

Assignment 4 Report

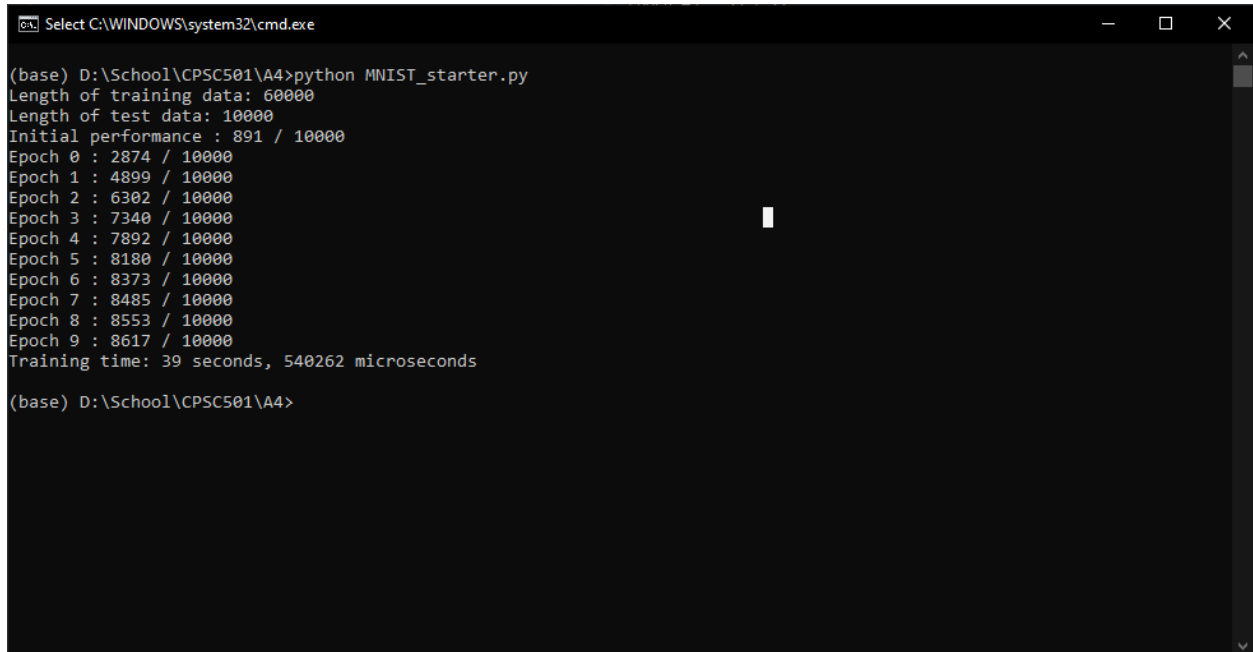
By: Braydon Pacheco (30100449)

The project can be accessed by going to the repository page at:

https://github.com/pacheeko/CPSC_501_A4

At the time of me finishing this report, gitlab is down, so I have included my project github page. You can email me at braydon.pacheco@ucalgary.ca with your github username or email to gain access to the repository.

Part 1



```

Select C:\WINDOWS\system32\cmd.exe

(base) D:\School\CPSC501\A4>python MNIST_starter.py
Length of training data: 60000
Length of test data: 10000
Initial performance : 891 / 10000
Epoch 0 : 2874 / 10000
Epoch 1 : 4899 / 10000
Epoch 2 : 6302 / 10000
Epoch 3 : 7340 / 10000
Epoch 4 : 7892 / 10000
Epoch 5 : 8180 / 10000
Epoch 6 : 8373 / 10000
Epoch 7 : 8485 / 10000
Epoch 8 : 8553 / 10000
Epoch 9 : 8617 / 10000
Training time: 39 seconds, 540262 microseconds

(base) D:\School\CPSC501\A4>
```

