The Use of Parsimonious Neural Networks for Forecasting Financial Time Series

Robert Dorsey[i] and Randall Sexton[ii]

Abstract

When attempting to determine the optimal complexity of a neural network the correct decision is dependent upon obtaining the global solution at each stage of the decision process. Failure to ensure that each optimization being considered is near a global solution may lead to misleading and often conflicting results jeopardizing any decision rule for choosing an optimal complexity. Here, a genetic algorithm is used for global search and, by modifying the objective function, is used to simultaneously select a parsimonious structure. The chosen structure often eliminates all connections to unnecessary variables and thus identifies irrelevant variables. Several models are examined to forecast the five day relative difference of the S&P 500 index. The first series of models uses only past S&P 500 values while the second series of models uses other explanatory variables. Models with the complete architecture are compared to those with the reduced structure. Based on the preliminary model analysis a composite model is constructed.

[i]Department of Economics, Conner Hall, University of Mississippi, 38677, (601)232-7575

[ii]Department of Management, Ball State University, Muncie, IN, 47306,(317) 285-5320

The Use of Parsimonious Neural Networks for Forecasting Financial Time Series

Extensive use has been made of artificial neural networks for modeling financial time series both in the academic literature and in professional publications. The results of these studies have been mixed. While it is often the case that the neural network is shown to dominate other estimation techniques, the reliability of these results is frequently found to be inconsistent. While there are likely many reasons for these mixed results, two reasons that are frequently discussed are the inability of most optimization algorithms to obtain an optimal set of weights for the neural network and the fact that it is common for neural networks to be over parameterized.

The neural network offers a great deal of promise as a highly flexible approximation tool which may be able to capture and approximate the underlying patterns in data that are difficult to identify through other techniques.[1] Increasing the parameters of the model by adding additional hidden nodes generally allows increasingly accurate fits of the training data. However, the practical use of the neural network for noisy data depends on the neural network being able to generalize. The flexibility of the neural network enables it, given sufficient parameters, to achieve a high degree of fit to the training data, which of course, includes the noise. The obvious goal is to obtain a neural network that captures the underlying pattern of the data without fitting the noise. Researchers have explored a variety of procedures for enhancing the ability of the neural network to generalize and these techniques commonly involve some means for reducing the number of

[1]See Hornik, Stinchcombe, & White, [1989] for a discussion of the potential capabilities of the neural network as a universal approximator.

parameters (weights) in the model. A primary concern in all these techniques is that the decision rule for inclusion or removal of the candidate weight depends crucially on being at an optimal solution. If, the neural network under consideration is not at a global optimum, the decision rule in general will not be valid.

This presents a serious problem since the hill-climbing techniques commonly used for optimizing the neural network are prone to becoming trapped at local optima. Such solutions can lead to erroneous decisions for including or excluding weights and thus to poor generalization performance for the model. Recent work on the use of global search algorithms for optimizing neural networks [Sexton, Dorsey and Johnson 1997, Alidae, Dorsey, Sexton and Johnson 1997, and Dorsey, Johnson and Mayer 1994] may enable researchers to more consistently achieve global solutions and thus make the correct choices for the neural network architecture to achieve a generalized model. However, the genetic algorithm is a time consuming search technique and repetitive search and test procedures may result in far too lengthy a process for real world applications.

In this paper we use the genetic algorithm for optimizing a neural network in three ways. We first use the conventional sum of squared errors as the optimization criteria. Next, after the neural network is trained using sum of squared errors we apply a penalty for unnecessary weights and allow the genetic algorithm to optimize further. The size of the penalty is equal to the current level of the RMSE or the typical error of one observation. Finally, we use only the objective function with the penalty for optimization and then compare the results from all three techniques. By setting the penalty equal to the RMSE we prevent the neural network from eliminating too many weights. As a result, a connection or weight is only eliminated if the sum of squared errors is increased by less than the RMSE. This procedure allows the genetic algorithm to search simultaneously over both the parameter space and potential architectures. A potential disadvantage of this approach is that in the early stages of the optimization substantial gains could be achieved by eliminating extensive portions of the neural network structure and thus removing too many parameters. To evaluate that possibility we compare optimization without the elimination of any structure to optimization where elimination only occurs at the end of the process to optimization including elimination of unnecessary parameters from the beginning

All optimization runs were conducted on a Pentium 133 Mhz machine. For all but one of the reported comparisons we limit the search by the genetic algorithm to approximately 30 minutes, a time we felt could be accommodated in most applications. In the next section we describe the procedures used for this study and the results. In the final section we summarize our findings.

