0.9341, at n= 50 it was 0.9565, at n = 100 it was 0.9652. This shows that n = 20 did not represent the data as well as n = 50 or n = 100, which is reflected in that the oscillation on test accuracy for n=20 is a lot worse than on the other two. However, there may have been some overfitting as the number of hidden layer neurons increased to n = 100 since test accuracy seems to be stay around what we got for n = 50. Confusion matrices show that as number of neurons increased, the neural network more times thought 8 was the right answer for different labels, which supports the idea there may have been some overfitting and picking up of noises.

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

I think n=20 neural network will possibly do better with more epochs of training. For n=50 and n=100, more epochs probably won't help training to converge since test accuracy seems to stay around 0.95-0.97 and there now might be possible overfitting. This is also reflected in the increase from Epoch 0 to Epoch 1 since n = 20 increased to 0.9015666666666666 while the other two were near 0.93, showing that n = 20 is converging slowly in comparison to n = 50 and n = 100.

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

While training accuracies seems to be steadily increasing, we see that there are

oscillations and flattening of the accuracy curves for the test accuracies. This seems to indicate that there is some overfitting. I think n = 20 might be not overfit but need more training, but n = 50 and n = 100 has increased gap between the training accuracies and testing accuracies. Further, confusion matrices also show that the most wrong guesses fell on guessing 8 as the right answer for most of the labels, which tells me it's picking up some noise from the training data.

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

The neural network with one hidden layer performed much better than my perceptron in HW1. As can be seen for all varying numbers of neurons in the hidden layers, the test and training accuracies are higher for the neural network than for the perceptron. My perceptron stayed around 0.86 test accuracy and 0.91 for training accuracy for learning rate of 0.1.

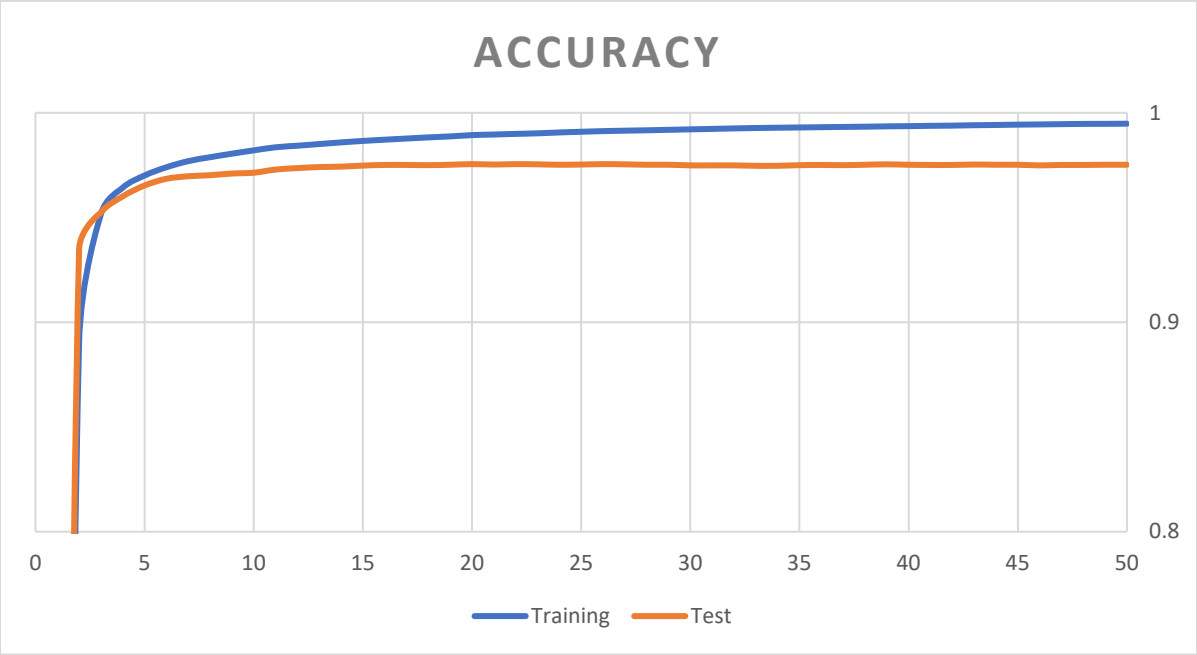
Experiment 2

The purpose of experiment 2 is to vary the rate of momentum, which is used in the weight update formula as part of the weight decay function, which helps decrease overfitting. The varying rate of momentum were 0, 0.25, 0.50 and 0.90. Learning rate was kept at 0.1 for all four cases. All other details remain same throughout the experiment.

