Application Of Cascade-Correlation Neural Networks In Developing Stock Selection Models For Global Equities

Kathleen Hodnett, PhD, University of the Western Cape, South Africa
Heng-Hsing Hsieh, PhD, CFA, University of the Western Cape, South Africa

ABSTRACT

We investigate the potential of artificial neural networks (ANN) in the stock selection process of actively managed funds. Two ANN models are constructed to perform stock selection, using the Dow Jones (DJ) Sector Titans as the research database. The cascade-correlation algorithm of Fahlman and Lebiere (1990/1991) is combined with embedded learning rules, namely the backpropagation learning rule and the extended Kalman filter learning rule to forecast the cross-section of global equity returns. The main findings support the use of artificial neural networks for financial forecasting as an active portfolio management tool. In particular, fractile analysis and risk-adjusted return performance metrics provide evidence that the model trained via the extended Kalman filter rule had greater strength in identifying future top performers for global equities than the model trained via the backpropagation learning rule. There is no distinguishable difference between the performances of the bottom quartiles formed by both ANN models. The zero-investment portfolios formed by longing the top quartiles and simultaneously shorting the bottom quartiles or the market proxy exhibit statistically significant Jensen's alpha and continues to accumulate positive returns over the out-of-sample period for both ANN models. On the other hand, the zero-investment portfolios formed by longing the bottom quartiles and simultaneously shorting the market proxy exhibit statistically significant Jensen's alpha and continues to accumulate losses over the out-of-sample period for both ANN models. The implementation of the extended Kalman filter rule in training artificial neural networks for applications involving noisy financial data is recommended.

Keywords: Stock Selection; Firm-Specific Attributes; Artificial Neural Networks

INTRODUCTION

Traditional financial asset pricing models such as the capital asset pricing model (CAPM) of Sharpe (1964) and Lintner (1965) and the arbitrage pricing theory (APT) of Ross (1976) are based on the assumption that a linear pricing relation between stock returns and the respective explanatory variables exist. A growing body of evidence has questioned this relationship suggesting that perhaps a nonlinear pricing relation, in addition to this linear relation might exist, particularly in the pricing relationship based on firm-specific attributes.[1] Thus, if one were to relax this assumption and allow for a nonlinear pricing relation, in addition to the linear pricing relation, models incorporating this nonlinearity could be constructed. An artificial neural network (ANN) is one such technique having the ability to map this nonlinear relation and potentially improve forecasting accuracy. Artificial neural networks, having its roots in cognitive psychology and computational neuroscience, are designed to mimic the functioning of the human brain. Like the human brain an ANN is often described as a nonlinear, parallel computer responsible for all the information-processing abilities that we possess (Attew, 2002).

[1] See for example Banz (1981); Fama and French (1992, 1993); Van Rensburg and Robertson (2004); Eakins, Stansell and Buck (1998, 2003); Cao, Leggio and Schniederjans (2005) and Cao, Parry and Leggio (2009), to name a few.

In this paper we build stock selection models to forecast stock returns and examine the potential of ANNs in the stock selection process of actively managed funds. We combine the cascade-correlation algorithm of Fahlman and Lebiere (1990) with embedded learning rules, namely the backpropagation learning rule and the extended Kalman filter learning rule, to forecast the cross-section of global equity returns. The cascade-correlation algorithm is a supervised learning algorithm that serves a two-fold purpose of determining the neural network architecture and is responsible for the learning process by modifying the network architecture and adding hidden neurons to the network until the optimal network is selected. This contrasts fixed-architecture structures which requires the network architecture to be clearly specified before any training (weight adjustment) of the network takes place. Much of the research into the financial application of ANNs involves the construction of fixed-architecture networks trained via the gradient descent backpropagation learning algorithm.[2] An extended Kalman filter is introduced in this research as an alternative to backpropagation for selecting stocks. The rationale for this lies in the fact that financial data is often noisy, thus an algorithm that specifically adjusts for this noise could improve the ANN model's ability to generalize. The extended Kalman filter, a nonlinear version of the Kalman filter proposed by Kalman (1960) is used to estimate nonlinear systems. Correlation analysis and fractile analysis is conducted to enhance the comparison between the predictability of the respective ANN models.

A series of financial ratios and firm-specific attributes are considered as candidate model inputs to predict global equity returns. A genetic algorithm is employed to select the best subset of input variables in the respective ANN models.

The main findings of this research support the uses of artificial neural networks for stock return forecasting as an active portfolio management tool. The use of the extended Kalman filter in training artificial neural networks for applications involving noisy financial data is recommended.

LITERATURE REVIEW

Over the past decades researchers have put forward a case for the use of ANNs in forecasting stock returns from a set of firm-specific variables such as firm size, book-to-market ratio, price-to-earnings ratio, to name a few. Research conducted by Eakins, Stansell and Buck (1998) is one of the first studies to examine nonlinearities in financial data for this purpose. The sample includes companies listed on the New York stock Exchange (NYSE), American Exchange (AMEX) or those trades through the national association of security dealers over the period 1988 to 1991. An ANN model is constructed which includes the lagged values of ten firm-specific variables as inputs, namely, net profit margin, operating profit margin, current ratio, total asset turnover, debt-to-asset ratio, return-on-assets, price-earnings ratio, market value, trading volume and percentage ownership as the model output. The model is trained via the gradient descent backpropagation learning algorithm and then compared to a Tobit regression model. In terms of predictive power, the ANN model outperforms the Tobit regression model over the examination period.

