Nick Smith
1/19/2019

CS 445 - Machine Learning
Homework 2: Neural Networks

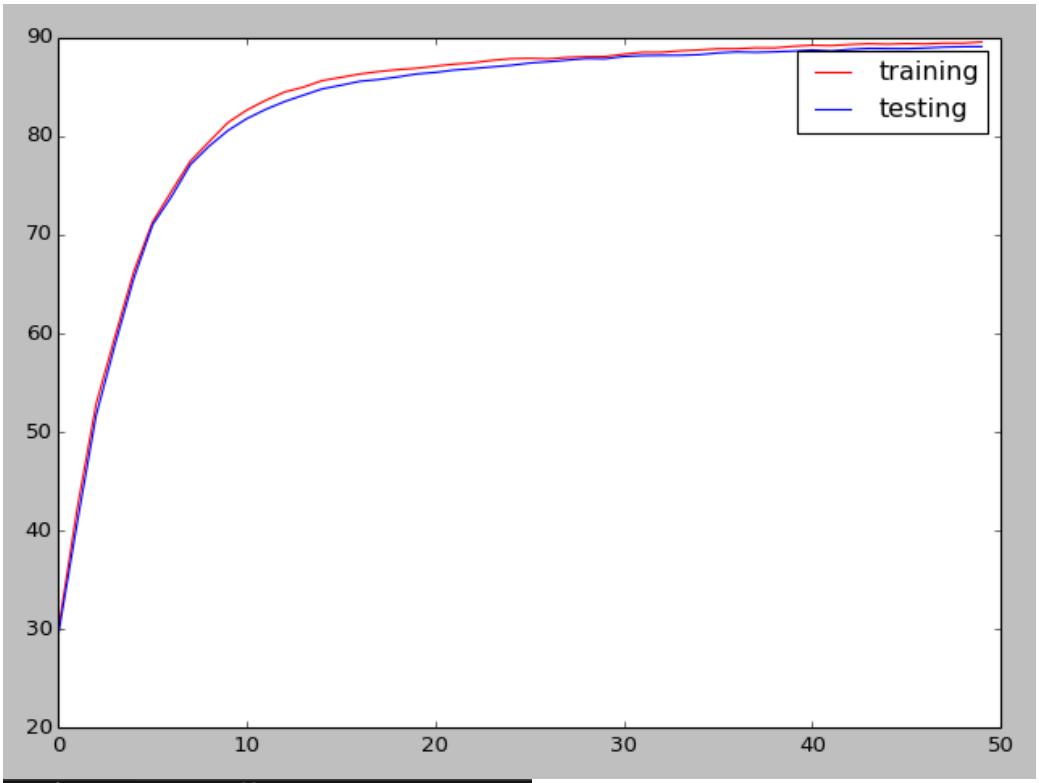
Description

In this experiment, we were tasked with updating our previous Perceptron program, and evolving it into a Neural Network. This Neural Network would still be similar to our previous Perceptron program, in that it would use Sigmoid Activations and Stochastic Gradient Descent to classify the digits, but we were to implement the Hidden Layer to convert the single Neuron into a Network. This included implementing back and forward propagation methods and introducing Momentum to our equation.