Procedures and Results

The data was provided by the *Journal of Computational Intelligence in Finance* and consisted of daily closing prices of the S&P 500 for the ten year period 1985 to 1994. The target variable was the relative difference in percent between the current period closing price and the closing price five days ahead. Two years of daily data from 1995 to 1996 for the same variables was also supplied for out of sample analysis. To identify a set of explanatory variables to use with the neural network the following combinations of the time series were examined. Ordinary least squares was used on the sets of explanatory variables to determine which set provided the best linear explanatory model for the target variable. Set 12, which consisted of the current value and nine lagged values of the difference in the S&P index performed best and was used as the starting point for the development of the model.

Data Series Evaluated With Ordinary Least Squares

1. $S\&P_t$
2. $S\&P_{t-i} \quad i=0,1$
3. $S\&P_{t-i} \quad i=0...4$
4. $S\&P_{t-i} \quad i=0...9$
5. $\log S\&P_t$
6. $\log S\&P_{t-i} \quad i=0,1$
7. $\log S\&P_{t-i} \quad i=0...4$
8. $\log S\&P_{t-i} \quad i=0...9$
9. $S\&P_t - S\&P_{t-1}$
10. $S\&P_{t-i} - S\&P_{t-(i+1)} \quad i=0,1$
11. $S\&P_{t-i} - S\&P_{t-(i+1)} \quad i=0...4$
12. $S\&P_{t-i} - S\&P_{t-(i+1)} \quad i=0...9$
13. $\dfrac{S\&P_t - S\&P_{t-1}}{S\&P_{t-1}}$
14. $\dfrac{S\&P_{t-i} - S\&P_{t-(i+1)}}{S\&P_{t-(i+1)}} \quad i=0,1$
15. $\dfrac{S\&P_{t-i} - S\&P_{t-(i+1)}}{S\&P_{t-(i+1)}} \quad i=0...4$
16. $\dfrac{S\&P_{t-i} - S\&P_{t-(i+1)}}{S\&P_{t-(i+1)}} \quad i=0...9$

The architecture for the feed forward neural network used for this analysis consisted of an input layer, a hidden layer and an output layer. The hidden layer contained 5 hidden nodes. Each node using the standard sigmoid nonlinear function:

$$y_t = \frac{1}{1+e^{-\sum_{i-1}^{n} b_i x_{it}}}$$

The hidden layer also contained a bias node set at -1. The input layer for contained 10 input nodes corresponding to the current and nine lagged values of the difference and one bias node. The output layer contained only one node and there was no nonlinear function on the output node. The neural network therefore has 55 weights on the input layer and 6 weights on the output layer for a total of 61 weights. The configuration can be seen in Figure 1.

Figure 1 about here.

The neural network was trained using a genetic algorithm. Dorsey and Mayer (1995) or Sexton, Dorsey and Johnson (1997) provide a detailed description of the training algorithm. There were 2,518 observations in the training set. The initial objective function used was minimization of the sum of squared errors,

$$Min\sum_{i=1}^{2518}\left(y_t - \hat{y}_t\right)^2$$

where y_t is the t-th observation of the target variable and $\hat{y}_t$ is the corresponding forecast of the neural network model.

The neural network was trained for 5000 generations ten times. The only difference between the runs was that the training was initiated with a different seed for the random number generator. Results were compared on the basis of directional symmetry and normalized root mean squared error. Directional symmetry is the percent of forecasts where the movement of the forecast was the same as the movement of the target variable. The normalized root mean squared error is the root mean squared error of the forecasts divided by the standard deviation of the data for the same period.

Table 1 shows the fit of the model on the training data and Table 2 shows how the models performed on the 506 out of sample observations. The average of the ten runs is shown along with the best performance from the ten replications both in terms of directional symmetry and in terms of the normalized root mean square error. Two additional columns show how the directional symmetry was distributed between up days and down days.

