Res. Lett. Inf. Math. Sci., 2004, Vol. **6**, pp 15-29

Available online at http://iims.massey.ac.nz/research/letters/

Equity trend prediction with neural networks

R.HALLIDAY

Institute of Information & Mathematical Sciences
Massey University at Albany, Auckland, New Zealand

Russell.Halliday.1@uni.massey.ac.

This paper presents results of neural network based trend prediction for equity markets. Raw equity exchange data is pre-processed before being fed into a series of neural networks. The use of Self Organising Maps (SOM) is investigated as a data classification method to limit neural network inputs and training data requirements. The resulting primary simulation is a neural network that can prediction whether the next trading period will be, on average, higher or lower than the current. Combinations of pre-processing and feature extracting SOM's are investigated to determine the more optimal system configuration.

---

1 Introduction

Prediction of financial markets has long been a holy grail in the minds of equity investors. With the advent of powerful computers much attention has been focused on this field. The ability of neural networks to learn from training data has not been overlooked and as such neural networks have been applied to a range trading market applications from equity markets to currency markets. Contrasting this is a level of skepticism surrounding the ability of a system to predict future prices of trading markets[2].

Equity market prices depend on many influences. Key factors that influence future equity prices can be broadly divided into quantitative and qualitative types. Primary quantitative factors include open, high, low, close and volume data for individual equities, market segments (equity groups), indexes and exchange markets as a whole. Qualitative factors include socio-economic, political, international, regional and performance factors to name but a few.

Due to the difficulty in accurately retrieving and quantifying historical qualitative factors, network inputs used in the model presented here have been confined to readily available quantitative data. However, from quantitative factors the key qualitative factor of the market sentiment can be derived. Market sentiment tells us if the market is bullish, where high of confidence and rising prices prevail, or bearish, where there is a lack of investor confidence and prices are in decline. Thus historical data quantitatively reflects qualitative market sentiment to some extent which in turn should give indication of future price movements.

Simulation data was sourced from Yahoo! Finance [2]. The data used for network training and verification is comprised of daily figures for individual equities listed on the New York Stock Exchange (NYSE) from January 1st 1985 to December 31st 2000.

Traditional trading systems are almost exclusively mechanical systems that apply mathematical formulae to securities data to produce turnery buy, sell and hold indicators. The weakness of this traditional approach is that the trading system must be programmed to make explicit use of certain trading rules. Conversely it is hypothesised that neural networks should be able learn from training data and in turn make use of data that is intrinsically present in the input data set.

The primary aim of this paper was the creation of a neural network that, given a set of historical daily data, is capable of predicting the direction of the future price trend. The price trend prediction simply being whether market prices would on average increase or decrease relative to a subset of the daily training data.

In the simulation carried out in this paper ranges of network-parameter combinations were tested in order to determine the best arrangement for the network.

2 Design Considerations

Many considerations must be examined in developing a neural network. Attention needs to be given to these design considerations before beginning the network implementation phase. Design considerations include feasibility of the proposed network, data collection limitations, pre-processing and training. All of these aspects influence the effectiveness of the desired goals. Ultimately it must be remembered that neural networks are the tool or vehicle, but the objective must be both measurable and viable given available data and computational limitations.

2.1 Feasibility

Feasibility is an important factor in the assessment of any neural network project [3]. A primary feasibility aspect is that of information inherent in training data. Like any statistical tool neural networks are limited by the intrinsic information in input data. It is not possible to predict information that is not reflected in training set. A simple example of this is real world events, such as company announcements, that due to lack of pertinent information beforehand cannot be anticipated by the market.

Even in isolation of external factors it is unrealistic to presume that any set of historical data points inherently contains required information suitable for precise prediction of future market prices. At best a trading system, be it mechanical or artificial intelligence by design, can only aim to maximise pertinent information extraction from a historical data set. With a neural network approach we can at best hope the system derives information otherwise obscured in the training set.

2.2 Data Collection and Adjustment

As stated previously Yahoo Finance [2] was selected as the source of market information. Yahoo Finance is a well-established and reliable source of equity prices on markets around the world.

Common occurrences in equity markets are so-called stock splits and reverse splits. Price data has been pre-adjusted to reflect these price abnormalities. Without such correction trading data would experience unexpected price jumps up and down for reverse splits and splits respectively. For example if a stock selling at $2 were split on a 2:1 basis then a downward price jump of $1 would be shown after the split, while the opposite scenario with a jump from $2 to $4 would be true for a reverse split. Without adjustment this would create abnormalities in the training data thereby adversely effecting the training and performance of the neural network.