Experiment 1: Hidden Units

- **Learning Rate = .1, Momentum = .9, Epochs = 50**

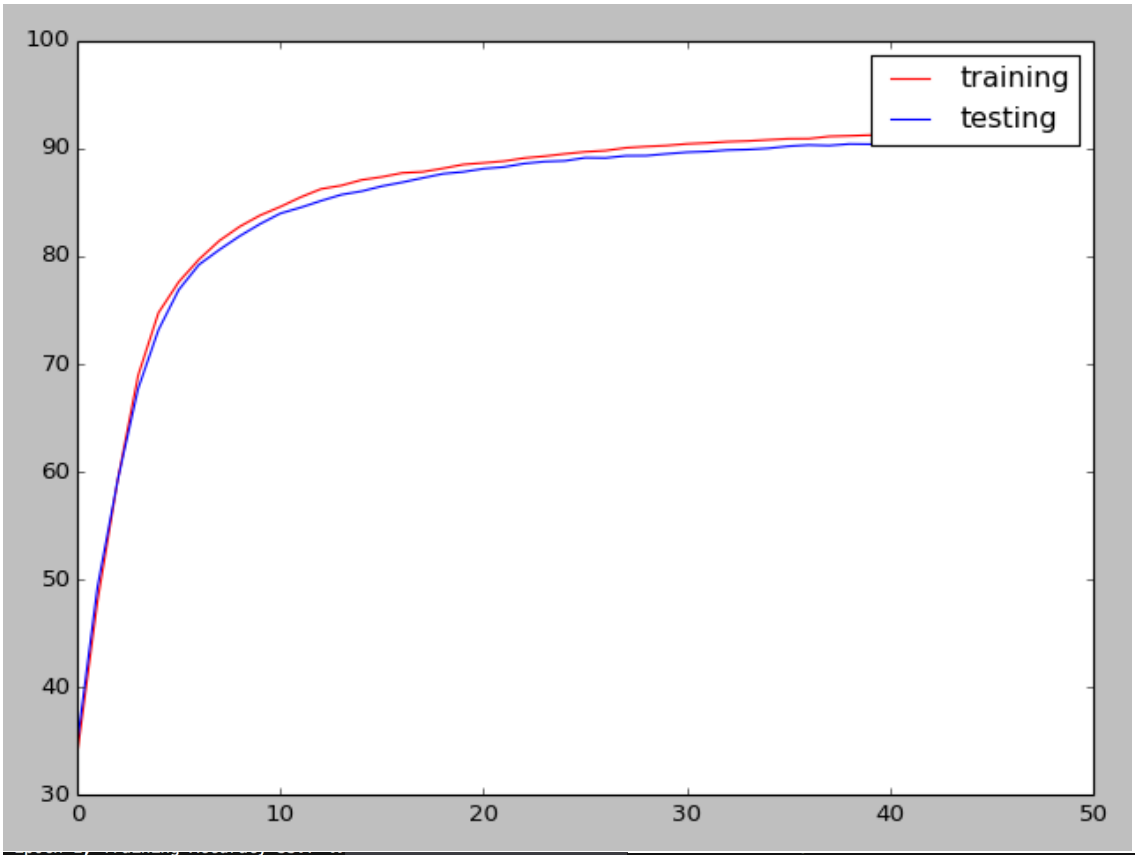
n=10




```
Epoch 1, Test Accuracy 29.5 %.
[[399.  15. 186.  91.  45. 156. 128.  59. 131.  19.]
 [ 92. 974. 223. 145. 166.  77. 150. 136. 189. 178.]
 [236.  25. 166. 325.  60. 160. 144.  15. 100.  19.]
 [  1.   0.   0.   8.   3.   5.   2.   9.   0.  10.]
 [  9.  18.  55.   1.  34.   3.  94.   7.   3.  12.]
 [  1.  17.   6.   0.  17.  16.   1.   1.   0.   0.]
 [178.  70. 294. 132. 118.  69. 360.  20. 200.  56.]
 [ 42.   8.  85. 205. 349. 279.  64. 679. 298. 401.]
 [  0.   0.   0.   2.   1.   1.   0.   7.   1.   1.]
 [ 22.   8.  17. 101. 189. 126.  15.  95.  52. 313.]]
Epoch 2, Training Accuracy 42.24 %.
Epoch 2, Test Accuracy 40.82 %.
```

```
Epoch 50, Test Accuracy 89.1 %.
[[ 928.    1.   19.    8.    3.   23.   28.    2.   13.   16.]
 [   0. 1099.    8.    1.    0.    6.    6.   21.   19.    4.]
 [   2.    6.  900.   30.    6.    7.    5.   26.    8.    1.]
 [  12.    5.   25.  880.    0.   50.    1.    6.   43.    6.]
 [   3.    1.   16.    0.  899.    6.   12.   10.    9.   61.]
 [  14.    0.    1.   47.    0.  701.   13.    3.   23.   20.]
 [  11.    2.   12.    0.   21.   20.  883.    1.   18.    1.]
 [   2.    3.   19.   14.    6.   15.    0.  930.    4.   30.]
 [   5.   18.   22.   23.   10.   60.   10.    2.  825.    5.]
 [   3.    0.   10.    7.   37.    4.    0.   27.   12.  865.]]
```

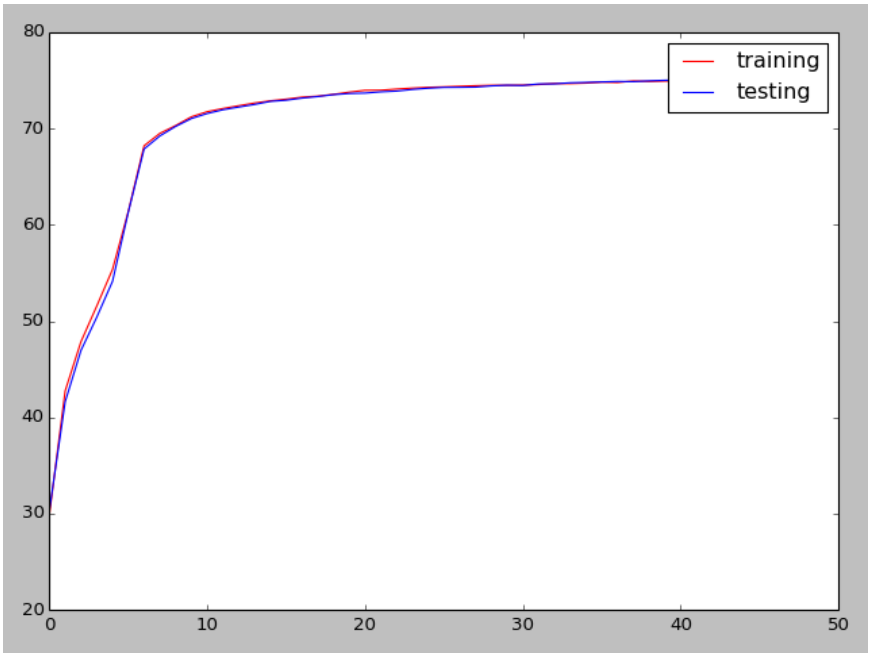
n=20




```
Epoch 1, Test Accuracy 34.71 %.
[[ 804.    3.  227.   55.  110.  119.  117.   47.  138.   46.]
 [  11. 1037.  266.  189.   74.  229.   80.  388.  204.  139.]
 [  13.    0.    2.    4.   11.    4.   12.    5.    3.    8.]
 [  13.   78.  119.  562.   61.  206.   31.  193.  198.  186.]
 [   9.    0.    8.    0.    9.    1.    2.    6.    1.    2.]
 [  28.    3.   39.   34.   90.   96.   45.   15.  138.   45.]
 [  83.    7.  287.  104.  331.  159.  613.  140.  189.  243.]
 [   7.    5.   20.   30.  113.   42.   14.  185.   58.  180.]
 [   1.    0.   21.    4.   14.    7.    7.   15.   14.   11.]
 [  11.    2.   43.   28.  169.   29.   37.   34.   31.  149.]]
```

```
Epoch 50, Test Accuracy 90.93 %.
[[ 950.    0.   16.    3.    2.   18.   22.    6.    7.    8.]
 [   0. 1100.    3.    0.    2.    3.    3.    7.    4.    5.]
 [   2.    3.  903.   29.    4.    3.    5.   24.    7.    2.]
 [   2.    5.   22.  898.    0.   38.    1.    6.   30.   14.]
 [   0.    1.   17.    1.  923.   14.   13.   11.   16.   53.]
 [   7.    4.    3.   30.    1.  765.   20.    7.   15.    8.]
 [   9.    4.   15.    3.   15.   20.  884.    0.   16.    4.]
 [   4.    1.   20.   14.    1.    7.    3.  946.    8.   28.]
 [   6.   16.   28.   24.    6.   16.    7.    9.  852.   15.]
 [   0.    1.    5.    8.   28.    8.    0.   12.   19.  872.]]
```