Cera Oh
ML
Winter 2021
1/25/2022

Data

Accuracy per Epoch Graph for momentum = 0



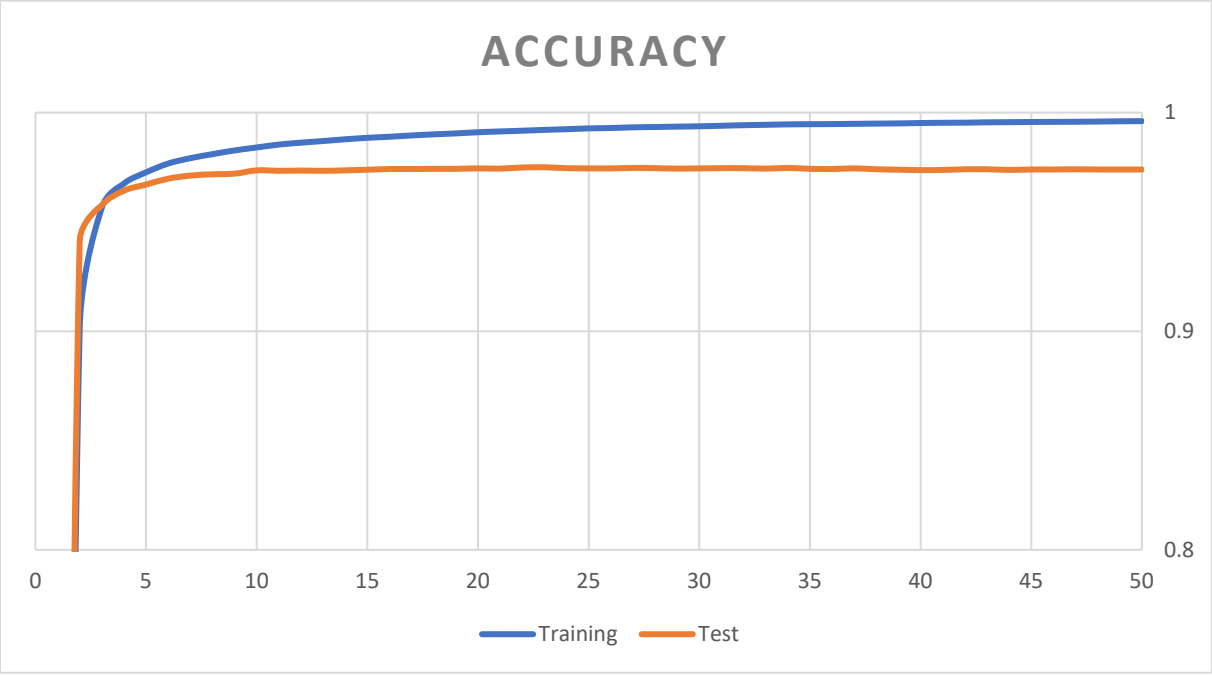
Comment: Epoch 0 had initial accuracy of 0.11305. After 1 epoch, training accuracy jumped to 0.8899833333333333. Final accuracy for training was 0.994833333 and for testing was 0.9751.

Confusion Matrix after 50 epochs of test for momentum = 0 (in 10^2)

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	484.53	0	0.06	0.75	0.09	0.31	1.34	0.50	1.95	0.47
1	0	561.71	0.94	1.69	0.02	4.90	1.16	0.22	1.27	0
2	2.15	0.86	499.20	3.01	0.80	2.00	1.23	3.44	4.81	0.48
3	0	0.02	1.24	496.01	0	0.93	2.30	2.20	2.52	1.85
4	0.50	0.02	1.47	0.03	477.22	0	2.50	0.11	0.63	8.52
5	2.86	0.19	0.24	5.51	0.18	427.52	2.55	1.08	3.10	2.77
6	3.97	1.50	0	0.61	1.28	2.83	466.97	0	1.84	0
7	0.10	2.63	6.15	1.10	0.68	0	0.11	494.31	1.42	7.50
8	1.82	0.62	0.92	3.56	2.55	2.58	1.99	1.60	469.21	2.15
9	1.81	2.34	0.02	3.50	4.10	0.97	0.99	2.46	1.16	487.15

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused the most.

