

High frequency trading with an artificial neural network

Kevin Bretney and Zack Coburn

November 29, 2008

I Introduction

In the 1940s, Warren McCulloch and Walter Pitts, a neurophysiologist and mathematician, began to speculate about how the human brain might work. They modeled the brain with a neural network, or a collection of interconnected neurons that can, as a whole, learn and make decisions. Today, artificial neural networks are commonly used for their incredible ability to learn the intricacies of nonlinear systems. ¹

In the widely-used feedforward neural network (see **Figure 1**), neurons are organized in layers, where the first layer is the input layer, the next layers are hidden layers, and the last layer is the output layer. Each node in a layer is connected by a weighted edge to each node in the next layer. Each hidden node and output node has a value, which is typically calculated as the dot product of the weights and the values of the nodes in the previous layer, plus an additional weight ($\sum_i w_i g_i + b$). Usually, an activation function, such as the hyperbolic tangent function (\tanh), is applied to the value. ^{2 3}

Various algorithms, such as the backpropagation algorithm, can be used to efficiently train a neural network so that it can learn the weights needed such that the outputs as a function of the inputs behave as desired. The backpropagation algorithm operates by giving the neural network input values, calculating the error in each of the output nodes, and adjusting the weights slightly in order to lower the error. Next, the algorithm assigns a blame value to each of the nodes in the previous hidden layer, where the blame values are proportional to each hidden node's weight. The algorithm then continues through the hidden layers until it has adjusted all of the weights as necessary. The algorithm can then be repeated for more data points, usually until the error is reduced to a certain level. ⁴

The newest trend in the stock market is high frequency trading. This has only come about in the past few years due to the availability of real time market data. High frequency trading gets its name from the trading of stocks many times per day depending on market information obtained as soon as it is available. Most trading strategies cannot easily be applied to high frequency trading since they are mostly based on a lot of past data and do not react as quickly as is needed for trading at a high frequency. Neural networks, on the other hand, can be set up to take in a number of data points and can be continuously trained on new data to be ideally suited for the current state of the market at all times. They can also learn trading rules with which humans would struggle by automatically learning nonlinear functions. In this paper, we shall develop a high frequency trading system powered by a neural network.

II Our set up

Data

Through Babson College's Cutler Center, we accessed Wharton's WRDS service and downloaded minute-by-minute data from the NYSE Trade and Quote Database consisting of bid price, ask

¹<http://cse.stanford.edu/class/sophomore-college/projects-00/neural-networks/History/history1.html>

²Fine. "Feedforward Neural Network Methodology."