n=100




```
Epoch 1, Test Accuracy 30.36 %.
[[  4.  24.   6.  12.   3.  32.  15.   0.  28.   0.]
 [  7.   4.   4.  15.   1.   8.   0.   7.  16.   4.]
 [267. 382. 629. 160.  19. 133. 326.  56. 171.  38.]
 [317. 403. 149. 599.  57. 349. 121. 130. 301.  73.]
 [108.  28.  89.  22. 754. 130. 241. 110. 150. 334.]
 [  9.  73.  22.   9.   9.  22.   3.   6.  11.   3.]
 [124.  23.  43.  44.  26.  53. 118.  20.  46.  23.]
 [ 89. 162.  59.  65.  40.  77.  36. 589.  96. 231.]
 [ 41.  22.   4.  10.   3.  23.   8.   3.  15.   1.]
 [ 14.  14.  27.  74.  70.  65.  90. 107. 140. 302.]]
```

```
Epoch 50, Test Accuracy 75.2 %.
[[   0.    0.    0.    0.    0.    0.    0.    0.    0.    0.]
 [   0. 1121.    1.    1.    1.    0.    2.   11.   33.    6.]
 [ 192.    2.  964.   22.    8.    4.    8.   25.  130.    2.]
 [  97.    2.   15.  933.    4.   29.    2.    5.  223.   21.]
 [   9.    0.   11.    0.  918.    7.   13.    8.   68.   41.]
 [ 490.    1.    8.   21.    2.  814.   15.    0.  233.   15.]
 [ 119.    7.    9.    4.   16.   19.  913.    2.   65.    2.]
 [  39.    2.   21.   14.    3.   15.    3.  953.   32.   18.]
 [   2.    0.    0.    0.    0.    0.    0.    0.    0.    0.]
 [  32.    0.    3.   15.   30.    4.    2.   24.  190.  904.]]
```