Accuracy per Epoch Graph for momentum = 0.25



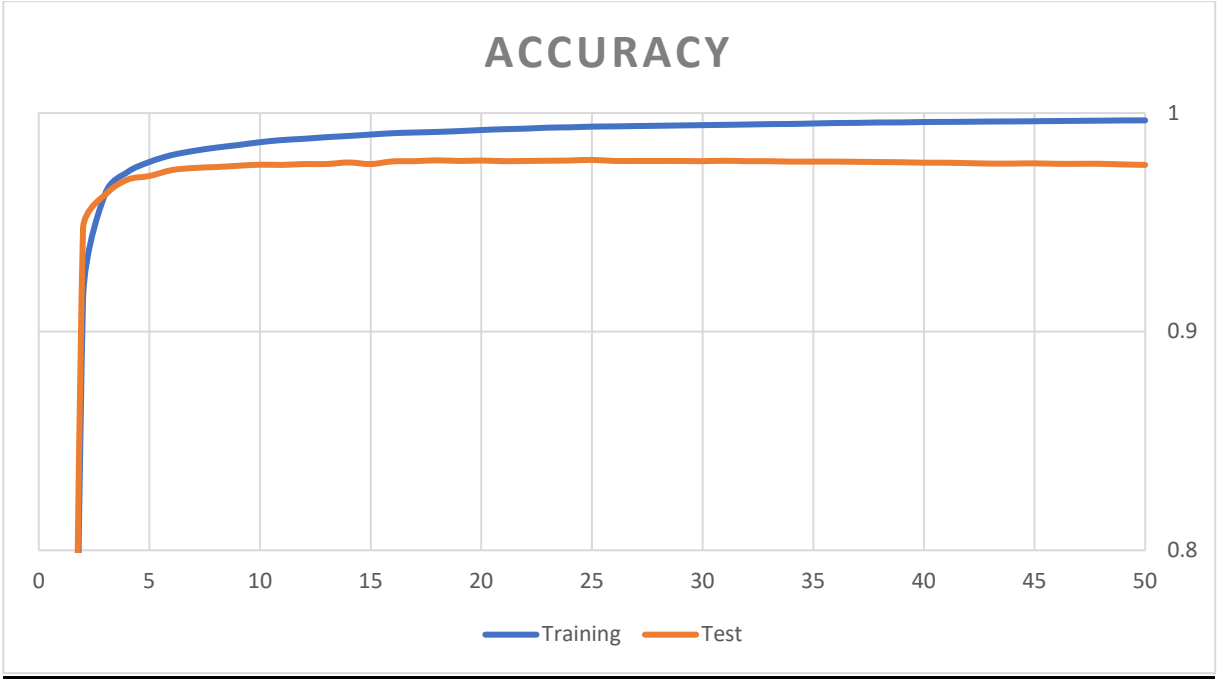
Comment: Epoch 0 had initial accuracy of 0.09756666666666666. After 1 epoch, training accuracy jumped to 0.9011166666666667. Final accuracy for training was 0.9961666666666666 and for testing was 0.9742.

Confusion Matrix after 50 epochs of test for momentum = 0.25 (in 10^2)

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	484.84	0.09	0.02	0.04	0.25	0.35	2.18	0.50	1.69	0.04
1	0	561.52	0.76	1.20	0.35	0.83	0.88	0.61	1.35	0
2	2.37	0.97	500.97	3.13	0.94	0.38	0.90	2.12	3.16	1.06
3	0	0	2.78	494.88	0.01	2.26	0	1.69	2.07	1.31
4	0.95	0.01	1.00	0	476.95	0	2.74	0.01	1.01	8.33
5	2.04	0.17	0.12	7.37	0.52	426.07	3.79	1.03	2.37	2.52
6	3.26	1.50	0.16	0.23	0.62	2.34	468.76	0	2.04	0.09
7	0.63	2.45	7.20	2.28	0.95	0.01	0	489.81	1.20	9.47
8	2.19	0.54	0.89	1.88	1.81	1.55	1.98	1.73	472.41	2.02
9	3.13	2.52	0	2.91	4.74	0.32	0.15	2.01	1.83	486.89

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused the most.