Figure 1. Initial neural net

```
C:\WINDOWS\system32\cmd.exe

(base) D:\School\CPSC501\A4>python MNIST_starter.py
Length of training data: 60000
Length of test data: 10000
Initial performance : 859 / 10000
Epoch 0 : 4061 / 10000
Epoch 1 : 5289 / 10000
Epoch 2 : 6465 / 10000
Epoch 3 : 7297 / 10000
Epoch 4 : 7839 / 10000
Epoch 5 : 8136 / 10000
Epoch 6 : 8299 / 10000
Epoch 7 : 8432 / 10000
Epoch 8 : 8511 / 10000
Epoch 9 : 8576 / 10000
Epoch 10 : 8626 / 10000
Epoch 11 : 8675 / 10000
Epoch 12 : 8728 / 10000
Epoch 13 : 8777 / 10000
Epoch 14 : 8817 / 10000
Epoch 15 : 8816 / 10000
Epoch 16 : 8843 / 10000
Epoch 17 : 8876 / 10000
Epoch 18 : 8885 / 10000
Epoch 19 : 8910 / 10000
Training time: 85 seconds, 874506 microseconds

(base) D:\School\CPSC501\A4>
```

Figure 2. Initial neural net using 20 epochs.

```
C:\WINDOWS\system32\cmd.exe

(base) D:\School\CPSC501\A4>python MNIST_starter.py
Length of training data: 60000
Length of test data: 10000
Initial performance : 1084 / 10000
Epoch 0 : 8978 / 10000
Epoch 1 : 9195 / 10000
Epoch 2 : 9199 / 10000
Epoch 3 : 9296 / 10000
Epoch 4 : 9340 / 10000
Epoch 5 : 9355 / 10000
Epoch 6 : 9408 / 10000
Epoch 7 : 9376 / 10000
Epoch 8 : 9440 / 10000
Epoch 9 : 9421 / 10000
Epoch 10 : 9447 / 10000
Epoch 11 : 9450 / 10000
Epoch 12 : 9478 / 10000
Epoch 13 : 9471 / 10000
Epoch 14 : 9493 / 10000
Epoch 15 : 9503 / 10000
Epoch 16 : 9497 / 10000
Epoch 17 : 9485 / 10000
Epoch 18 : 9537 / 10000
Epoch 19 : 9510 / 10000
Training time: 246 seconds, 948921 microseconds

(base) D:\School\CPSC501\A4>
```

Figure 3. Final neural net [784,20,10], epochs = 20, eta = 5, mini batch size = 20.

The initial neural net for part 1 had the neural net structure [784,10,10], it ran at 10 epochs, with a mini-batch size of 10 and a learning rate (eta) of 0.1. The accuracy of the final epoch for the initial neural net was 86.17% (Figure 1). My first instinct was to increase the number of epochs to 20, to see how this net will perform given more iterations to learn. This approach increased the accuracy to 89.10% (Figure 2), and the net accuracy seemed to peak around that point. Therefore, I decided next to increase eta, to ensure that the net was not landing in a local minimum for optimization. After experimenting with several values between 0.1 and 10, I landed at eta = 5 as the optimal learning rate. Then, I increased the mini-batch size because my computer can handle the increased processing requirements. This approach got me to around 94% accuracy. To get that last bit of accuracy, I increased the number of hidden nodes in the neural net to 20 to allow for more complexity in the functions that the neural net is processing. These hyperparameters led to the final epoch of my best neural net for part 1 to have an accuracy of 95.10% (Figure 3). The trained net can be found in the file 'part1.pkl'. The following is the code excerpts of me creating and training the net:

```
net = network.Network([784,20,10])  
  
net.SGD(trainingData, 20, 20, 5, test_data = testingData)
```

When later evaluating the test data on the trained net, the accuracy was 95.32%. The script `fine_failed_tests.py` found all the tests that the neural net failed. I have included three of them below with the image, the test index, the number guessed, and the actual number.