Training data length is important in order to correctly assess the systems ability to achieve accurate predictions in reasonable range of market conditions. 10 – 20 years is considered a reasonable range for system assessment [1] and in accordance with this the length of historical data used for this paper was 15 years. The 15 years of data across stocks listed on the New York Stock Exchange should provide balanced data with sufficient predictive power to forecast price trend movements as given in [3].

2.3 Pre-processing

Pre-processing data is essential to the learning and subsequent predictive ability of a neural network. Consider Figure 1. If a network were trained over section A of the then it would be unlikely to be able to generalise for the data covered in section B. The solution to this problem as the use of simple first order differences (equation 1) [4].

$$\delta(k+1) = x(k+1) - x(k) \tag{1}$$

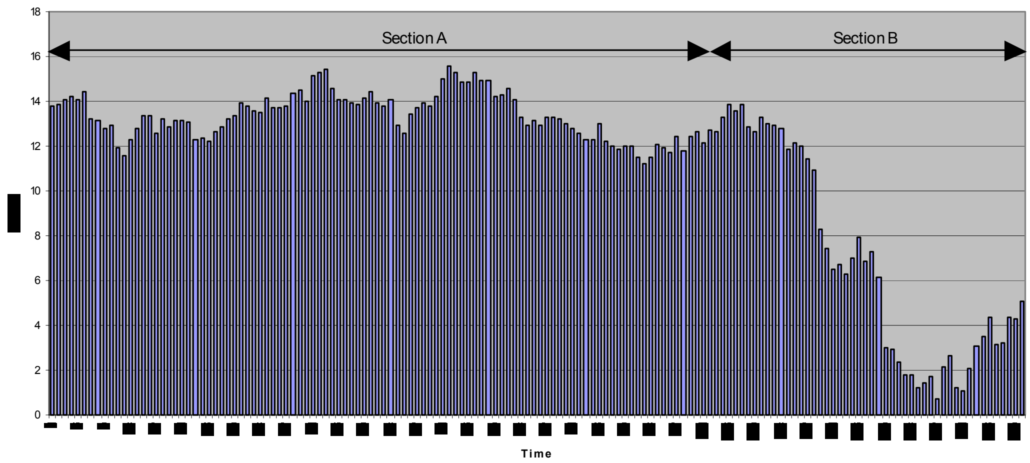

Figure 1: Example of difficulty in generalising from raw training data. Generalisation of section B from A is difficult given the entirely different trend of the data.

Normalisation is a key part of data pre-processing for neural networks and should enable more accurately predict future price trends. Consider Figure 2.

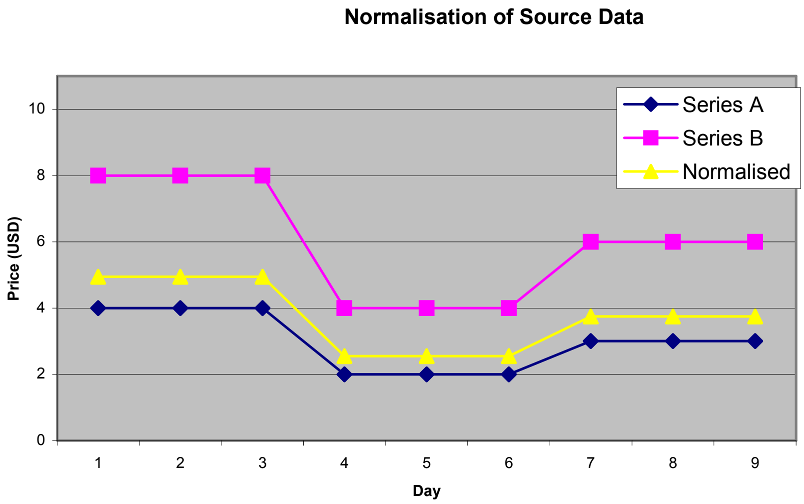

Figure 2: Normalisation of Source Data

In the above figure both equities follow the same price changes but on a different scale of magnitude. The important feature of data is the relative changes in daily stock prices rather than the absolute stock price [5]. Normalisation yields an identical price graph for both. Before normalisation both data sets exhibit the same qualities at different orders of magnitude. By normalising the data the trend prediction neural network can be trained to identify generic trends in data rather than specific data arrangements.

Another form of pre-processing implemented in this paper is logarithmic scaling. Logarithmic scaling makes better use of input data scope by evenly spreading data across the input range. This is a useful approach when reducing the effect of outliers [4]. Due to the sequential nature of equity price data and the importance of inter-day price changes outlier elimination, as outlined in [3] was not considered.

Pre-processing can greatly influence the effectiveness of a network. Careful pre-process selection can increase the success of a networks output. In line with this various combinations of pre-processing were trialed. The various pre-processing combinations are further explored under the implementation details (Section 3).