Table 1 - In Sample Results of SSE Objective Function on S&P 500 Data

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	4.2973	0.9640	59.85%	75.51%	38.16%
Best nRMSE	4.2973	0.9640	59.85%	75.51%	38.16%
Average	4.3389	0.9686	58.68%	73.46%	38.21%

Table 2 - Out of Sample Results of SSE Objective Function on S&P 500 Data

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	2.5860	1.0923	57.71%	66.86%	38.79%
Best norms	2.5860	1.0837	54.74%	66.86%	38.79%
Average	2.6638	1.1086	53.97%	59.77%	42.00%

The same data was then used for ten more trials. However, in this case, after the ten neural networks had each been trained for 5000 generations, the objective function was changed in an attempt to eliminate unnecessary parameters (weights) of the neural network. This alternative objective function used for the genetic algorithm is:

$$Min\sum_{i=1}^{2523}\left(y_i - \hat{y}_i\right)^2 + \sum_{j=1}^{61} kIND\{w_j\}$$

where IND{} takes on the value 1 if the weight is non zero and zero otherwise. The constant k is assigned the value of the current value of the RMSE. This objective function thus imposes a penalty for neural network connections. Connections will be removed when their elimination does not increase the SSE by more than the RMSE.

The use of this alternative objective function requires a slight modification to the genetic algorithm as described in Dorsey and Mayer (1995). The mutation operation randomly draws parameter values over the real line and as a result would never draw a hard zero. The mutation operator is therefore modified to choose a zero approximately 20% of the time. This 20% is randomly selected and the remaining mutations are drawn, as usual, from the real line.

Each neural network was trained for an additional 1000 generations and then out of sample forecasts were made. The average and best of those forecasts are shown in Table 3 for the training data and Table 4 for the out of sample data. As can be seen, there was not a significant increase in the MSE of the forecasts and, the forecasts of the direction of movement for the out of sample data actually improved. On average, the 1000 generations of training with the alternative objective function resulted in approximately 10% of the weights being eliminated from the neural network. The reason for improvement is likely due to improved optimization once unnecessary parameters are removed. The performance of the genetic algorithm for any fixed number of generations will obviously improve as the sample space is reduced.

Table 3 - In Sample Results with Combined Function on S&P 500 Data

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	4.2977	.09641	59.49%	75.03%	37.97%
Best RMSE	4.2977	0.9641	59.49%	75.03%	37.97%
Average	4.3388	0.9687	58.66%	73.56%	38.03%

Table 4 - Out of Sample Results with Combined Function on S&P 500 Data

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	2.5780	1.0906	58.30%	67.16%	40.00%
Best RMSE	2.5542	1.0856	54.15%	67.16%	40.00%
Average	2.6657	1.1090	54.05%	59.88%	42.00%

In a third set of ten trials on the same data, only the alternative objective function was used. Each neural network was trained for 5000 generations. The best and average values are shown in Table 5 for the training data and Table 6 for the out of sample data. Here, there was a slight reduction in the out of sample MSE and the forecasts of the direction of movement of the target variable increased to an average of over 59% for the out of sample data. This alternative objective function eliminated an average of 64% of the weights of the ten neural networks so that, on average, only 22 of the original 61 weights remained. Despite the substantial reduction in the number of weights there was no substantive loss of predictive capability.

Table 5 - In Sample Results with Alternative Objective Function on S&P 500 Data

	MSE	NRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	4.4348	0.9794	59.13%	80.85%	29.07%
Best RMSE	4.3815	0.9734	58.74%	82.69%	25.57%
Average	4.4000	0.9755	58.43%	83.46%	23.11%

Table 6 - Out of Sample Results with Alternative Objective Function on S&P 500 Data

	MSE	NRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	2.7370	1.1237	60.47%	68.62%	39.39%
Best RMSE	2.6556	1.1069	52.96%	68.62%	39.39%
Average	2.7370	1.1237	58.06%	68.62%	39.39%

To demonstrate the effects of this objective function, Figure 2 shows the neural network after the connections have been removed for the model that performed best at forecasting Directional Symmetry out of sample. This neural network initially had the architecture shown in Figure 1. Clearly, inputs 1, 2, 3 and 4 are not necessary since the neural network can perform equally well without them. These inputs correspond to the current and the first three lagged differences. None of the remaining inputs required more than two connections to the hidden nodes. Those which did require two connections were the differences lagged 4, 5 and 9 days or approximately one and two week lags.

Figure 2 about here

Additional data was also provided by the *Journal of Computational Intelligence in Finance* to enable models to be developed which contain other potentially relevant explanatory variables. These seven additional variables are shown below.