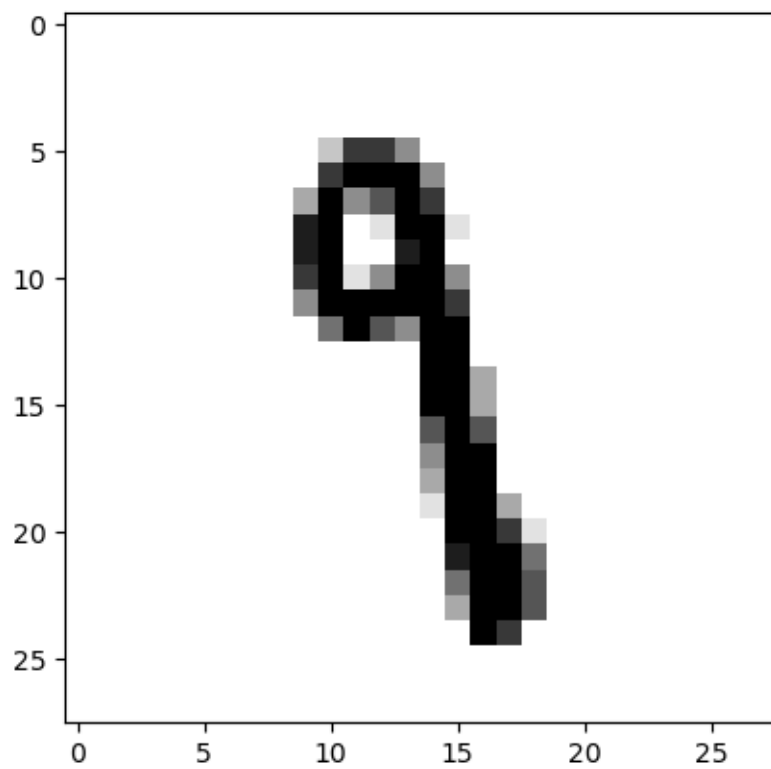


Figure 4. MNIST test data index 3503. The neural net guessed that this was a one. It is actually a nine.

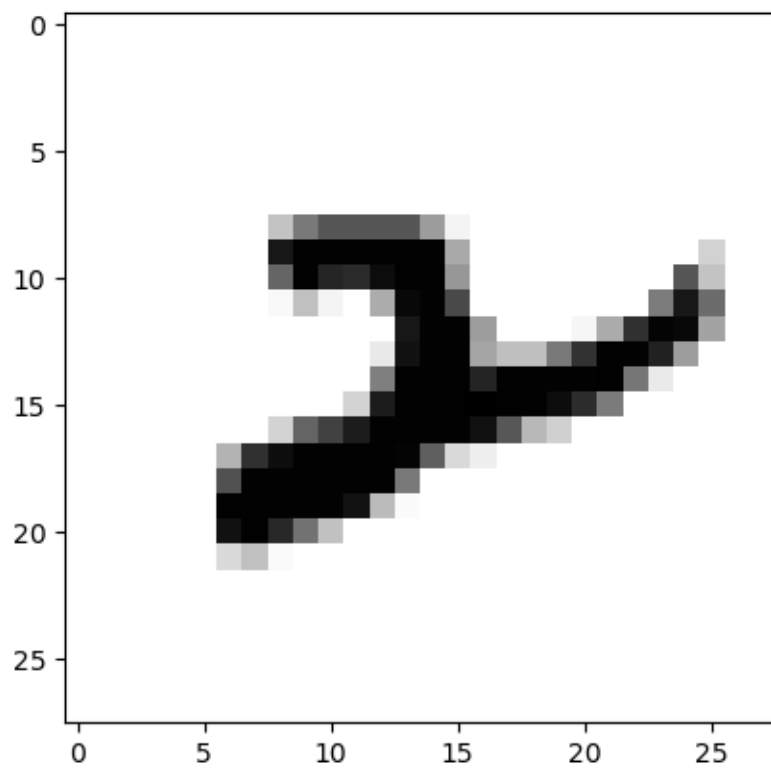


Figure 5. MNIST test data index 6744. The neural net guessed that this was a six. It is actually a two.