Accuracy per Epoch Graph for momentum = 0.50



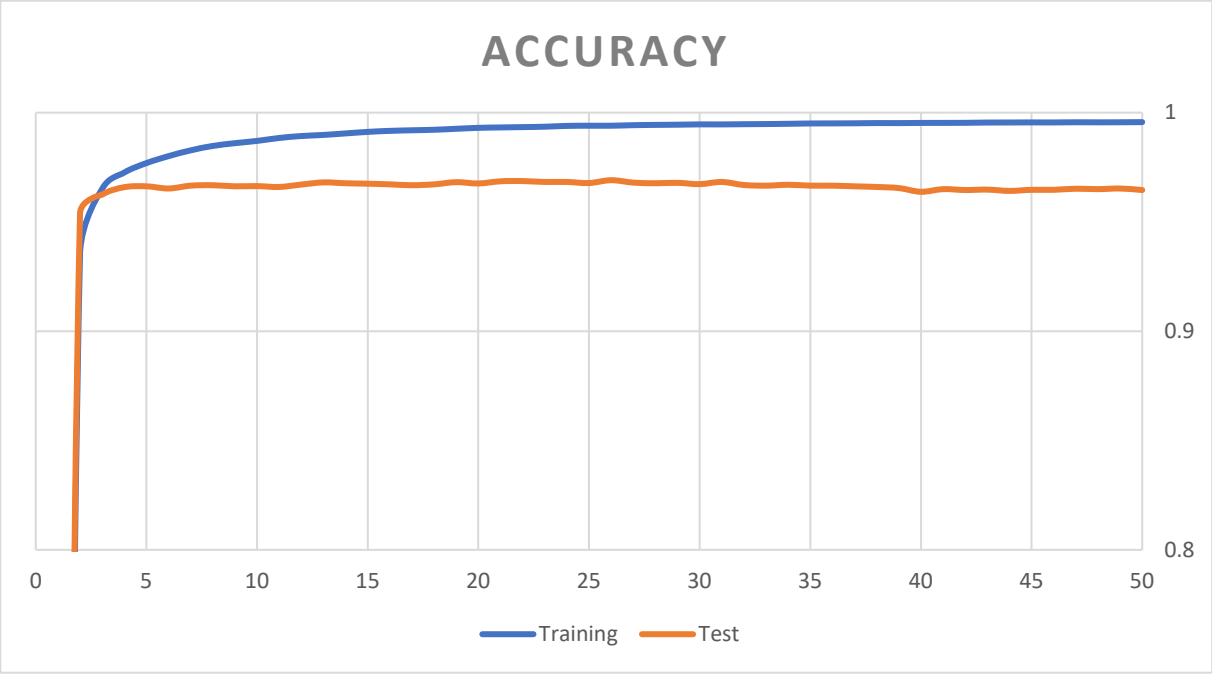
Comment: Epoch 0 had initial accuracy of 0.09035. After 1 epoch, training accuracy jumped to 0.9142666666666667. Final accuracy for training was 0.9967166666666667 and for testing was 0.9762.

Confusion Matrix after 50 epochs of test for momentum = 0.50 (in 10^2)

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	484.35	0.50	0	0.77	0.11	0.57	2.09	0.65	0.72	0.24
1	0	563.00	0.64	0.67	0	0.48	1.36	0	1.35	0
2	1.40	1.24	501.48	4.06	0.18	0.07	0.87	3.27	2.36	1.07
3	0.64	0.12	1.17	494.13	0	1.79	0.09	2.38	2.04	2.64
4	0.50	0.01	1.02	0.02	476.21	0	2.74	0	0.69	9.81
5	1.85	0.12	0.22	3.84	0.06	430.13	3.07	1.39	2.23	3.09
6	2.94	1.50	0	0	0.32	1.34	471.30	0	1.50	0
7	0.43	1.90	5.30	1.52	1.37	0.30	0.06	497.49	1.27	4.36
8	1.53	0.58	1.79	1.07	2.05	0.70	0.94	1.84	474.41	2.09
9	2.12	2.71	0	1.86	3.93	0.71	0.17	2.38	2.48	488.14

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused most.

Accuracy per Epoch Graph for momentum = 0.9



Comment: Epoch 0 had initial accuracy of 0.1044. After 1 epoch, training accuracy jumped to 0.93505. Final accuracy for training was 0.99565 and for testing was 0.9652. While training accuracy stably increased, minor oscillation can be seen for testing accuracy.