1. Dow Jones Industrial Average
2. Dow Jones Transportation Average
3. NYSE Volume
4. Dow Jones 20 Bond Average
5. Dow Jones Utility Average
6. S&P High
7. S&P Low

For each of these data series plus the S&P 500 series, the following summary statistics were calculated.

$$\text{Difference}: DIF_t = V_t - V_{t-1}$$

$$\text{Moving Average}: MA_t = \frac{\sum_{i=0}^{4} V_{t-i}}{5}$$

$$\text{Standard Deviation}: S_t = \sqrt{\frac{\sum_{i=0}^{4} \left(V_{t-i} - MA_t\right)^2}{5}}$$

This generated 24 variables which were used as the inputs for a neural network. The neural network again had 5 hidden nodes plus a bias term and one output node. There were 24 inputs plus a bias term in the input layer. Thus, the 125 weights in the input layer and the six weights in the output layer generated a total of 131 weights in the neural network. The neural network was trained on 2523 observations and 506 observations were used as the hold out sample. The models were trained as before and the results are shown in Tables 7 - 12.

Using the sum of squared errors as the objective function, the MSE is similar to the corresponding MSE for the models based on the S&P 500 data alone. The forecast of direction, however, is significantly better, rising to nearly 63% on average for the out of sample forecasts.

Table 7 - In Sample Results with SSE Objective Function on Eight Data Series

	MSE	NRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	4.4348	0.9794	59.13%	80.85%	29.07%
Best RMSE	3.6837	0.8927	58.46%	77.51%	32.01%
Average	4.0192	0.9309	59.02%	87.89%	18.91%

Table 8 - Out of Sample Results with SSE Objective Function on Eight Data Series

	MSE	NRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	3.6096	1.9930	66.80%	84.46%	22.42%
Best RMSE	2.3046	1.0312	58.70%	84.46%	22.42%
Average	3.4790	1.2319	62.67%	84.81%	16.91%

When the alternative objective function was combined with the SSE both the MSE and the direction rate stayed approximately the same. On average 47.4% of the weights were eliminated or approximately 62 of the 131 weights.

Table 9 - In Sample Results with Combined Objective Function on Eight Data Series

	MSE	NRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	3.9101	0.9185	60.44%	83.16%	28.88%
Best RMSE	3.7129	0.8950	58.07%	78.73%	29.36%
Average	4.0002	0.9290	59.03%	84.87%	23.14%

Table 10 - Out of Sample Results with Combined Objective Function on Eight Data Series

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	4.4796	1.4376	66.80%	82.99%	23.64%
Best RMSE	2.3153	1.0335	59.49%	82.99%	23.64%
Average	3.4637	1.2356	62.21%	81.14%	23.09%

When only the alternative objective function was used there was an improvement in the out of sample MSE but there was also a decrease in the average out of sample Directional Symmetry performance. There was, however, a substantial reduction in total number of weights. Across the ten neural networks there was an average reduction in weights of 70.23% or approximately 92 of the 131 weights were eliminated.

Table 11 - In Sample Results with Alternative Objective Function on Eight Data Series

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	3.8455	0.9109	61.04%	81.25%	32.95%
Best RMSE	3.8455	0.9109	61.04%	81.25%	32.95%
Average	3.8923	0.9163	59.71%	79.02%	32.89%

Table 12 - Out of Sample Results with Alternative Objective Function on Eight Data Series

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
Best DS	2.2091	1.0096	65.02%	86.80%	20.00%
Best RMSE	2.2091	1.0096	65.02%	86.80%	20.00%
Average	2.3366	1.0380	58.97%	68.62%	39.03%

For the particular neural network that estimated the DS best out of sample, of the original 131 weights, 104 were eliminated. All connections to 10 of the 24 variables were eliminated indicating that those ten variables provided no additional explanatory power to the model. Of the remaining 14 explanatory variables, six and the bias term had two connections to the hidden layer and all other variables only had one connection. The explanatory variables that remained are shown in Table 13 along with the number of connections to the hidden layer.

Table 13 - Remaining Connections for Model 6 Example

	Net Difference	5 Day Moving Average	Standard Deviation
DJIA	-	2	-
DJTA	-	-	2
NYSE Volume	-	1	1
DJ 20 Bond Avg	-	2	1
DJ Utility Avg	-	2	1
S&P High	-	1	1
S&P Low	2	1	-
S&P 500	1	2	-

The final model that was examined combined the six variables which were not eliminated in the neural network shown in Figure 2 with the remaining 14 variables from the neural network shown in Table 13. This created a neural network with a total of 19 explanatory variables since one was common to both. Using five hidden nodes, this generates a total of 106 connection weights. For this model we trained the neural network for 45000 generations or approximately 10 hours on the Pentium PC. The in sample and out of sample fit is shown in Table 14.