2.4 Dimensionality Reduction and Noise

The number of training examples required increases with the number of network weights. This sometimes-exponential increase puts strain on both data collection and computation requirements [3]. Because of this a Self Organising Map (SOM) stage was introduced to reduce the number of inputs to the neural network. This methodology groups input data into classifications according to data similarity, which in turn limits the number of input weights required.

The addition of noise to neural network training data helps to reduce the risk of overtraining therefore allowing the network to generalise. While raw market data is inherently noisy, data passed into the neural network from the optional SOM stage was assessed with various amounts of post-processing including both the addition of random noise as well as normalisation.

2.5 Training

Neural networks can suffer from over training. By over training a network its ability to generalise is diminished. This phenomenon is common to all forms of neural network training in addition to human tuning of mechanical trading systems [1]. A separate set of data was used for network error verification with network training being stopped once the verification set begins to increase [6].

3 Network Implementation and Architecture

The network architecture implemented in this paper is shown in figure 3.

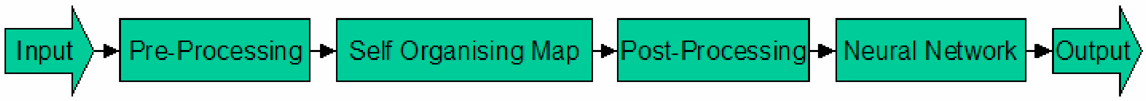

Figure 3: Neural Network Architecture

This basic design was proposed and implemented in [4] however the application has been transferred from international exchange markets to equity markets. Extension to this basic model has been performed in the ability of the network to undertake various types and combinations of neural network components and parameters. Pre-processing has been extended to include a wider range of alternatives. The SOM stage has been trained with a broad range of dimensions in addition to a stage of post-processing before SOM data is fed into the main neural network. The neural network phase was simulated with both standard feed-forward and Elman network types.

The trend prediction neural network was programmed in Matlab and took 23 training parameters. A wrapper batch program was written that called the trend predication neural network training program with various combinations of parameters. This design allowed better control of trend prediction in addition to a separation of functionality.

3.1 Data Pre-processing

Due to the inherently noisy nature of raw data in addition to neural network limitations such as the curse of dimensionality, some level of pre-processing is required. In this implementation up to two stages of pre-processing were applied. To determine the effectiveness of various combinations of pre-processing each stage could also be turned off, in effect acting in a simple pass-through manner.

Stage one of pre-processing calculates simple or logarithmic differences between days. A simple difference calculates the percentage between the current and previous days (equation 2). The logarithmic difference performs a log transformation on the simple difference (equation 3). Logarithmic scaling also was undertaken on a simple difference between the current and first days trading (equation 4). Again a simple pass-through mechanism was allowed (equation 5).

$$h_5(i) = x(i) - x(i-1) \tag{2}$$

$$h_3(i) = sign(\sigma(i))(\log(|\sigma(i)|+1)) \text{ where } \sigma(i) = x(i) - x(i-1) \tag{3}$$

$$h_2(i) = sign(\sigma(i))(\log(|\sigma(i)|+1)) \text{ where } \sigma(i) = x(i) - x(1) \tag{4}$$

$$h_1(i) = x(i) \tag{5}$$

Stage two of pre-processing implemented pre-process normalisation (equation 6).

$$x_i(i) = \frac{x(i) - mean(X)}{stddev(X)} \text{ where } X = x_1, x_2, \ldots x_{chunksize} \tag{6}$$

The combination of these two stages allow for a total of 10 combinations of pre-processing. Testing of all combinations allowed for determination of the more optimal form of pre-processing.

3.2 Self Organising Map & Post Processing

A feature extracting Self Organising Map stage was optionally used to cluster similar input data. This allows for a large reduction in the number of inputs to the NN. Each SOM works on one set of data only. SOM were used on a variety of combinations of source data and pre-processing functions. Self Organising Maps are able to organise input data into similar groups. This organisation reduces the number of neural network inputs, in turn limiting the effects of the curse of dimensionality. The size of the SOM was varied between 1-by-4, 1-by-8, 5-by-5 and 8-by-8. The SOM stage of the network was optionally removed.

Two SOM post-processing phases were added to the network design. The first stage of normalisation introduced noise to the SOM output while the second optionally normalised the output of the SOM to a standard deviation of one and mean zero, similar to that performed in (equation 6).

3.3 Neural Network

Two varieties of neural network tested were feed-forward back propagation & Elman back propagation, representing non-recursive and recursive neural network types respectively. The input layer to the neural network was varied to include a number of data chunks either fed from the SOM or where no SOM was not used, bipassing the SOM stage. These data chunks were allowed to be overlapping. Each chunk is created by taking a sample of *chunksize* days. An increment factor allows chunks to overlap i.e. if the *increment* factor is smaller than the *chunksize* then the next chunk will overlap by (*chunksize* – *increment*) days. For example three chunks of increment 10 and size 20 would cover 40 days as illustrated in Figure 4.