Confusion Matrix after 50 epochs of test for momentum = 0.9 (in 10^2)

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	484.43	0.39	0.38	1.23	0.03	0.65	0.67	0.60	1.47	0.15
1	0.39	560.83	1.62	0.52	0	0.76	1.15	0.49	1.74	0
2	1.08	2.03	494.26	3.77	1.11	0.41	1.01	2.74	7.56	2.03
3	1.00	0.05	1.58	489.67	0.01	3.40	0.08	2.37	4.86	1.98
4	0.53	0.67	1.41	0	472.03	0	3.00	0.01	0.78	12.57
5	2.11	0.89	0.01	4.67	0.50	426.10	3.06	0.92	5.40	2.34
6	2.20	2.34	0.01	0.09	0.86	5.44	464.09	0	3.55	0.42
7	0.90	3.01	6.18	1.11	2.09	0.04	0.18	490.49	1.82	8.18
8	3.62	0.68	0.32	0.68	1.84	2.85	2.42	1.67	467.73	5.19
9	2.22	2.60	0.17	4.61	4.74	1.33	0.54	1.30	5.43	481.56

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused the most.

Discussion

Generally, training accuracy was high for training accuracy between 0.995-0.997 and testing accuracy did well around 0.97 for momentums 0, 0.25, and 0.50. Testing accuracy dropped slightly below 0.97 when momentum was 0.90. Confusion matrices made more sense when momentum was 0, 0.25, and 0.50, but most confused numbers conversed to 8 and 9 for momentum 0.9. Therefore, momentum 0.9 behaved worst out of the four, and then momentum 0 behaved the second worst, indicating no momentum and too high of momentum makes training worse. Momentum at 0.5 had the highest accuracies.

1) How does the momentum value affect the final accuracy on the test data?

When momentum is 0, training accuracy is 0.994833333 and testing accuracy is 0.9751, training accuracy is 0.9961666666666666 and testing accuracy is 0.9742 when momentum is 0.25, training accuracy is 0.9967166666666667 and testing accuracy is 0.9762 when momentum is 0.50, and training accuracy is 0.99565 and testing accuracy is 0.9652 when momentum is 0.90. Surprisingly, momentum at 0, 0.25, and 0.50 all did well with test accuracy around 0.975. Momentum at 0.9 did the worst with the test accuracy at 0.9652. This shows that 0.9 momentum is too high, and the neural network isn't converging as well.

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

For momentum 0, the training accuracy after epoch 1 was 0.8899833333333333. This is lower than other values for momentum, which shows 0 momentum means we will need more epochs for the training to converge. After epoch 1 value for momentum 0.25 was 0.9011166666666667, for momentum 0.50 was 0.9142666666666667, and for momentum 0.9 was 0.93505. This shows that in general, higher momentum means you need less epochs for training to converge.

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

I think there is overfitting when momentum is 0.90 since it drops in test accuracy and more oscillation can be seen in the accuracy graph. Also, the confusion matrix seems to be overfitted