³http://en.wikipedia.org/wiki/Feedforward_neural_network

⁴<http://en.wikipedia.org/wiki/Backpropagation>

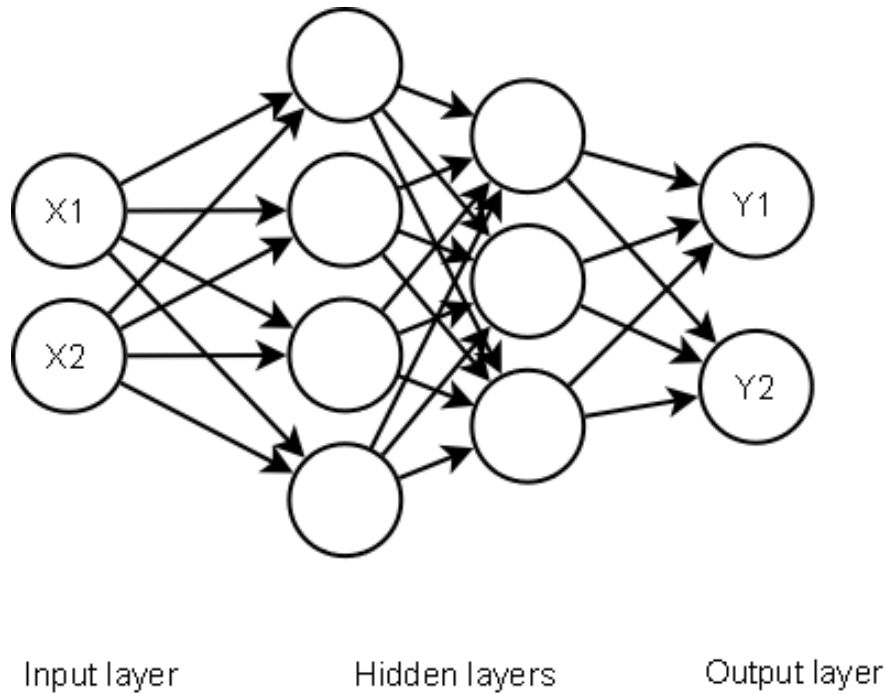


Figure 1: This is an example of a feedforward neural network with two outputs, two inputs, and two hidden layers.

price, bid size, and ask size for a variety of technology stocks. We used 30-minute intervals⁵ from this data and constructed a data table consisting of the following columns:

- percent change in price
- average price
- bid price
- ask price
- bid size
- ask size
- bid-ask spread
- standard deviation of previous hundred prices
- five previous percent changes
- five previous average prices
- five previous standard deviations
- five previous bid-ask spreads

⁵Originally, we considered using 5-minute intervals, but discovered that prices do not move enough in 5-minute intervals for the neural network strategy to overcome the bid-ask spread in each trade.

Neural network

We constructed a feedforward neural network using the percent change in price as the output, the rest of the columns as the inputs, and a single hidden layer with 100 hidden nodes. We trained the network using backpropagation and used the difference in its predicted output and the actual output to determine how well the model worked. We used the Netlab neural networks library for MATLAB to do this. ⁶

Trading rules

Rather than simply test the accuracy of the neural network, we decided to construct a strategy using the neural network and then test its profitability. We used the following simple trading rules:

- If the neural network predicts the price will increase and you are
 - Neutral: go long
 - Long: stay long
 - Short: close the position and go long
- If the neural network predicts the price will decrease and you are
 - Neutral: go short
 - Short: stay short
 - Long: close the position and go short

Training and testing

We used a rolling training and testing strategy, so that the neural network would be trained using a number of periods from the immediate past (represented by the *lookback* parameter), used for a number of periods, and then retrained every so many periods (represented by the *retrain every* parameter). Before generating results, we optimized the *lookback* and *retrain every* parameters using AAPL in January 2000 as a trading subject. **Figure 2** shows the results of this optimization, which led us to choose 25 for the *lookback* parameter and 20 for *retrain every* parameter.

III Results

Figures and **Tables** 3 through 8 show the results of trading various stocks in January 2000 using the neural network strategy. Out of seven cases, in four cases the neural network strategy outperforms the buy and hold strategy, but in only two cases does the neural network strategy actually profit. Although the neural network's strategy on AAPL seems outstanding, we should discount this success considering we used AAPL to optimize the strategy parameters on the same data set, which may have caused overfitting.