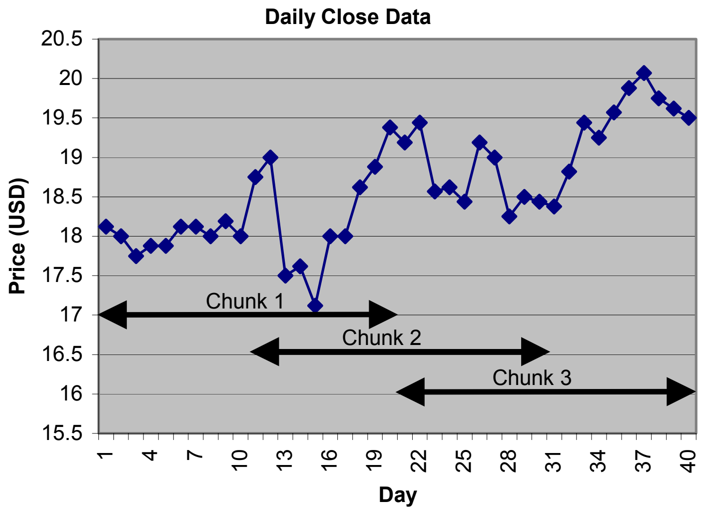

Figure 4: Three Chunks of 20 days with 10-day increment

The number of *chunks* and *chunksize* parameters were kept constant at 4 and 10 respectively for the simulation.

The Elman network was chosen as suggested by [4]. Elman networks have feedback to all hidden nodes from each hidden node (Figure 5). For comparative reasons a standard feed-forward network was also used in the simulation. A feed forward network has no internal state memory therefore its predictive ability is limited to the data provided to the inputs at the current time instance.

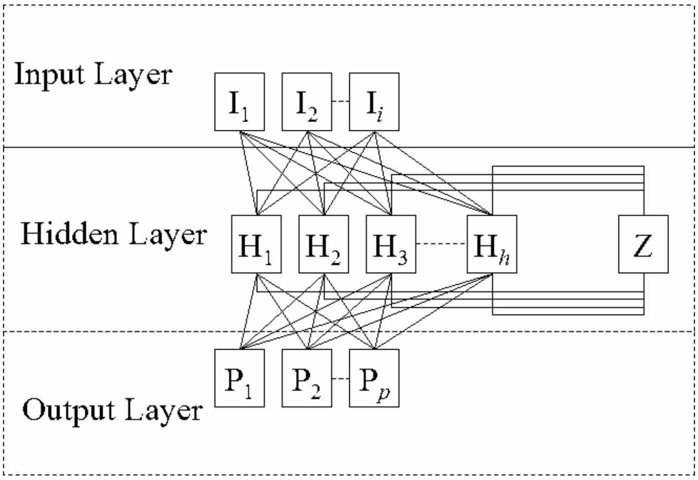

Figure 5: Elman Network

Various parameters for the neural networks were experimented with during training. Only one layer of hidden nodes was used in the experimentation. The number of nodes in the hidden layer was modified between 3, 5 and 10.

The transfer function used by the neural network was *tansig* (Fig 6). The transfer function was not modified during the training of the network.

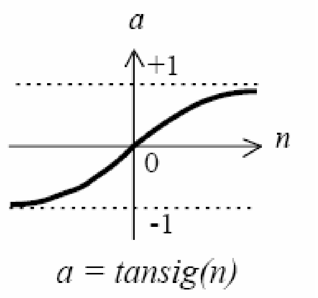

Figure 6: Tan-Sigmoid Transfer Function

3.4 Result Analysis

The network was trained to predict if a defined number of days ahead of time would be, on average, higher or lower than reference days in the training set. Thus the network is only expected to predict a 'higher' or 'lower' result from training data. Outputs to the neural network were tested in both singular and dual arrangements. The singular configuration has only one output from the neural network; this output is set to near-one for a 'higher' prediction and near-zero for a 'lower' prediction (equation 7). Singular output configuration is classified as correct if both the predicted and actual future values are both higher or lower than 0.5 (equation 8).

$$T_x = \begin{matrix} 0.8 & , D_x > D_{x-1} \\ 0.2 & , D_x \leq D_{x-1} \end{matrix} \tag{7}$$

where

D_x = Daily price data for day x
T_x = Target trend direction for day (x –1) to day x

$$Correct = \begin{matrix} 1 & , R_i > 0.5 \wedge T_i > 0.5 \\ 1 & , R_i \leq 0.5 \wedge T_i \leq 0.5 \\ 0 & , otherwise \end{matrix} \tag{8}$$

where

R_x = Predicted trend direction for day (x –1) to day x

Dual outputs were simply set to alternate values (one high, one low) depending on if the target price was higher or lower than the reference days (equation 9). Classification of correct results was categorised if actual and predicted results were of the same polarity (equation 10). The target and predicted trend are defined as primary and secondary for reasons of clarity.