In a similar study Eakins and Stansell (2003), in an attempt to determine whether value stocks provide superior investment returns, construct an ANN model using the descriptors of value, that is, price-to-cash flow, price-to-book, dividend yield, earnings yield, sales and market capitalization as inputs in the neural network to forecast returns for stocks listed on COMPUSTAT over the period from 1975 to 1996. The model is trained using the backpropagation algorithm. The results provide evidence that the risk-adjusted returns of stocks selected by the ANN model are greater than the results achieved by other forecasting models. Consistent with Fama and French (1992, 1993) value stocks provide higher returns with lower risk than can be obtained from the random walk process.

Cao, Leggio and Schniederjans (2005) examine the influences of firm-specific attributes in pricing stocks traded on the Shanghai Stock Exchange over the period from 1999 to 2002. ANN models trained via the backpropagation model outperform the linear counterparts in terms of their predictive power. Cao, Parry and Leggio

[2] Past financial research employing this technique includes Abhyankar and Wong (1997) and Kanas and Yannopoulos (2001) for the forecasting of stock market index returns. Other applications employing this technique include research conducted by Singleton and Surkan (1990) for bond classification; Utans and Moody (1995) for bond rating; Qi and Maddala (1995) and White (1995) for option pricing and Swanson and White (1995a, 1995b) for macroeconomic forecasting.

(2009) extend the methodology of Cao, Leggio and Schniederjans (2005) by comparing three linear models with three ANN models on the Shanghai Stock Exchange over two sub periods, namely, 1999 to 2002 and 2003 to 2008. The models include a linear and an ANN univariate time series model; a multivariate linear and nonlinear version of the CAPM (where stock returns are regressed on market returns); and the linear and nonlinear version of Fama and French (1993) three-factor model. The results confirm the forecasting superiority of the ANN models relative to the linear models in forecasting Chinese stock returns.

Other tests that support the use of ANN models in financial research include studies conducted by Hung, Liang and Liu (1996) and Kana and Yannopoulos (2001). Hung, Liang and Liu (1996) examine the feasibility of integrating the APT and ANN models to forecast stock returns on the Taiwan Stock Exchange. The results provide evidence that the integrated approach provides a more optimal solution than when the APT or ANN is used alone. The integrated model outperforms the market benchmark, as well as other integrating techniques such as blending ARIMA with APT. Kanas and Yannopoulos (2001), on the other hand, attempt to forecast movements in the Dow Jones (DJ) and Financial Times (FT) indices, using dividends and trading volume as the explanatory variables for the linear and ANN models. The study discovered an underlying nonlinear relation between stock returns and fundamental attributes.

FUNDAMENTALS OF ARTIFICIAL NEURAL NETWORKS IN ASSET PRICING

Neural Network Architectures

Fixed Architecture Networks

An artificial neural network is made up of a group of artificial neurons, with weighted interconnections. Structured in layers with parallel processing, there exists an input layer, one or more hidden layers and an output layer. With fixed-architecture networks, every neuron is specified in advance, prior to network training and does not change. Feed-forward neural networks are arranged in layers where information passes in one direction from input to output layer via interconnection weights. Each neuron in the input layer is connected to every neuron in the hidden layer, which in turn is connected to the output layer. There are no interconnections between neurons occupying the same layer. A feed-forward neural network with multiple neurons occupying multiple hidden layers may exist for certain applications which are more complex or chaotic. According to McNelis (2005), the single layer feed-forward network with one hidden layer is the most popularly used neural network in both economic and financial applications. Equation 1, adapted from Hodnett (2010) (modified from Kanas and Yannopoulos (2001)) is the algebraic expression of a standard fixed-architecture ANN model with direct connections between the input neurons and output neuron that is used to estimate $E(R_{i,t})$, the expected return of share i in month t.

$$E(R_{i,t}) = a_{i,0,t} + \sum_{j=1}^{J} g_{i,j,t} F_{j,t-1} + \sum_{k=1}^{K} b_{i,k,t} f\left(d_{i,0,t} + \sum_{j=1}^{J} c_{i,j,k,t} F_{j,t-1} \right) \tag{1}$$

Where:

$a_{i,0,t}$	=	the bias term to the output layer for period t;
$g_{i,j,t}$	=	the direct payoff to input neuron $F_{j,t-1}$ from the output neuron (the equity returns) for period t;
$F_{j,t-1}$	=	the input neuron representing the transformed lagged value of attribute j in the input layer;
$b_{i,k,t}$	=	the factor payoff to the hidden neuron k from the output neuron for period t (it measures the sensitivity of the equity returns to movements in hidden neuron k);
$d_{i,0,t}$	=	the bias term to the hidden layer for period t;
$c_{i,j,k,t}$	=	the factor payoff to input neuron $F_{j,t-1}$ from the hidden neuron k for period t (it measures the sensitivity of hidden neuron k to movements in input $F_{j,t-1}$);

$\sum_{j=1}^{J} g_{i,j,t} F_{j,t-1}$ = the linear function;

$f\left(d_{i,0,t} + \sum_{j=1}^{J} c_{i,j,k,t} F_{j,t-1} \right)$ = the nonlinear activation function (the sigmoid function);

and

$d_{i,0,t} + \sum_{j=1}^{J} c_{i,j,k,t} F_{j,t-1}$ = the input signal within the nonlinear activation function.