⁶<http://www.ncrg.aston.ac.uk/netlab/index.php>

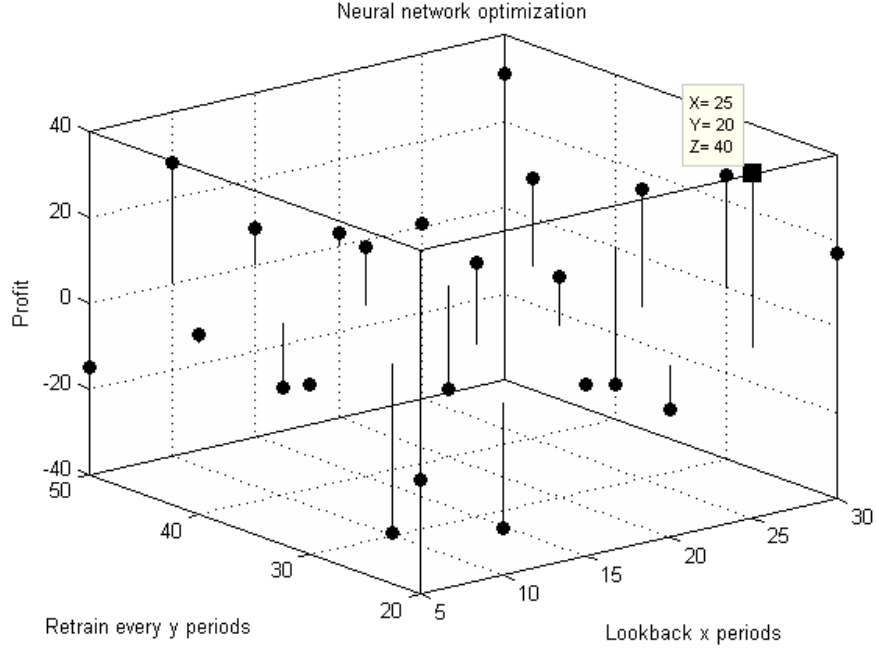


Figure 2: This figure shows the optimization for the *lookback* and *retrain every* parameters. The X and Y axes represent the two parameters, and the Z axis represents the profit after one month of trading.

Table 3: AAPL, January 2000

Category	Count	Mean profit
<i>Profitable trades</i>	32	2.3887
<i>Unprofitable trades</i>	28	-1.433
<i>Long trades</i>	30	0.60208
<i>Short trades</i>	30	0.60833
<i>Profitable long trades</i>	13	3.1442
<i>Profitable short trades</i>	19	1.8717
<i>Unprofitable long trades</i>	17	-1.3419
<i>Unprofitable short trades</i>	11	-1.5739

Table 4: CSCO, January 2000

Category	Count	Mean profit
<i>Profitable trades</i>	32	1.0625
<i>Unprofitable trades</i>	35	-1.2554
<i>Long trades</i>	33	-0.20455
<i>Short trades</i>	34	-0.09375
<i>Profitable long trades</i>	16	1.0039
<i>Profitable short trades</i>	16	1.1211
<i>Unprofitable long trades</i>	17	-1.3419
<i>Unprofitable short trades</i>	18	-1.1736

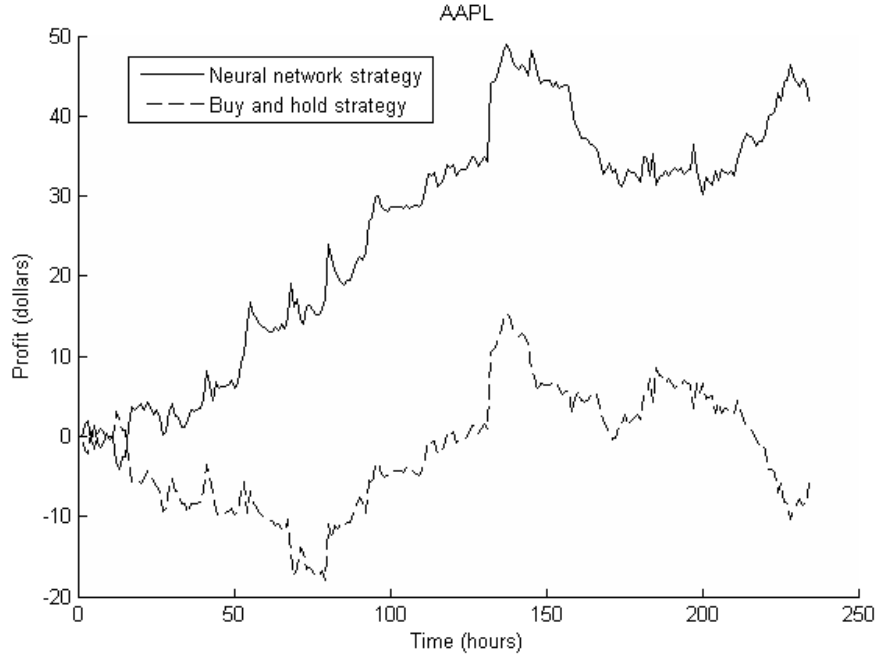


Figure 3: This figure shows the result of trading AAPL in January 2000 using the neural network strategy.