$$U_x = \begin{matrix} 0.2 & , D_x > D_{x-1} \\ 0.8 & , D_x \leq D_{x-1} \end{matrix} \tag{9}$$

where

U_x = Secondary Target trend direction for day (x –1) to day x
D_x = Daily price data for day x

$$Correct = \begin{matrix} 1 & , R_i > S_i \wedge T_i > U_i \\ 1 & , R_i < S_i \wedge T_i < U_i \\ 0 & , otherwise \end{matrix} \tag{10}$$

where

A correct prediction is defined as 1 and incorrect prediction is defined as 0
R_x = Primary Predicted trend direction for day (x –1) to day x
S_x = Secondary Predicted trend direction for day (x –1) to day x
T_x = Target trend direction for day (x –1) to day x

For comparative purposes simple mechanical experimentation was carried out on the sample data collected. The intention of this analysis was to verify if a simple moving average could predict the next trading day. This tested whether the moving average increase from the previous day to the current day predicted that the following day would also be higher than the current (equation 11) and visa versa for lower days. This test was carried out based on a 1, 5 and 15-day moving averages.

$$R_x = \begin{matrix} 1 & ma_x > ma_{x-1} \\ 0 & ma_x < ma_{x-1} \end{matrix} \tag{11}$$

where

R_x = Predicted trend direction
ma_x=moving average of daily values

4 Results

Due to the enormous number of combinations of parameters that could be fed into the neural network a batch layer was used to cycle through the possible combinations. Limitations in available computing time limited the number of combinations that could be tested. Fortunately the addition of the batch-processing layer revealed quickly which parameters greatly impacted the performance of the network and which had little or no effect.

4.1 Neural Network Outputs

Outputs from the neural network were either dual or singular. It was found that a singular output gave vastly superior performance to that of dual outputs. In order to reduce the computational requirements the follow results have been limited to a singular network output only.

4.2 Data Pre-Processing

Both stages of pre-processing proved highly important in the predictive ability of the network. Results revealed that in the first stage of pre-processing, performance was not significantly altered between the simple-difference and logarithmic-difference pre-processing methodologies. However the performance of the network was negatively affected by setting this first pre-processing stage to pass-through. The average percentage of correct predictions with pass-through was just 55.7% versus 72.7% for other forms of pre-processing.

Normalisation of the second stage of pre-processing, proved to have a significant impact on the results of the neural network. Without normalisation the network predicted results with a lower accuracy of around 55% and narrow standard deviation of approximately 3% (Table 1). By contrast with normalisation results gave a much higher average accuracy in the range of 73% but with a wider standard deviation of approximately 15%.

Pre-Process (Stage 1)	Pre-Process Normalisation (Stage 2)	Average Success (%)	Standard Deviation (%)
None	N	55.57	2.64
None	Y	55.85	2.98
Relative Logarithmic Difference	Y	73.60	14.68
Origin Logarithmic Difference	N	56.14	3.62
Origin Logarithmic Difference	Y	72.03	15.16
Relative Simple-Difference	Y	73.18	14.87

Table 1: Effectiveness of Pre-processing

These results show the effectiveness of normalisation on the output of the neural network (table 1). Without normalisation results were significantly degraded. Additionally without any form of differencing correct predictions were also low. It is interesting to note that while normalisation and pre-processing yielded a higher average result the subsequent larger standard deviation shows that the consistency of the result was reduced.

4.3 Self Organising Maps

The addition of post-processing normalisation to the SOM did not yield a significant improvement or degradation in predictive ability of the network (table 2).

Pre-Processing	SOM Post-Processing	Average Success (%)	Standard Deviation (%)
Relative Simple-Difference	None	56.46	3.98
Origin Logarithmic-Difference	None	58.94	5.23
Relative Simple-Difference	Normalised	58.18	3.30
Origin Logarithmic-Difference	Normalised	57.97	3.22
Relative Logarithmic-Difference	Normalised	58.13	3.51

Table 2: Effectiveness f Post-Processing Normalisation on Self Organising Maps

The effective of adding varying amounts of noise to the SOM was examined. The results show a minor drop in the average success of the neural networks predictive ability as noise is added (table 3).

Average Percentage Error Added	Average Success (%)	Standard Deviation (%)
0%	58.47	3.77
2.5%	58.41	3.36
5.0%	58.20	3.76
12.5%	57.21	3.40

Table 3: Effect of Introduced Error on SOM Stage

Changes to the size of the SOM showed accuracy of the network output was improved with a larger 1-dimensional SOM (table 4).

