Assignment 5 - Neural Network Ranking

Aleksander Skraastad

May 2, 2016

1 Implementation

First of all, I had to clean up the code, as it was very badly formatted. It did not even come close to following the Python code style guide. This is of course not something needed to make the algorithm work, but it makes it easier for both the student and TA to work with. Additionally, the following alterations were made to the files:

1.1 nnet.py (Backprop)

- Compute output delta (Just straight forward implementing formulas from the assignment text.)
- Compute hidden delta (Just straight forward implementing formulas from the assignment text.)
- Update weights (Just straight forward implementing formulas from the assignment text.)
- Backpropagation (Compute output, compute hidden, update weights)
- Train (Just propagate A and B, then backpropagate)
- Count miss (Simply count errors and divide by total)

1.2 dataset.py (DataLoader)

- Creating the training and testing pairs (NxN matrix, iterate over half (triangle) and keep those x-y pairs that have differing rating.
- Store and calculate mean training and testing error rates (Yield result tuples from each epoch in the training loop)
- Add matplotlib code for plotting the graph

2 Graph analysis

Looking at the mean training and testing error rates in figure 1, we can see that the network performs a bit better on the training data, compared to the test data. This is what we rationally would assume, as the network weights are tuned to that exact data after all.

We also see that they both converge at around 75% at epoch 10. This tells us that given the current training data and network topology, further increase in accuracy is not possible. Increasing the number of epochs will have no positive effect. It may lead to some slight overfitting on the training data, which might trigger a decrease in testing correctness. Adjusting learning rate will have no positive effect, and will only cause the graph to oscillate, as it is the data and network topology that prevents any further correctness. Increasing the node count in the hidden layer only encourages over-fitting on this data set, something we can determine by witnessing the training error go down while testing error go up.

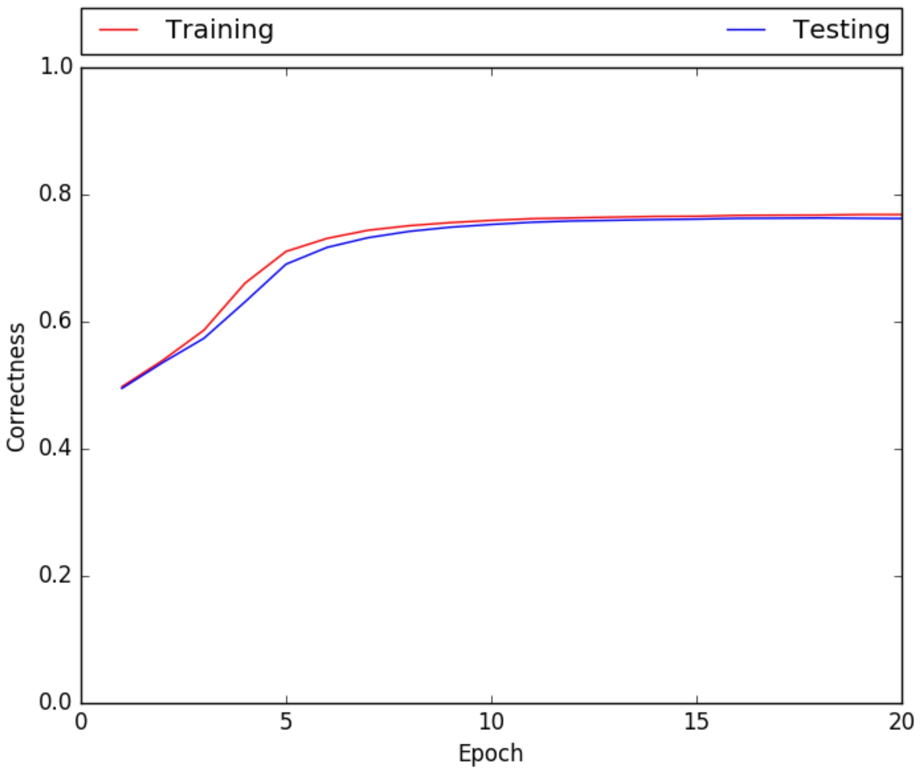

Figure 1: Training and testing correctness averaged over 5 runs

3 Problems

I have worked with neural networks in both AI programming and sub-symbolic AI methods, so this was very straight forward, and I faced no problems.