Table 5: INTC, January 2000

Category	Count	Mean profit
<i>Profitable trades</i>	46	1.1563
<i>Unprofitable trades</i>	43	-0.97965
<i>Long trades</i>	44	0.25284
<i>Short trades</i>	45	-0.0013889
<i>Profitable long trades</i>	20	1.7156
<i>Profitable short trades</i>	26	0.72596
<i>Unprofitable long trades</i>	24	-0.96615
<i>Unprofitable short trades</i>	19	-0.99671

Table 6: MSFT, January 2000

Category	Count	Mean profit
<i>Profitable trades</i>	31	1.1855
<i>Unprofitable trades</i>	25	-1.8725
<i>Long trades</i>	28	-0.49107
<i>Short trades</i>	28	0.1317
<i>Profitable long trades</i>	13	0.70192
<i>Profitable short trades</i>	18	1.5347
<i>Unprofitable long trades</i>	15	-1.525
<i>Unprofitable short trades</i>	10	-2.3937

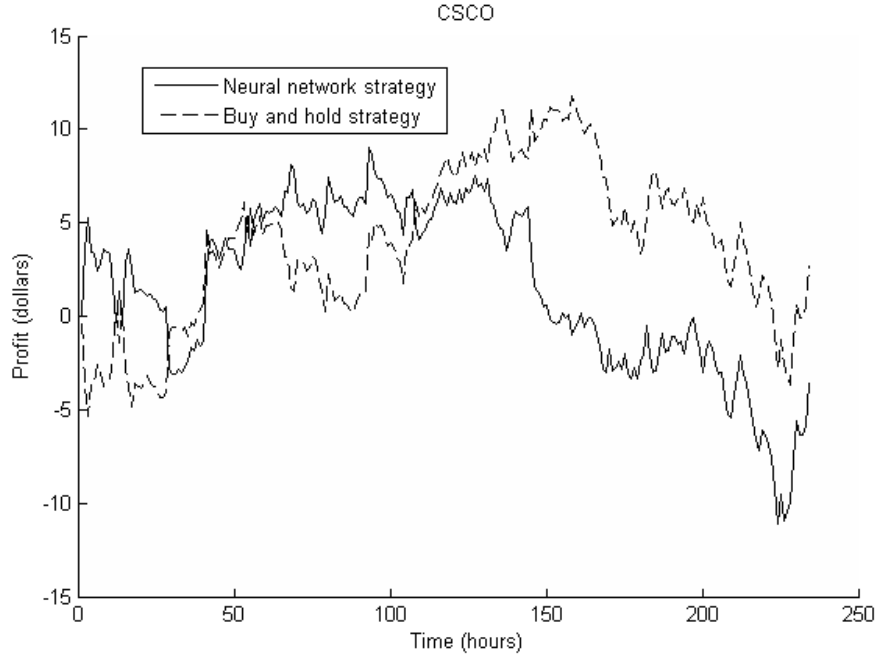


Figure 4: This figure shows the result of trading CSCO in January 2000 using the neural network strategy.

Table 7: ORCL, January 2000

Category	Count	Mean profit
<i>Profitable trades</i>	36	0.82986
<i>Unprofitable trades</i>	42	-2.6503
<i>Long trades</i>	39	-1.8221
<i>Short trades</i>	39	-0.26603
<i>Profitable long trades</i>	16	0.66406
<i>Profitable short trades</i>	20	0.9625
<i>Unprofitable long trades</i>	23	-3.5516
<i>Unprofitable short trades</i>	19	-1.5592

Table 8: QCOM, January 2000

Category	Count	Mean profit
<i>Profitable trades</i>	23	3.6821
<i>Unprofitable trades</i>	21	-4.8601
<i>Long trades</i>	22	-1.5227
<i>Short trades</i>	22	0.73295
<i>Profitable long trades</i>	12	3.7292
<i>Profitable short trades</i>	11	3.6307
<i>Unprofitable long trades</i>	10	-7.825
<i>Unprofitable short trades</i>	11	-2.1648

