Theresa Inzerillo

Preston Mueller

February 1, 2016

CS4341 Project 2, Artificial Neural Networks

Experimental Write-up

Experiment 1: Holdout Percentage, Data, and Error Rate

For this experiment, since our current implementation pulls the first x% of data with which to train, we randomized the lines in the input files to get a different data set for each iteration.

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment #** | **Holdout Percentage** | **Data** | **Error Rate** |
| 1 | 5% | Data1.txt | 15.79% |
| 2 | 10% | Data1.txt | 16.061% |
| 3 | 15% | Data1.txt | 15.93% |
| 4 | 20% | Data1.txt | 15.976% |
| 5 | 25% | Data1.txt | 16.314% |
| 6 | 5% | Data2.txt | 15.97% |
| 7 | 10% | Data2.txt | 15.888% |
| 8 | 15% | Data2.txt | 15.937% |
| 9 | 20% | Data2.txt | 15.945% |
| 10 | 25% | Data2.txt | 16.012% |
| 11 | 5% | Data3.txt | 16.257% |
| 12 | 10% | Data3.txt | 16.156% |
| 13 | 15% | Data3.txt | 16.23% |
| 14 | 20% | Data3.txt | 16.162% |
| 15 | 25% | Data3.txt | 16.354% |
| 16 | 5% | Data4.txt | 16.182% |
| 17 | 10% | Data4.txt | 16.376% |
| 18 | 15% | Data4.txt | 16.467% |
| 19 | 20% | Data4.txt | 16.541% |
| 20 | 25% | Data4.txt | 16.709% |
| 21 | 5% | Data5.txt | 15.829% |
| 22 | 10% | Data5.txt | 15.873% |
| 23 | 15% | Data5.txt | 15.649% |
| 24 | 20% | Data5.txt | 15.297% |
| 25 | 25% | Data5.txt | 14.986% |

Our results are inconclusive. In general, the error does not change substantially through the tests. However, the tests using data5.txt did show a decreasing error as the test set size increased. These results convey that choosing the test set can be an important factor in training a network effectively, but are inconclusive regarding whether the size of the test data has an effect.

Experiment 2: Hidden Layer Count and Error Rate

Using a holdout percentage of 20, as specified by our command line argument of “20”.

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment #** | **Hidden Layer Count** | **Data** | **Error Rate** |
| 1 | 2 | Data1.txt | 12.41% |
| 2 | 3 | Data1.txt | 12.54% |
| 3 | 4 | Data1.txt | 12.20% |
| 4 | 5 | Data1.txt | 15.98% |
| 5 | 6 | Data1.txt | 14.98% |
| 6 | 7 | Data1.txt | 14.72% |
| 7 | 8 | Data1.txt | 12.56% |
| 8 | 9 | Data1.txt | 14.94% |
| 9 | 10 | Data1.txt | 15.14% |

Table – Changes in error rate as the hidden layer size increases.

Figure – Plot of the data displayed above in the table.

As shown by the above data, it appears our artificial neural network functions similarly despite the change in the neuron count of the hidden layer. As a result, this experiment is inconclusive. It is worth noting, though, that we have seeded the random number generator in our artificial neural network, which means that the change in neuron count is at least slightly changing the success of the network. A future experiment might be better served by a far larger test set.

Program Description and Simplifying Assumptions (Question 3)

While we implemented our neural network in Python, we opted to build a modular program structure that would encourage customizability going forward. We built a NeuralNetwork class that manages the whole network, as well as a Node class and a Synapse class that connect layers together. The size of a network constructed using our classes can be controlled directly via parameters input on the command-line in the case of hidden layer or via variables in the NeuralNetwork in the case of the other layers. The network topology is mostly set in stone—that is, the network always accepts two-dimensional input and executes an exclusively binary classification. We believe that configuration was most appropriate for this assignment, however we recognize there are classification problems in Computer Science that require multi-attribute output.