SOM Size	Average Success (%)	Standard Deviation (%)
1 x 4	57.82	3.56
1 x 8	58.30	3.64
5 x 5	51.86	1.47
8 x 8	55.74	3.29

Table 4: Effectiveness of SOM dimensions

Removal of the SOM from the network architecture was also tested. The results showed a significant increase in the average success of the network. This result is in contrast to proposed by Giles, Lawrence and Tsoi (2001). Note that SOM noise and normalisation post-processing phases are of course not used when the SOM phase is not present. Origin relative logarithmic differencing yielded a slightly worse average success rate when compared to other forms of differencing. Additionally the consistency of the result was compromised by the removal of the SOM layer as revealed by an increase in standard deviation of results.

Pre-Processing	SOM Post-Processing	Average Success (%)	Standard Deviation (%)
Relative Simple-Differnce	No SOM	77.32	14.05
Origin Logarithmic-Difference	No SOM	73.30	15.17
Relative Logarithmic-Difference	No SOM	77.58	13.81

Table 5: Average Success with SOM Stage Removed

The trend prediction network was very successful in its ability to predict price direction with prediction ability averaging 77%.

4.4 Neural Network Type and Configuration

The network type was modified with and without a SOM stage to show the impact this would have. Results from simulation showed no significant difference in performance between Elman and standard Feed-Forward network topologies. Results demonstrated a direct correlation between the number of hidden nodes and average predictive ability when input was taken from the SOM stage. Conversely when input was taken directly from the pre-processed data no significant difference in average performance was demonstrated however a slight improvement in variation of the results was observed (Table 6).

Number of Hidden Layers	NN Type	SOM	average success (%)	standard deviation (%)
3	Elman	Y	58.35	3.34
5	Elman	Y	57.23	3.82
10	Elman	Y	52.68	2.40
3	Feed-Forward	Y	57.85	2.92
5	Feed-Forward	Y	58.77	4.13
10	Feed-Forward	Y	52.69	2.54
3	Elman	N	76.47	15.63
5	Elman	N	75.88	14.02
10	Elman	N	76.47	13.68
3	Feed-Forward	N	76.13	15.11
5	Feed-Forward	N	76.03	13.96
10	Feed-Forward	N	75.06	13.30

Table 6: Effect of Neural Network Type and Number of Hidden Layers on Network Performance

Finally the effects of predictive range and reference were examined. Predictive range is the range of trading days over which the neural network is to predict the future trend. Predictive reference is the historical trading-day range used for calculating relative trend movement measures in days relative to the last trading day. For example a predictive range and reference of 1-5 and 1-16 define that the network will try to predict whether the next 5 trading days (predictive range) will be higher or lower on average than the previous 16 trading days (predictive reference).

The results of varying the predictive range (horizontal) and predictive reference (vertical) of the neural network are shown below for 3, 5 and 10 hidden nodes (Table 7,8 and 9 respectively). The below data was obtained without inclusion of the SOM stage due to the higher predictive ability of a SOM-free network architecture. Averaging was used to simplify data representation.

	1-1	1-2	1-5	1-10	Average
1-20	91.41	87.15	72.49	68.57	79.90
1-16	88.45	83.90	74.37	66.39	78.28
1-6	92.85	76.08	62.83	63.72	73.87
1-3	86.27	74.59	63.38	65.06	72.33
1-1	92.50	76.55	69.79	66.49	76.33
Average	90.30	79.66	68.57	66.05	76.14

Table 7: Network Predictive Success with 3 Hidden Neurons

	1-1	1-2	1-5	1-10	Average
1-20	91.21	84.36	73.65	68.54	79.44
1-16	90.09	78.52	71.45	66.94	76.75
1-6	95.25	80.61	68.68	63.00	76.88
1-3	88.16	79.33	69.95	65.62	75.76
1-1	94.35	79.22	66.37	64.29	76.06
Average	91.81	80.41	70.02	65.68	76.98

Table 8: Network Predictive Success with 5 Hidden Neurons

	1-1	1-2	1-5	1-10	Average
1-20	97.26	82.24	73.64	64.90	79.51
1-16	97.23	81.98	72.57	65.76	79.38
1-6	97.38	80.14	71.75	57.67	76.73
1-3	95.52	76.94	72.55	62.45	76.86
1-1	91.50	77.21	63.7	63.50	73.98
Average	95.78	79.70	70.84	62.86	77.29

Table 9: Network Predictive Success with 10 Hidden Neurons

Figure 7 shows the plotted averages for 3, 5 and 10 hidden layers across varying predictive ranges. Graphical representation of averages clearly shows the detrimental effect of an increased predictive range on the networks predictive ability.