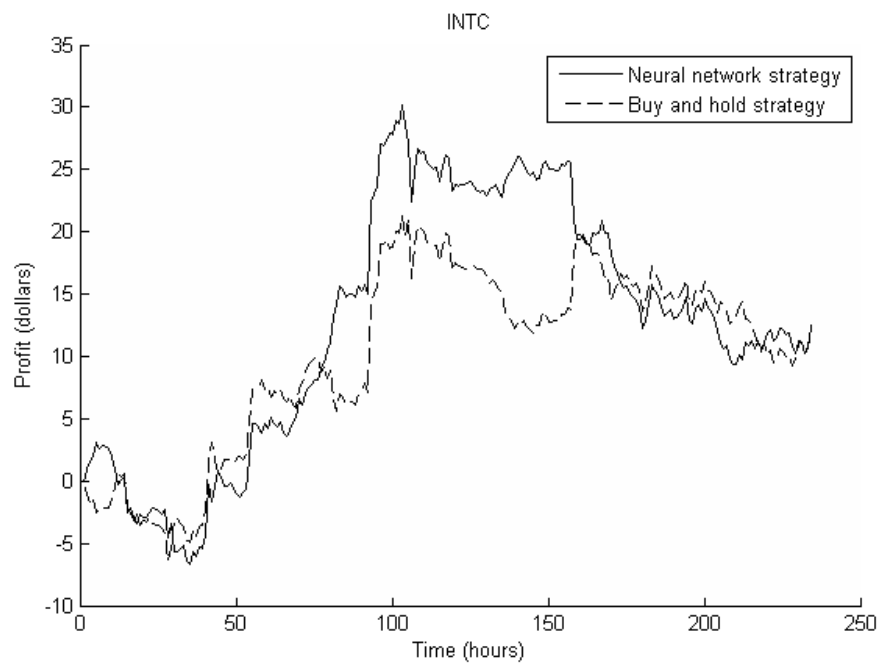


Figure 5: This figure shows the result of trading INTC in January 2000 using the neural network strategy.

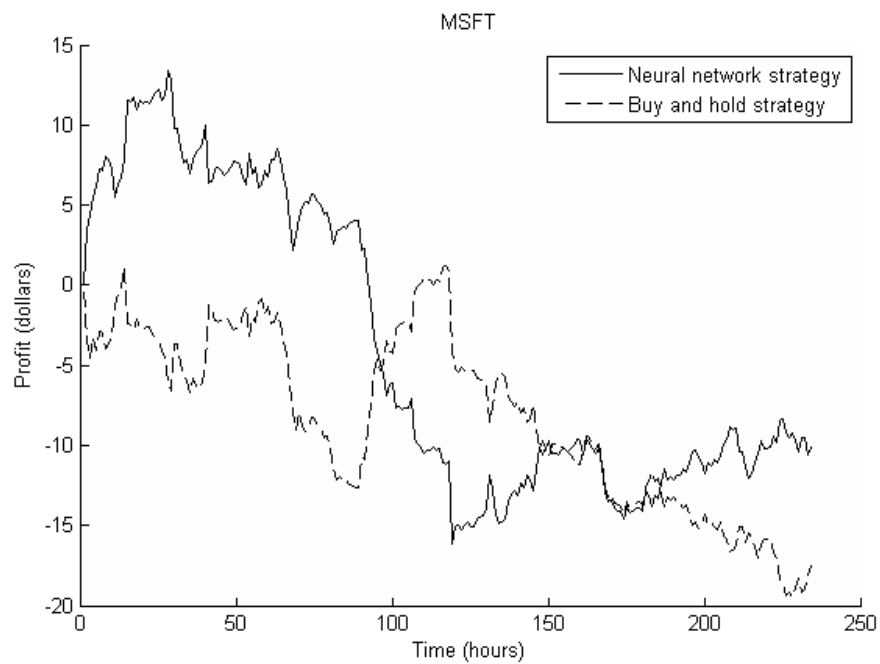


Figure 6: This figure shows the result of trading MSFT in January 2000 using the neural network strategy.