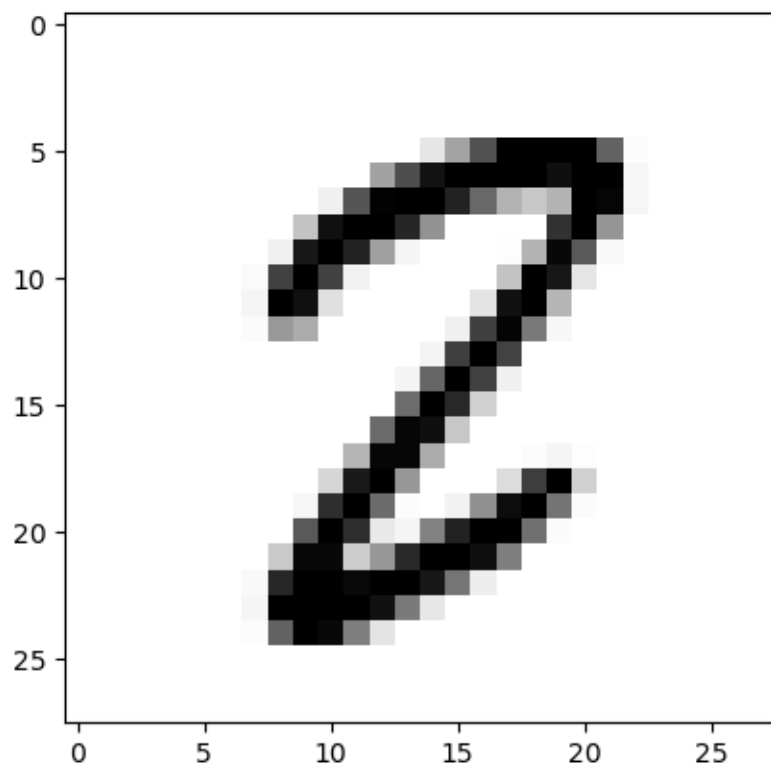


Figure 6. MNIST test data index 9893. The net guessed that this was an eight. It is actually a two.

Part 2

```
Select C:\WINDOWS\system32\cmd.exe

(base) D:\School\CPSC501\A4>python notMNIST_starter.py
Length of training data: 60000
Length of test data: 10000
Initial performance : 1047 / 10000
Epoch 0 : 8726 / 10000
Epoch 1 : 8847 / 10000
Epoch 2 : 8830 / 10000
Epoch 3 : 8840 / 10000
Epoch 4 : 8915 / 10000
Epoch 5 : 8911 / 10000
Epoch 6 : 8900 / 10000
Epoch 7 : 8913 / 10000
Epoch 8 : 8903 / 10000
Epoch 9 : 8920 / 10000
Training time: 27 seconds, 380823 microseconds

(base) D:\School\CPSC501\A4>
```

Figure 7. *notMNIST* initial neural net

```
C:\WINDOWS\system32\cmd.exe

(base) D:\School\CPSC501\A4>python notMNIST_starter.py
Length of training data: 60000
Length of test data: 10000
Initial performance : 1005 / 10000
Epoch 0 : 8827 / 10000
Epoch 1 : 8967 / 10000
Epoch 2 : 8954 / 10000
Epoch 3 : 8986 / 10000
Epoch 4 : 9035 / 10000
Epoch 5 : 9051 / 10000
Epoch 6 : 9077 / 10000
Epoch 7 : 9077 / 10000
Epoch 8 : 9101 / 10000
Epoch 9 : 9101 / 10000
Training time: 47 seconds, 285066 microseconds

(base) D:\School\CPSC501\A4>
```