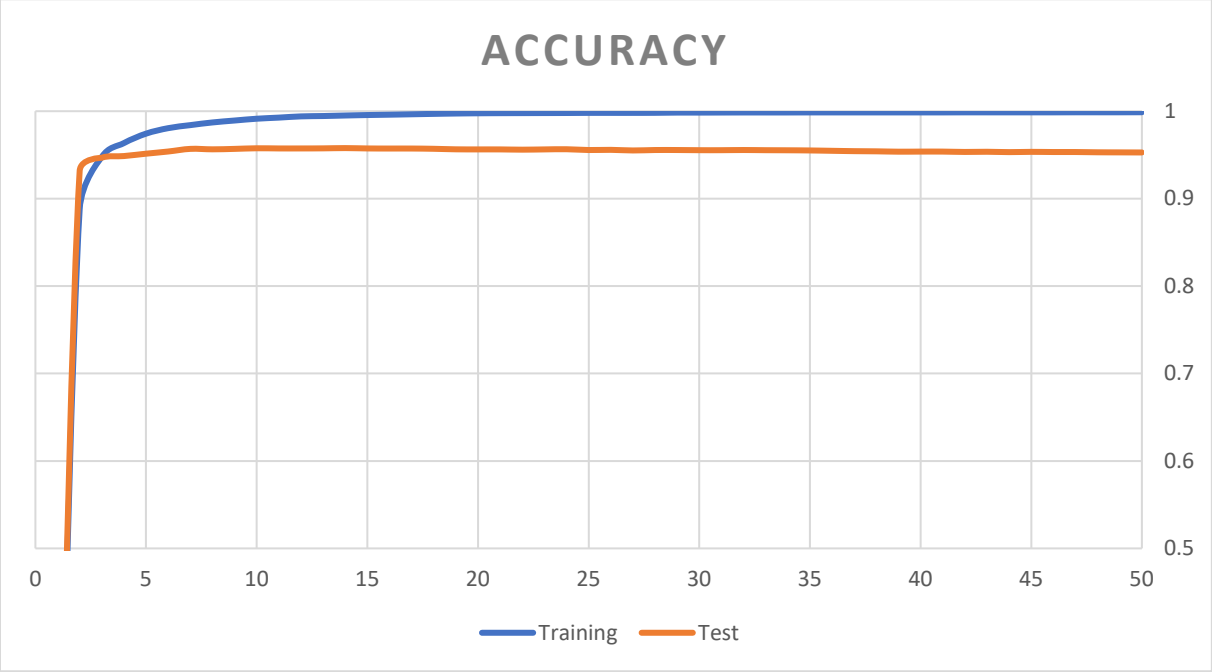
to guessing 8 or 9 when momentum is 0.90 in comparison to the other three, which seems to show neural networks made fair guesses (ex. Confusing 3 with an 8).

Experiment 3

The purpose of experiment 3 is to vary the number of training data the neural network is trained on. The varying number of training data was a quarter of the data (15,000 samples) and half of the data (30,000 samples). Momentum was kept constant at 0.9 and learning rate was kept at 0.1 for both cases. All other details remain same throughout the experiment.

Data

Accuracy per Epoch Graph for quarter of training data



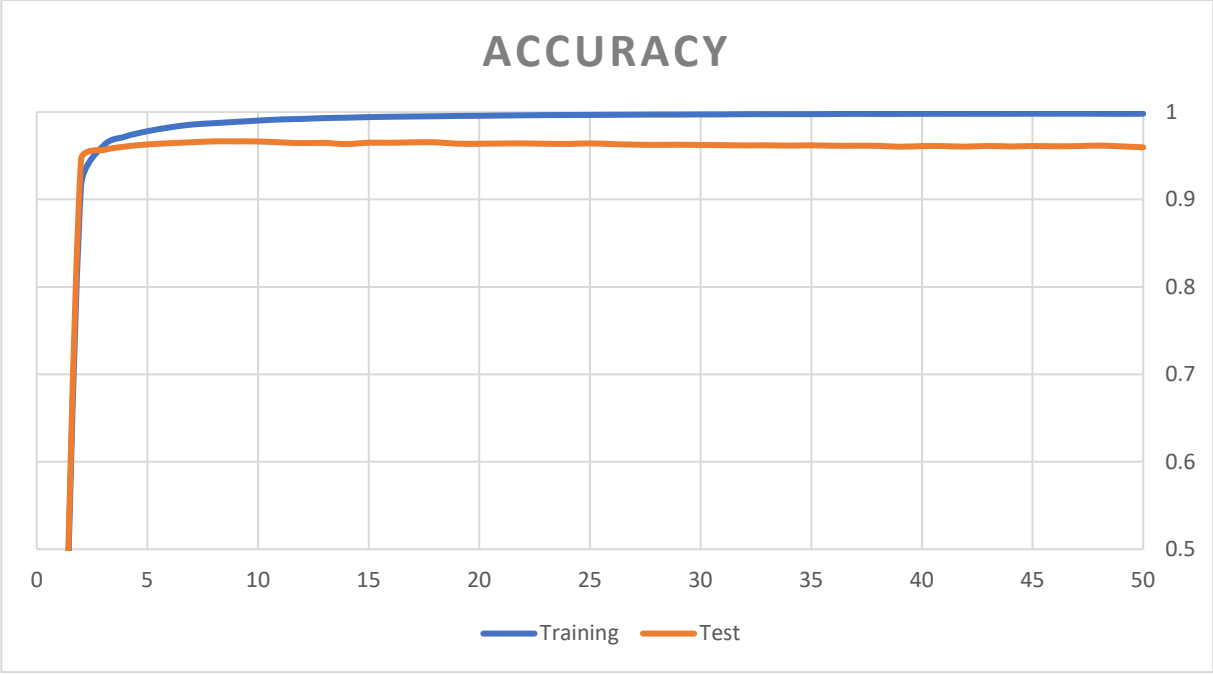
Comment: Epoch 0 had initial accuracy of 0.11266666666666666. After 1 epoch, training accuracy jumped to 0.8877333333333334. Final accuracy for training was 0.9988666666666667and for testing was 0.9521.