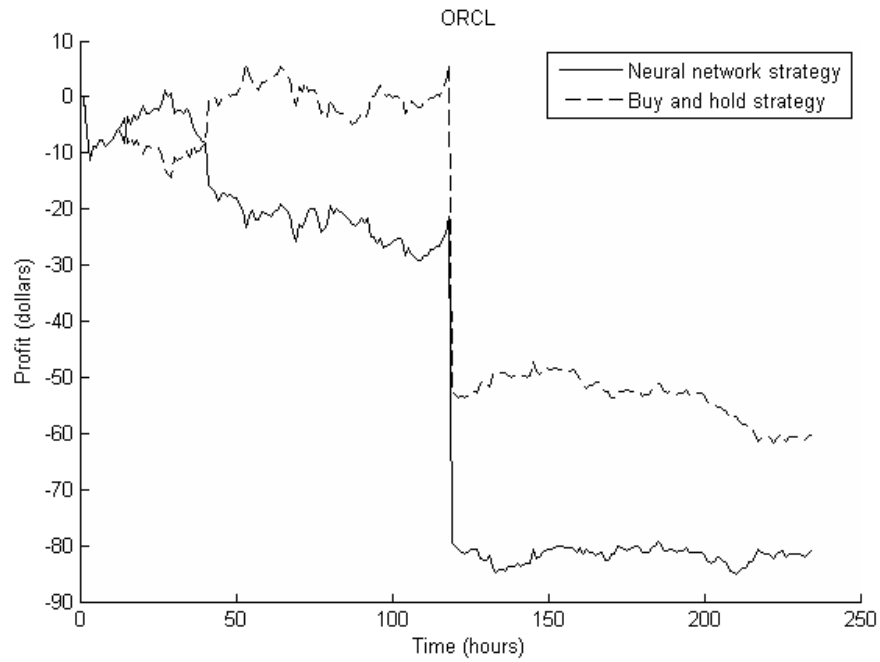


Figure 7: This figure shows the result of trading ORCL in January 2000 using the neural network strategy.

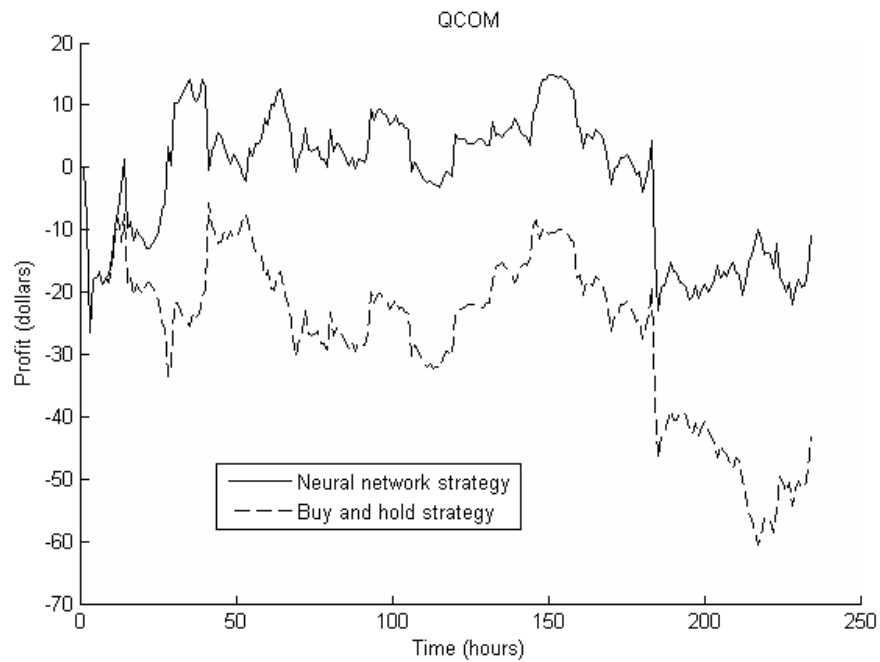


Figure 8: This figure shows the result of trading QCOM in January 2000 using the neural network strategy.