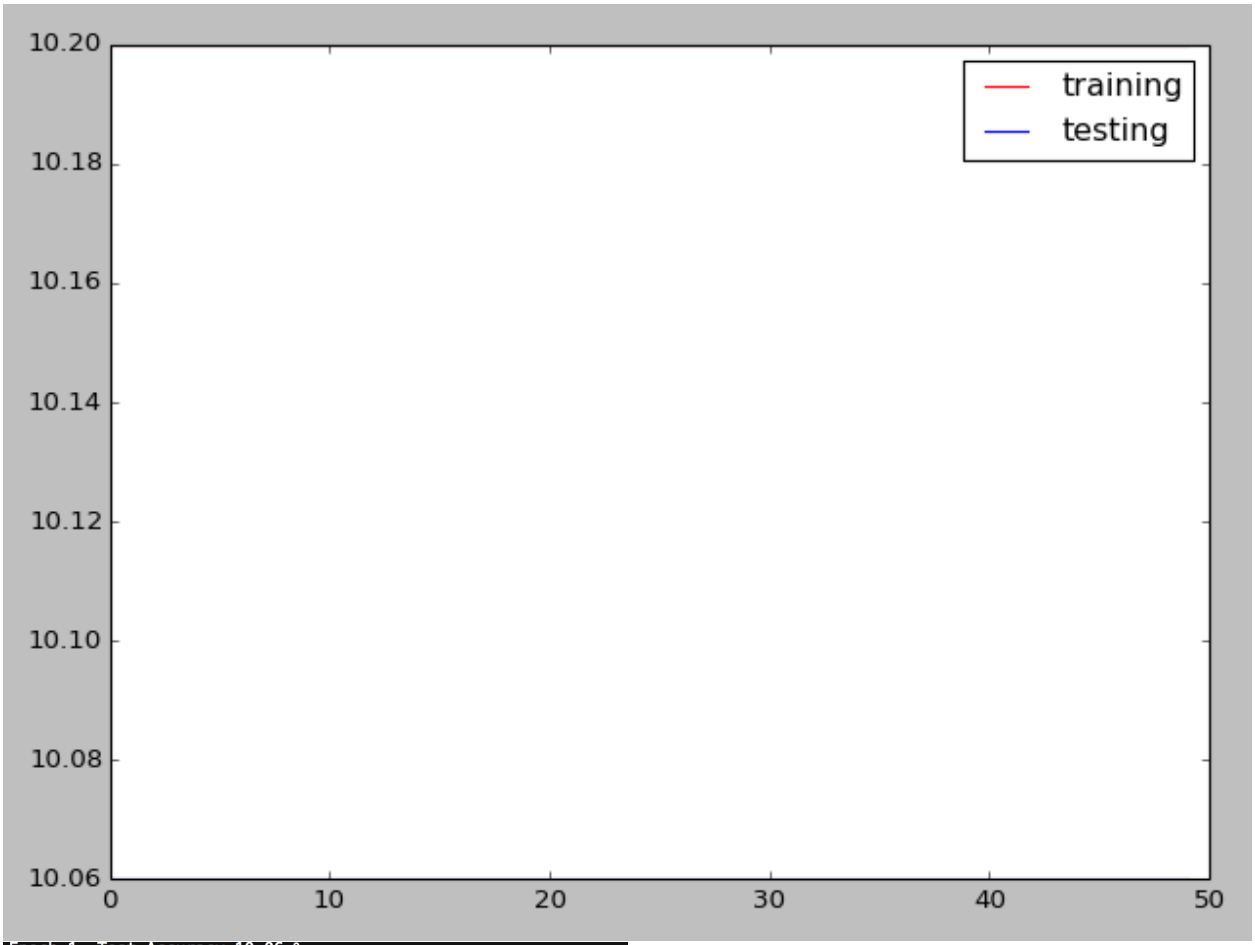
Review

The difference between 10 and 20 hidden units was negligible in this case, with both starting out and ending at roughly the same accuracy, and both experiments had their data sets converging around ten epochs. However, when the hidden units were cranked up to 100, the accuracy plateaued dramatically early compared to the previous two iterations. Instead of the accuracy of the previous two, which were both around 90%, the 100 node iteration converged around 70%, which implies some data over-fitting. Upon viewing the confusion matrix, we can see that 0's and 8's were all but neglected. That being said, these three iterations fared much better than my Perceptron, which topped out at 50% and completed the task in a fraction of the time, even with 100 hidden nodes.