Figure 8. Neural net with hidden layer containing 20 neurons added.

```
C:\WINDOWS\system32\cmd.exe
Epoch 3 : 9067 / 10000
Epoch 4 : 9082 / 10000
Epoch 5 : 9073 / 10000
Epoch 6 : 9064 / 10000
Epoch 7 : 9034 / 10000
Epoch 8 : 9104 / 10000
Epoch 9 : 9123 / 10000
Epoch 10 : 9127 / 10000
Epoch 11 : 9125 / 10000
Epoch 12 : 9135 / 10000
Epoch 13 : 9117 / 10000
Epoch 14 : 9107 / 10000
Epoch 15 : 9100 / 10000
Epoch 16 : 9100 / 10000
Epoch 17 : 9131 / 10000
Epoch 18 : 9111 / 10000
Epoch 19 : 9152 / 10000
Epoch 20 : 9118 / 10000
Epoch 21 : 9104 / 10000
Epoch 22 : 9116 / 10000
Epoch 23 : 9145 / 10000
Epoch 24 : 9091 / 10000
Epoch 25 : 9121 / 10000
Epoch 26 : 9108 / 10000
Epoch 27 : 9128 / 10000
Epoch 28 : 9102 / 10000
Epoch 29 : 9142 / 10000
Training time: 132 seconds, 702518 microseconds
(base) D:\School\CPSC501\A4>
```

Figure 9. Part 2, final neural net. Params changed: increased epochs to 30, mini-batch size to 20, and eta to 15.

The initial neural net for part 2 included the neural net structure [784,10], using 10 epochs, 10 data points per mini-batch, and $\eta = 10$ as the hyperparameters. This neural net was able to achieve an accuracy of 89.20% using the testing data (Figure 7), which is much higher than predicted in the assignment description (60%). Therefore, my goal was to increase this accuracy not just to above 90%, but to at least 92%. My first step was to introduce a hidden layer of neurons to help with the complexity of the functions that the net can use. This neural net achieved an accuracy of 91.01% (Figure 8). In order to increase this by an additional percentage point, I increased the number of epochs to 30, the mini-batch size to 20, and η to 15. This training took much longer than any previous training (132 seconds), and was only able to increase the accuracy to 91.42%. The trained net can be found in the file 'part2.pkl'. This accuracy did not meet my personal goal, but after trying several different hyperparameters this was the highest I could achieve. The time it took to train was a hinderance to how many different hyperparameters I could try. I believe that the reason the notMNIST data is much harder to get a high accuracy for than the MNIST data is because of the nature of the images. The numbers from 0-9 are more distinct than the letters a-j, and the letters have more ways that they can be drawn. This would make it much more likely for the neural net for the notMNIST data to overfit on the training data, since if it finds a letter that doesn't look like one that it's seen before, it will not guess correctly. Below is the code snippet of the hyperparameters for the final neural net:

```
net = network.Network([784,20,10])
net.SGD(trainingData, 30, 20, 15, test_data = testingData)
```



```
Select C:\WINDOWS\system32\cmd.exe

(base) D:\School\CPSC501\A4>python heart_starter.py
Column names are row.names, sbp, tobacco, ldl, adiposity, famhist, typea, obesity, alcohol, age, chd
Length of training data: 385
Length of test data: 77
Initial performance : 25 / 77
Epoch 0 : 23 / 77
Epoch 1 : 42 / 77
Epoch 2 : 46 / 77
Epoch 3 : 50 / 77
Epoch 4 : 50 / 77
Epoch 5 : 51 / 77
Epoch 6 : 54 / 77
Epoch 7 : 55 / 77
Epoch 8 : 55 / 77
Epoch 9 : 54 / 77
Training time: 0 seconds, 169905 microseconds

(base) D:\School\CPSC501\A4>
```