Confusion Matrix after 50 epochs of test for quarter of training data (in 10^2)

	0	1	2	3	4	5	6	7	8	9
0	481.99	0	0.66	0.09	0.08	0.72	1.99	0.94	2.67	0.86
1	0.35	559.31	2.18	0.23	0.01	0.77	1.11	0.52	3.02	0
2	2.77	0.40	492.70	1.74	1.32	0.59	2.49	3.78	8.70	1.51
3	0.25	0.01	4.65	478.50	0.77	3.23	0.66	2.30	12.13	2.50
4	0.53	0.40	2.30	0	458.86	0	6.05	0.08	2.25	20.53
5	2.95	0.44	1.20	12.37	1.53	408.62	3.39	0.94	11.13	3.43
6	3.59	1.50	2.45	0.10	3.20	3.81	456.73	0.28	6.75	0.59
7	0.78	3.28	5.54	2.04	1.29	0	0.01	490.54	2.50	8.02
8	3.36	0.10	1.61	2.30	1.97	2.45	1.47	1.45	468.92	3.37
9	2.06	2.10	1.10	2.80	5.95	0.93	0.03	3.10	10.34	476.09

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused the most.

Accuracy per Epoch Graph for half of training data



Comment: Epoch 0 had initial accuracy of 0.09753333333333333. After 1 epoch, training accuracy jumped to 0.9153333333333333. Final accuracy for training was 0.9979666666666667 and for testing was 0.9598. While training accuracy stably increased with minimal oscillation, some oscillation can be seen for testing accuracy.

Confusion Matrix after 50 epochs of Test for half of training data (in 10^2)

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	483.20	0.31	1.19	0.43	0.44	1.04	0.74	0.54	1.56	0.55
1	0.64	556.20	1.62	1.24	0.10	0.98	1.37	1.12	3.92	0.31
2	4.04	0.31	486.11	2.82	2.60	0.54	1.90	7.35	8.12	2.21
3	0.49	0.27	2.61	484.73	0.24	4.97	0.02	2.71	5.78	3.18
4	0.76	0.25	0.21	0.17	476.98	0	2.74	1.15	0.62	8.12
5	3.67	0.45	0.30	7.56	1.83	420.79	2.52	1.17	5.10	2.61
6	5.94	1.28	0.10	0.27	2.82	3.28	459.30	0.03	5.41	0.57
7	0.27	2.32	3.54	1.54	2.72	0.02	0	495.00	2.41	6.18
8	2.68	0.50	0.65	2.50	2.49	2.41	1.30	2.78	468.87	2.82
9	2.20	2.71	0	3.77	6.50	1.15	0.20	3.39	4.30	480.28

Green indicates correct answers by the neurons. Yellow is the prediction the neurons confused most.

Discussion

Both networks do not perform badly and reach around 0.95-0.96 test accuracies but does worse than the network trained on the full data set.

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

For quarter of data, the training accuracy was 0.9988666666666667 and the testing accuracy was 0.9521. For half of the data, the training accuracy was 0.9979666666666667 and the testing accuracy was 0.9598. As a comparison, final accuracy for training was 0.99565 and for testing was 0.9652 for full data sample. Therefore, testing accuracy was higher with half the data than with quarter data, which makes sense since there were more samples for the network to train on so the accuracy would be expected to be higher.

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

Epoch 1 accuracy was 0.8877333333333334 for the quarter data, 0.9153333333333333 for the half data, 0.93505 for full training data. Looking at the data points from Epoch 1, the results seem to indicate that more sample data seems to mean faster convergence.

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

Because both graphs show that the test accuracy are not oscillating much, I don't think there is overfitting for the half and quarter data. I think the low accuracy is due to not enough training samples and not due to overfitting, as test accuracies increase as more data samples are provided in this case.

In conclusion, the number of neurons in the hidden layer, momentum, and data size all have an impact on the test accuracy, and there needs to be a balance between too much and too little for all three.