Experiment 2: Varied Momentum

- **Hidden Units = 100, Epochs = 50**

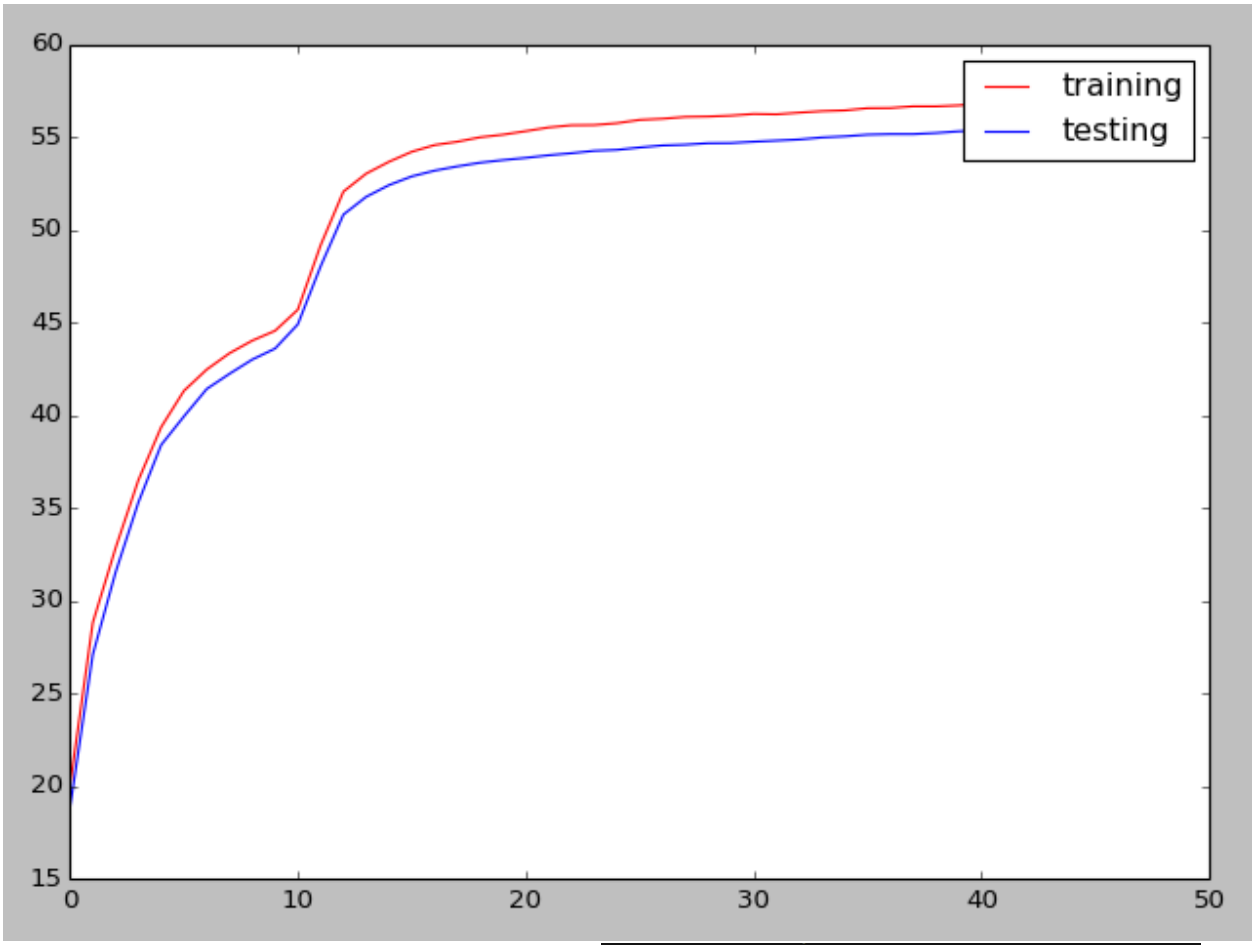
Momentum = 0.0




```
Epoch 1, Test Accuracy 10.06 %.
[[   0.    0.    2.    0.    8.    0.    3.    4.    0.    7.]
 [  35.    8.   14.   37.   18.   33.   25.   49.   45.   68.]
 [   6.    7.   14.    8.   42.   14.    6.   31.    4.   49.]
 [ 158.   63.  205.  154.  175.  185.   34.  106.  280.   41.]
 [  17.    0.   54.   15.   85.    8.  144.   62.    5.   99.]
 [ 141. 1056.  628.  665.  346.  403.  272.  175.  407.  253.]
 [   0.    0.    0.    0.    0.    0.    0.    0.    0.    0.]
 [   0.    0.    0.    0.    0.    0.    0.    0.    0.    0.]
 [  36.    1.   33.   33.  120.   36.  128.  175.   67.  217.]
 [ 587.    0.   82.   98.  188.  213.  346.  426.  166.  275.]]
```

```
Epoch 50, Test Accuracy 10.06 %.
[[   0.    0.    2.    0.    8.    0.    3.    4.    0.    7.]
 [  35.    8.   14.   37.   18.   33.   25.   49.   45.   68.]
 [   6.    7.   14.    8.   42.   14.    6.   31.    4.   49.]
 [ 158.   63.  205.  154.  175.  185.   34.  106.  280.   41.]
 [  17.    0.   54.   15.   85.    8.  144.   62.    5.   99.]
 [ 141. 1056.  628.  665.  346.  403.  272.  175.  407.  253.]
 [   0.    0.    0.    0.    0.    0.    0.    0.    0.    0.]
 [   0.    0.    0.    0.    0.    0.    0.    0.    0.    0.]
 [  36.    1.   33.   33.  120.   36.  128.  175.   67.  217.]
 [ 587.    0.   82.   98.  188.  213.  346.  426.  166.  275.]]
```