Figure 10. Part 3 initial neural net.

```
Select C:\WINDOWS\system32\cmd.exe

(base) D:\School\CPSC501\A4>python heart_starter.py
Column names are row.names, sbp, tobacco, ldl, adiposity, famhist, typea, obesity, alcohol, age, chd
Length of training data: 385
Length of test data: 77
Initial performance : 25 / 77
Epoch 0 : 53 / 77
Epoch 1 : 54 / 77
Epoch 2 : 55 / 77
Epoch 3 : 53 / 77
Epoch 4 : 53 / 77
Epoch 5 : 53 / 77
Epoch 6 : 55 / 77
Epoch 7 : 57 / 77
Epoch 8 : 53 / 77
Epoch 9 : 55 / 77
Epoch 10 : 52 / 77
Epoch 11 : 56 / 77
Epoch 12 : 56 / 77
Epoch 13 : 56 / 77
Epoch 14 : 58 / 77
Epoch 15 : 56 / 77
Epoch 16 : 57 / 77
Epoch 17 : 56 / 77
Epoch 18 : 57 / 77
Epoch 19 : 56 / 77
Training time: 0 seconds, 322999 microseconds

(base) D:\School\CPSC501\A4>
```

Figure 11. Part 3, intermediate neural net. Params changed: increased epochs to 20, mini-batch size to 20, eta to 1, and number of neurons in hidden later to 20.

```
C:\WINDOWS\system32\cmd.exe
Epoch 73 : 59 / 77
Epoch 74 : 58 / 77
Epoch 75 : 61 / 77
Epoch 76 : 60 / 77
Epoch 77 : 59 / 77
Epoch 78 : 60 / 77
Epoch 79 : 60 / 77
Epoch 80 : 58 / 77
Epoch 81 : 59 / 77
Epoch 82 : 61 / 77
Epoch 83 : 59 / 77
Epoch 84 : 60 / 77
Epoch 85 : 59 / 77
Epoch 86 : 59 / 77
Epoch 87 : 59 / 77
Epoch 88 : 60 / 77
Epoch 89 : 59 / 77
Epoch 90 : 60 / 77
Epoch 91 : 59 / 77
Epoch 92 : 59 / 77
Epoch 93 : 59 / 77
Epoch 94 : 59 / 77
Epoch 95 : 60 / 77
Epoch 96 : 60 / 77
Epoch 97 : 60 / 77
Epoch 98 : 60 / 77
Epoch 99 : 60 / 77
Training time: 1 seconds, 961260 microseconds
(base) D:\School\CPSC501\A4>
```

Figure 12. Part 3, final neural net. Params changed: increased epochs to 100, increased mini-batch size to 30, and decreased eta to 0.9.

The initial neural net for part 3 had the neuron structure of [9,10,2] with the training running for 10 epochs, with a mini-batch size of 10, and an eta of 0.1 (Figure 10). I used the first 385 data points as the training data, and the last 77 data points as my testing data. The initial run of training achieved an accuracy of 70.13%. Since the training was extremely quick, I decided to increase the epochs and mini-batch size both to 20, since training is not limited by processing power at this time. I also increased the eta to 1 to ensure we do not land in a local minimum, and increased the number of neurons in the hidden layer to 20, to allow for more complex computation. These hyperparameters achieved an accuracy of 72.72%, while still training quickly (Figure 11). In order to increase this accuracy even further, I increased the epochs all the way to 100, and increased the mini-batch size to 30. I then tried several different eta values to find one large enough to avoid local minimums but small enough that it will incrementally increase the accuracy. The value I landed at was 0.9, which allowed me to achieve an accuracy of 77.92% (Figure 12). This accuracy is higher than is needed for bonus, however I did not include any code adjustments that were not discussed in lecture, so I'm not sure if I am eligible for the bonus. The trained net can be found in the file 'part3.pkl'. The following is the part of the code that I edited in order to train the neural net to 77.92%:

```
net = network.Network([9,20,2])
net.SGD(trainingData, 100, 30, 0.9, test_data = testingData)
```