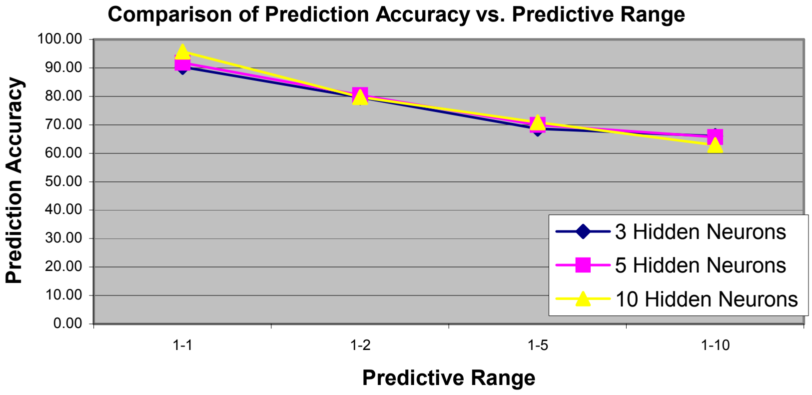

Figure 7: Predictive Accuracy vs. Predictive Range for Neural Network

Results of changes in predictive reference (Figure 8) show an increase in the predictive ability of the network as the size reference group is increased.

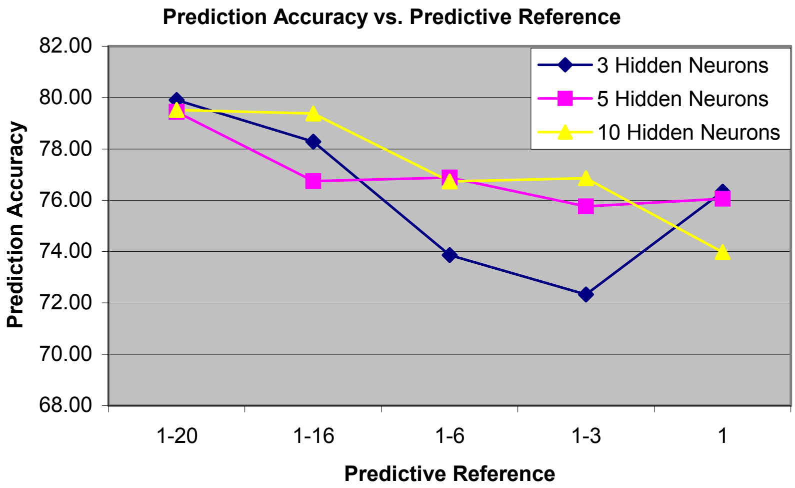

Figure 8: Predictive Accuracy vs. Predictive Reference for Neural Network

4.5 Mechanical Averaged-Based Prediction

Mechanical prediction was carried out to verify if the network was simply carrying on the prevailing trend as described in 3.4 (Table 10).

Average Type	Average Success (%)
1-day	51.01
5-day	64.32
15-day	57.53

Table 10: Simple Average Trend Prediction

This result demonstrates the effectiveness of a simple mechanical system. While an average success rate of 64% was achieved by the simple mechanical system when using 5-day averaging it falls short of the result obtained by the neural network. The trained neural network was able to achieve a much higher level of accuracy with an average success rate exceeding 94% for a 1-1 predictive reference and range (Table 8). This demonstrates that the neural network was performing more than simple averaging to achieve its result.

5 Conclusion

The results of experimental simulation have given tangible results for both pre-processing requirements and general network architecture. Results show that raw data should always undergo some form or differencing, be it between the previous a day or a fixed reference point. The use of logarithmic function on this differenced data yielded no significant increase or decrease in the networks predictive ability. Furthermore normalisation across the input window proved to be a critical element of the pre-processing procedure.

Self Organising Maps were found to decrease the networks predictive abilities. This result could be attributed to an over-simplification of training data. Information inherently in the training set was likely not reflected in SOM classification.

When using SOM’s it was found that the addition of a post-processing phase was unnecessary. This post-processing phase added output normalisation and noise addition to the SOM-output data. While it was hypothesized that noise would reduce effects of over-fitting and therefore improve performance it was found not to be the case in this circumstance. Noise was not added to input data as it is already inherently noisy. It was however found that when using a SOM stage the size of the SOM had only a slight effect on the quality of the final neural network output.

Both Elman and standard Feed-Forward networks performed almost equally in the simulations. This suggests that information needed to predict the immediate direction of future prices is limited primarily to data contained in recent trading statistics. Comparatively long term (seasonal) trends would very likely require more memory or a lower resolution such as weekly instead of daily statistics.