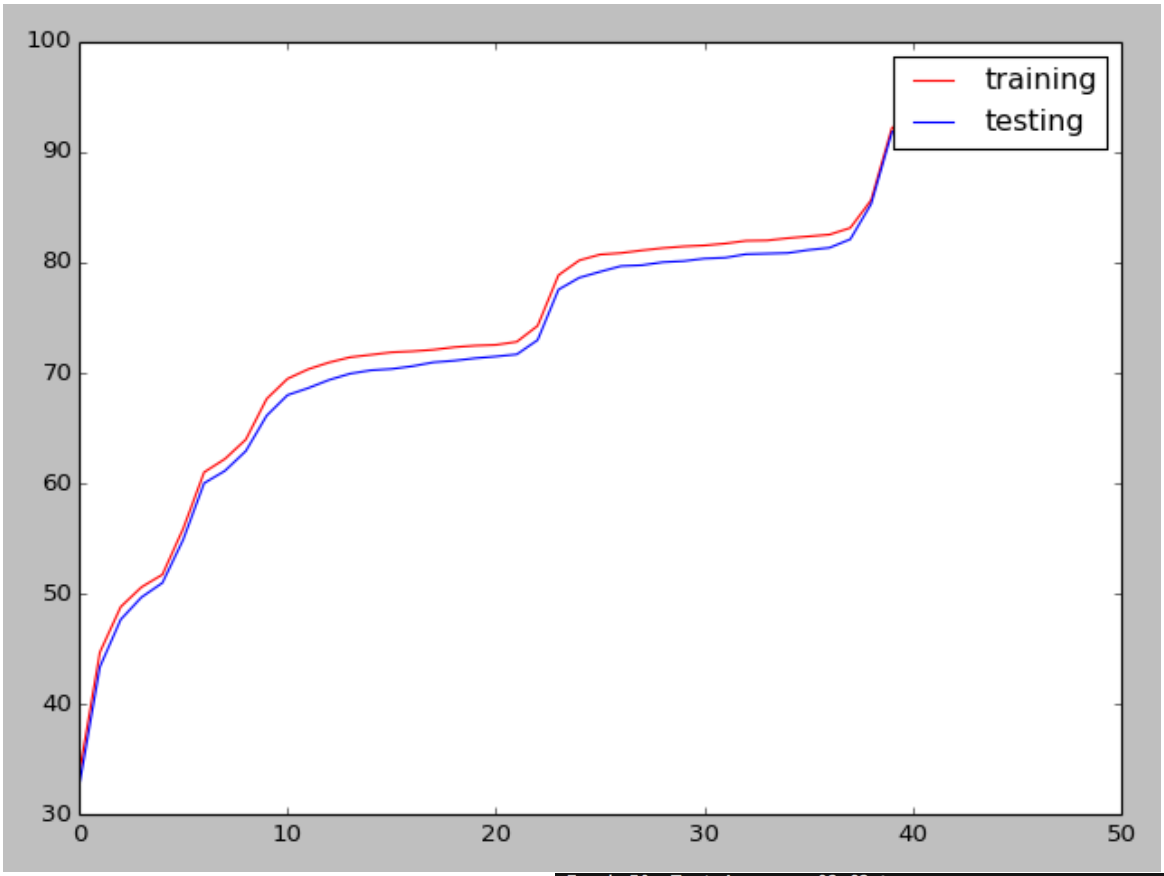
Momentum = 0.5




```
Epoch 1, Test Accuracy 18.85 %.
[[  7.   1.   4.   6.  16.   8.  13.  22.   3.  10.]
 [ 39.   3.  21.  11.  17.   8.  16.   5.   5.   9.]
 [237. 239. 508. 217. 134. 265. 156.  73. 188. 101.]
 [ 34.   9.  15.  86.  55.  52.  32. 103.  10.  39.]
 [305.  88.  64.  90. 431. 207. 136. 345. 150. 439.]
 [  1.  14.  38.  45.  11.   1.   2.   8.   7.   6.]
 [155. 516. 299. 148. 164. 118. 563. 139. 282. 223.]
 [ 20.  86.  14. 166.  24.  20.  12.  18.  40.  39.]
 [176. 115.  46. 205. 107. 200.  23. 302. 255. 130.]
 [  6.  64.  23.  36.  23.  13.   5.  13.  34.  13.]]
```

```
Epoch 50, Test Accuracy 55.65 %.
[[  2.   0.   0.   0.   0.   3.   0.   0.   0.   0.]
 [  3.   0.   0.   0.   0.   0.   0.   0.   0.   0.]
 [158. 236. 921.  30.   5.  53.   8.  32.  16.  19.]
 [162. 389.  26. 937.   7. 280.   4.  11.  29.  98.]
 [ 14.   6.  12.   0. 926.  92.  15.  11.  13. 446.]
 [  0.   1.   0.   0.   0.   0.   0.   0.   0.   0.]
 [157.  46.   9.   2.  18.  64. 916.   0.  12.  22.]
 [123. 174.  26.  19.  10.  71.   4. 968.   9. 258.]
 [359. 281.  38.  22.  16. 329.  10.   6. 895. 166.]
 [  2.   2.   0.   0.   0.   0.   1.   0.   0.   0.]]
```

Momentum = 1.0




```
Epoch 1, Test Accuracy 32.46 %.
[[807. 146. 162.  73.  98. 174. 267.  32. 180. 105.]
 [  0.  17.  13.   5.  28.  10.  70.   2.  11.  23.]
 [ 35. 262. 374.  32. 210.  86. 155.  26.  95. 113.]
 [ 31. 246. 300. 827.  69. 288.  90.  46. 323. 140.]
 [  0.  80.  39.   3.  44.   8. 124.   7.  40.  30.]
 [ 44.  72.  12.  25. 195. 196.  32.  37.  74. 201.]
 [  1.   0.   1.   0.   1.   1.   8.   0.   0.   0.]
 [ 51. 194. 104.  38. 251.  97. 163. 859. 178. 284.]
 [  6.   7.   1.   1.  45.  10.   1.   6.  11.  10.]
 [  5. 111.  26.   6.  41.  22.  48.  13.  62. 103.]]
```

```
Epoch 50, Test Accuracy 92.63 %.
[[ 955.    0.   12.    2.    2.    9.   11.    3.    4.    9.]
 [   0. 1110.    3.    0.    3.    4.    2.    9.    5.    5.]
 [   2.    5.  946.   26.   10.    6.   11.   26.   10.    1.]
 [   3.    5.   13.  922.    4.   22.    2.    8.   18.   13.]
 [   0.    1.   12.    1.  908.    6.    7.    7.    7.   39.]
 [   5.    0.    1.   27.    2.  797.   13.    2.   23.   14.]
 [   8.    4.   11.    4.    8.   16.  907.    0.   10.    3.]
 [   2.    1.   12.   10.    2.    7.    0.  940.    6.   14.]
 [   4.    9.   19.   13.   11.   17.    5.    6.  879.   12.]
 [   1.    0.    3.    5.   32.    8.    0.   27.   12.  899.]]
```