Table 14 - Combined Data Results

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
In Sample	3.6766	0.8917	58.82%	98.50%	3.88%
Out of Sample	2.3512	1.0415	67.59%	68.62%	39.39%

When the above model was trained for an additional 1000 generations with the alternative objective function, 32 of the connections were eliminated. The effect on the neural networks forecasting ability is shown in Table 15. Despite the fact that nearly one third of the weights were eliminated the forecast accuracy appears to be unchanged.

Table 15 - Combined Data Reduced NN Model Results

	MSE	nRMSE	Directional Symmetry	% of Up Days Correct	% of Down Days Correct
In Sample	3.6863	0.8929	58.72%	98.63%	3.50%
Out of Sample	2.3855	1.0491	67.39%	68.62%	39.39%

Summary and Conclusions

This paper explores an alternative protocol for optimizing neural networks for financial time series data. Two of the most common problems with neural networks are the difficulty in obtaining a global solution in the training phase and the uncertainty as to whether or not the model is over paramiterized. In this study, a global search technique, the genetic algorithm, is used not only to better obtain the global solution but also to eliminate unnecessary weights. The parsimonious neural networks that result are equally accurate in sample and perform as well or better that the corresponding non reduced neural networks when forecasting out of sample.

By allowing the penalty weight to vary during the training process there does not appear to be any tendency for the genetic algorithm to eliminate structure too quickly. In each of the examples the results obtained with the three training methods were quite similar. If the penalty weight is set too high then the genetic algorithm will tend to eliminate structure at a cost of explanatory power. On the other hand, if the weight is set too low unnecessary connections will be maintained and thus there will be a loss of generality. Although this study used the current value of the RMSE as the penalty weight, this is arbitrary and may not be optimal. Further research to identify the range for optimal values of the penalty is clearly necessary.

In total there were three sets of explanatory variables used in the neural networks and the neural networks were optimized three different ways. The best in sample normalized root mean squared error was 0.8917 using the combined data and the complete neural network architecture. The best out of sample normalized root mean squared error was 1.0096 which was achieved by the neural network using the eight data series after the neural network structure had been reduced by using the alternative objective function for the full training run. The best Directional Symmetry result for the in sample data was also the neural network using the eight data series and which had been reduced by using the alternative objective function for the full training run. It was able to correctly forecast the direction of movement 61.04% of the time. The best out of sample results were achieved by the combined data with the full architecture. It was able to correctly forecast the direction of the movement 67.59% of the time. This was reduced only slightly to 67.39% after 32 of the connections had been removed by the alternative objective function.

References

Dorsey, R.E. and J. D. Johnson, [1997], AEvolution of Dynamic Reconfigurable Neural Networks: Energy Surface Optimality Using Genetic Algorithms,A *Optimality in Biological and Artificial Networks?*, Daniel S. Levine and W. R. Elsberry Editors, Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 185-202.

Dorsey, R. E., & Mayer W. J., [1995], AGenetic Algorithms for Estimation Problems with Multiple Optima, Non-Differentiability, and other Irregular Features,@ *Journal of Business and Economic Statistics*, 13(1), pp. 53-66.

Dorsey, R. E., Johnson, J. D., & Mayer, W. J., [1994], AA Genetic Algorithm for the Training of Feedforward Neural Networks,@ *Advances in Artificial Intelligence in Economics, Finance and Management* (J. D. Johnson and A. B. Whinston, eds.,). Vol. 1. Greenwich, CT: JAI Press Inc., pp.93-111.

Hornik, K., Stinchcombe, M., & White, H., [1989], AMultilayer Feed-Forward Networks are Universal Approximators,@ *Neural Networks*, 2(5), pp. 359-66.

Sexton, R. , B. Alidaee, R. Dorsey, and J. Johnson, [1997] AGlobal Optimization for Artificial Neural Networks: A Tabu Search Application,@ *European Journal of Operational Research,* Forthcoming

Sexton, R., Dorsey, R., & Johnson, J, [1997] AToward a Global Optimization of Neural Networks: A Comparison of the Genetic Algorithm and Neural Networks,@ *Decision Support Systems,* Forthcoming