When the SOM stage was used an increase in the number of hidden nodes in the second dimension resulted in a slight decrease in predictive ability of the network. However when the SOM stage was excluded from the network an increased number of nodes resulted in a very slight reduction in the variation of the network output.

In conclusion making use of normalised relative inputs into a neural network is significant in gaining an increase result. A SOM stage in processing gives worse results when predicting trend; however when a SOM stage was used it was best to keep a one-dimensional structure. Noise and normalisation added to the SOM stage had no effect on the predictive ability. Network type has no effect on network output, however the use of more hidden layers proved beneficial to output variation when not using a SOM layer and detrimental to average predictive ability when using a SOM layer.

The optimal configuration of the network has proven to be one without a SOM layer. Pre-processing should include relative-normalisation. To minimise the output variation an increased number of hidden layers should be used. Best results were yielded with prediction calculated relative to a wider trading period for the following trading day only. This implies the difficulty in predicting multiple days into the future with such a model; however next day price prediction, a more useful measure in real world applications, also yielded highly favorable results. When compared to a simple mechanical averaging form of trend prediction the neural network trained achieved on average a higher success rate. Simple mechanical average trend prediction was only able to predict with 65% accuracy versus the optimally trained networks with predictive ability exceeding 90%.

6 Future Research Direction

Further research will utilise current results in serving as a foundation for a hierarchical neural network structure (Figure 9). Under this network topology many sub-networks are trained on focused and independent tasks. An example of one of these sub-networks is the trend prediction network developed in this paper. The funneling of these multiple networks is beneficial due to an increase in training speed and reducing training data requirements over one larger network [7]. Each neural network allows a more focused form of prediction that doesn't rely upon complex formation of a predictive function from the input data set. Through combination of individual network results a higher predictive ability can be obtained.

Components of this future network will rely upon additional and more complex pre-processing. Much of the pre-processing in the future will likely be based on technical indicators. The aim of this future research will be the development of a trading system with the ability to predict Buy, Sell and Hold price points. The quality of system output from such a system can be easily measured by the net profit or loss made by the system after accounting for transactional costs i.e. brokerage.

Additional raw data will be necessary for such a system. This data will come in the form of foreign exchange and sector indexes. Such indexes will allow the system to account for international and sector qualitative events in a limited fashion. Due to the time difference between various markets international indexes should signal major world events and allow the system to take this into account when making a daily position decisions. Indexes of sector should signal trends of various sectors such as mining, agriculture and banking. These trends can assist in decisions of distribution of funds across sectors.

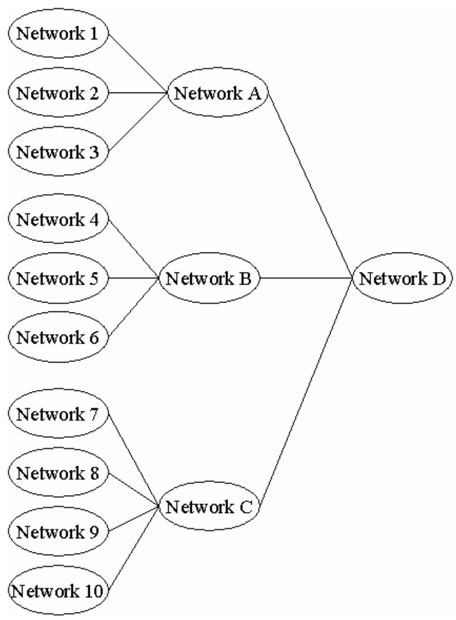

Figure 9: Hierarchical Network Topology

Results presented in this paper are primarily exploratory. The aim of this initial phase of simulations was the discovery of better network topologies. The trend prediction system however has proven to be highly successful and will be used in later phases of the research.

7 References

[1] Schwager, J. (1999). Getting Started in Technical Analysis. John Wiley & Son. p.2,253-258

[2] Yahoo Finance. http://finance.yahoo.com

[3] Swingler, K. (1996). Applying Neural Networks a Practical Guide. University of Stirling, Stirling, Scotland and Neural Innovation Limited. p. 24-25,106, 191.

[4] Giles C., Lawrence S. and Tsio A. (2001). Noisy Time Series Prediction using a Recurrent Neural Network and Grammatical Inference. *Machine Learning*, Volume 44, Number 1/2, July/August. p.7,161-183.

[5] Pring, M. (2001) Introduction to Technical Analysis. McGraw-Hill. p.114-115

[6] Shadbolt, J. (2002). Overfitting, Generalisation and Regularisation. In Shadbolt J. and Taylor J. (Eds) Neural Networks and the Financial Markets. Springer-Verlag. p.56

[7] Mendelsohn, L. (1993) Using Neural Networks For Financial Forecasting. *Stocks & Commodities*, Volume 11:12, October. p.518-521