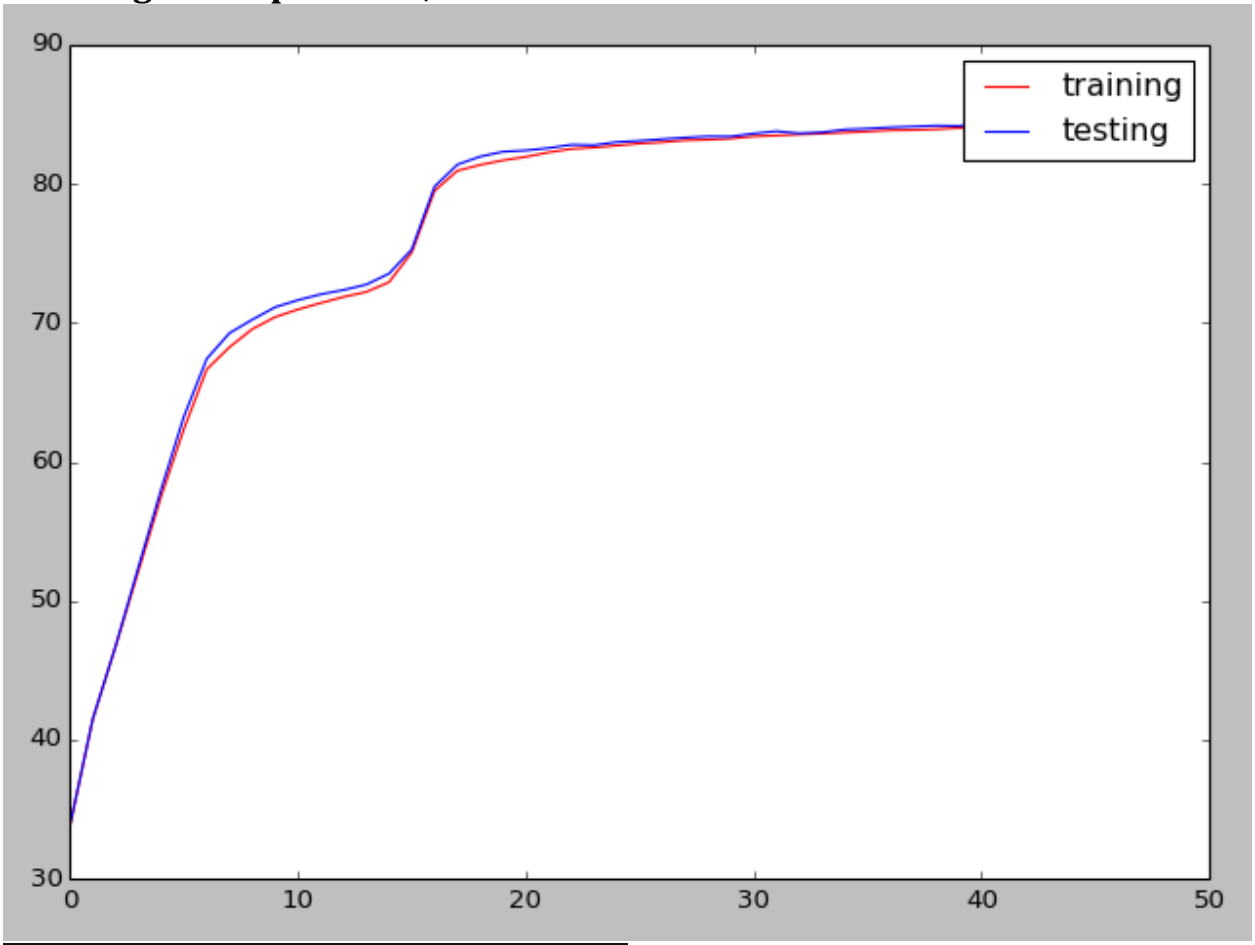
Review

For momentum 0.0, the algorithm gave us a repeated 10% accuracy. This makes sense as the 0 momentum negates any change the algorithm would make. With momentum 0.5, we saw the training and the test data diverge with each epoch, as well as some overfitting, with 0, 5, and 9 almost being omitted from the confusion matrix. With momentum 1, the results resembled that of the previous iterations. With the higher moments, we were able to get similar accuracy to the initial 100 hidden units, but still it also came with the same overfitting issues in the early epochs, with 1 and 4 being improperly classified. However, the graph shows the moments when the accuracies jumped, implying that it learned to correctly classify those digits.

Experiment 3: Varied Training Examples

- **Hidden Units = 100, Momentum = 0.9**

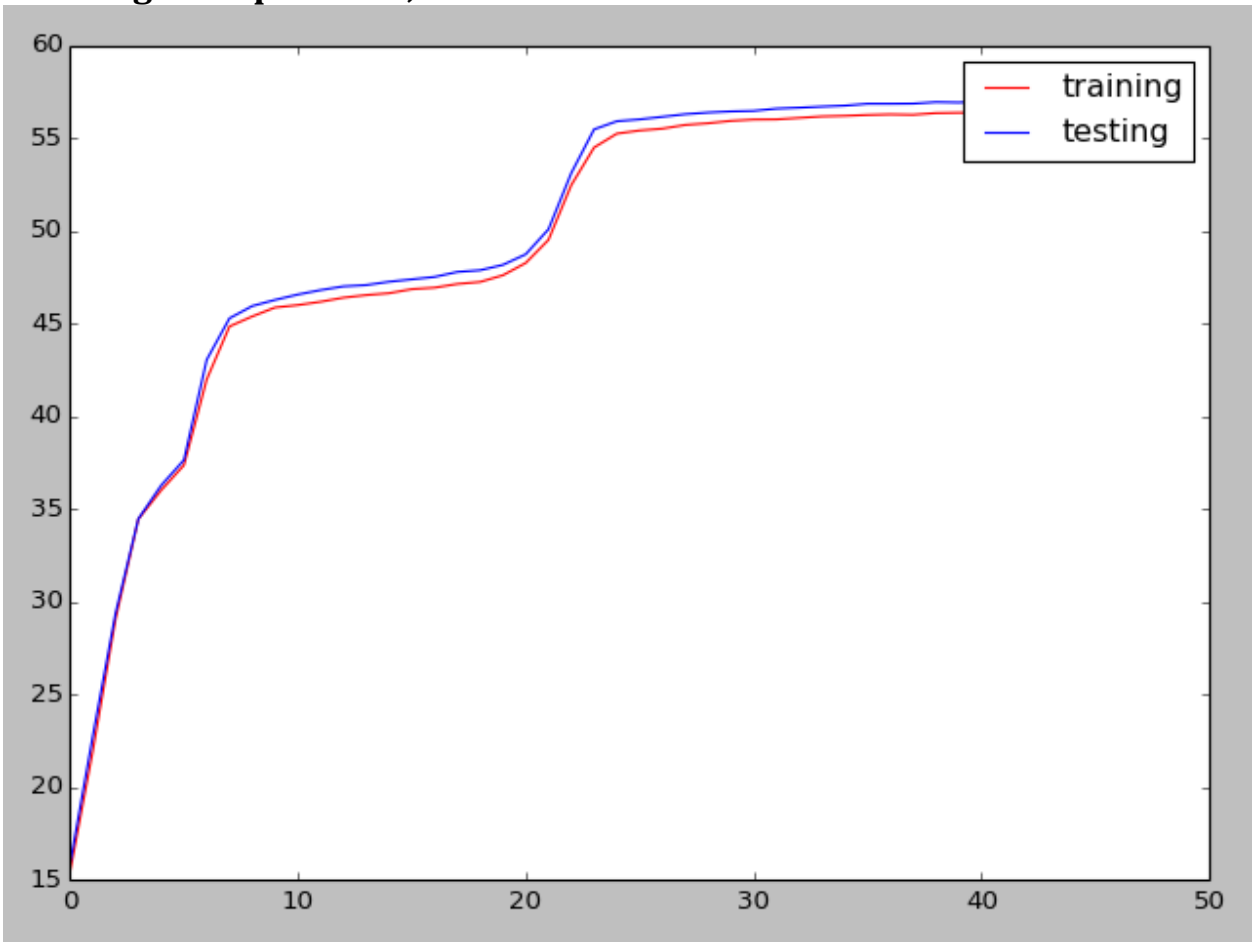
Trainings examples = 30,000




```
Epoch 1, Test Accuracy 33.89 %.
[[ 841.    7.  247.  343.   24.  295.  179.  155.  310.   94.]
 [   2. 1055.  203.  225.   41.   55.   24.  286.  158.   65.]
 [  11.    0.   15.    9.    4.   16.    5.    3.    0.    5.]
 [   0.    0.   17.   14.    4.    8.   10.   79.    5.    4.]
 [  13.   46.  244.  146.  778.  168.  186.  281.  293.  668.]
 [  36.    0.   24.  143.   10.  178.   12.   50.   47.   69.]
 [  64.   24.  183.   63.  117.  124.  483.  121.  138.   81.]
 [   1.    1.   19.   12.    0.   11.    5.    4.    1.    1.]
 [   5.    2.   26.   42.    2.   12.    5.   13.    2.    3.]
 [   7.    0.   54.   13.    2.   25.   49.   36.   20.   19.]]
```

```
Epoch 50, Test Accuracy 84.65 %.
[[ 963.    0.   12.    2.    1.    9.   11.    4.   61.   16.]
 [   0. 1117.    4.    1.    2.    1.    2.    8.   25.    5.]
 [   4.    4.  955.   18.    7.    4.   11.   25.  154.    3.]
 [   2.    4.   12.  942.    0.   35.    1.    6.  194.   14.]
 [   0.    1.   14.    2.  935.    7.   12.    4.   85.   33.]
 [   4.    4.    3.   24.    1.  800.   13.    5.  259.    9.]
 [   5.    4.   10.    4.    8.   19.  907.    0.   47.    2.]
 [   1.    1.   19.    8.    0.   10.    1.  949.   25.   30.]
 [   0.    0.    0.    0.    0.    0.    0.    0.    0.    0.]
 [   1.    0.    3.    9.   28.    7.    0.   27.  124.  897.]]
```

Training examples = 15,000



Review

With 30,000 training examples, the algorithm completely failed to identify a single 8, instead mistaking them the most for 5's and generating a respectable 84% accuracy. With 15,000 results, the algorithm did not fare so well. The inability to properly train the weights shows when the training images are reduced (as expected). However, in this case, it mean that the plateau happened in much earlier in the epochs, then it jumped (learned a new digit) and then plateaued again. With 15,000 I think the algorithm was underfit instead of over fit.

Conclusion

The number of hidden nodes, the momentum, and number of training examples all have their ideal values. With the hidden nodes, more does not necessarily mean better, and is this case it only meant slower. The moment um of .9 worked the best, and the more training examples the better the algorithm performs.