Based on Equation 1, the expected return determination process receives signals from the linear function which is the sum of the products of the input neurons and their associated factor payoffs (that is, $\sum_{j=1}^{J} g_{i,j,t} F_{j,t-1}$). It also receives signal from the nonlinear transformation function. Each input neuron $F_{j,t-1}$ takes on the raw lagged value of the attribute *j*, or a transformed lagged value of the attribute *j* via one or more transformation functions. The input signal, $d_{i,0,t} + \sum_{j=1}^{J} c_{i,j,k,t} F_{j,t-1}$, has to be activated in order to reflect the nonlinearity. The sigmoid logistic cumulative distribution function is employed for the research.[3] The nonlinear transformation function, $f\left(d_{i,0,t} + \sum_{j=1}^{J} c_{i,j,k,t} F_{j,t-1} \right)$ of the ANN models can thus be redefined as $\frac{1}{\left(1 + e^{-\left(d_{i,0,t} + \sum_{j=1}^{J} c_{i,j,k,t} F_{j,t-1} \right)} \right)}$.

A feed-forward network, together with the sigmoid activation function is referred to as a multilayer perceptron (MLP) network. Equation 2, adapted from Gonzalez (2000) and Hodnett (2010), displays an example of an ANN model possessing 2 input neurons and 2 hidden neurons in a single hidden layer. In Equation 2, the input neurons are the lagged values of the firm-specific attributes of the sample shares, and the output neuron is the expected returns of the sample shares in time period *t*.

$$E(R_{i,t}) = a_{i,0,t} + (g_{i,1,t} \times F_{1,t-1}) + (g_{i,2,t} \times F_{2,t-1}) + \left(\frac{b_{i,1,t}}{1 + e^{-(d_{i,0,1,t} + c_{i,1,1,t} \times F_{1,t-1} + c_{i,2,1,t} \times F_{2,t-1})}} \right) + \left(\frac{b_{i,2,t}}{1 + e^{-(d_{i,0,2,t} + c_{i,1,2,t} \times F_{1,t-1} + c_{i,2,2,t} \times F_{2,t-1})}} \right) \tag{2}$$

The advantage of employing this model is that it separates the contribution of the nonlinearity from that of the linearity. The nonlinear factor payoffs, $b_{i,1,t}$ and $b_{i,2,t}$ in Equation 2 are equal to zero if the share returns do not exhibit any nonlinear traits.

Figure 1 provides a representation of Equation 2. ($g_{i,1,t} \times F_{1,t-1}$) + ($g_{i,2,t} \times F_{2,t-1}$) represents the input signal from the linear transformation function to the output. The input signals, ($d_{i,0,1,t}$ + $c_{i,1,1,t} \times F_{1,t-1}$ + $c_{i,2,1,t} \times F_{2,t-1}$) and

[3] The sigmoid logistic cumulative distribution function $f(x)$ is defined as $1/(1+e^{-x})$. Other activation functions include a *linear*, *Tanh*, *Gaussian* and *Sine* transfer function.

($d_{i,0,2,t}$ + $c_{i,1,2,t} \times F_{1,t-1}$ + $c_{i,2,2,t} \times F_{2,t-1}$), are first transformed into the nonlinear format using the sigmoid logistic cumulative distribution function before they are received by the output neuron. Next, the hidden neurons, $H_{1,t}$ *and* $H_{2,t}$ are then established with their values being depicted by $\left(\frac{1}{1+e^{-(d_{i,0,1,t}+c_{i,1,1,t}\times F_{1,t-1}+c_{i,2,1,t}\times F_{2,t-1})}}\right)$ and $\left(\frac{1}{1+e^{-(d_{i,0,2,t}+c_{i,1,2,t}\times F_{1,t-1}+c_{i,2,2,t}\times F_{2,t-1})}}\right)$ respectively. The final signal of the nonlinear transformation function connecting the hidden layer and the output neuron, $E(R_{i,t})$, is equal to the sum of the products of the hidden neuron values and their factor payoffs, $\left(\frac{b_{i,1,t}}{1+e^{-\left(d_{i,0,1,t}+c_{i,1,1,t}\times F_{1,t-1}+c_{i,2,1,t}\times F_{2,t-1}\right)}}\right) + \left(\frac{b_{i,2,t}}{1+e^{-(d_{i,0,2,t}+c_{i,1,2,t}\times F_{1,t-1}+c_{i,2,2,t}\times F_{2,t-1})}}\right)$.

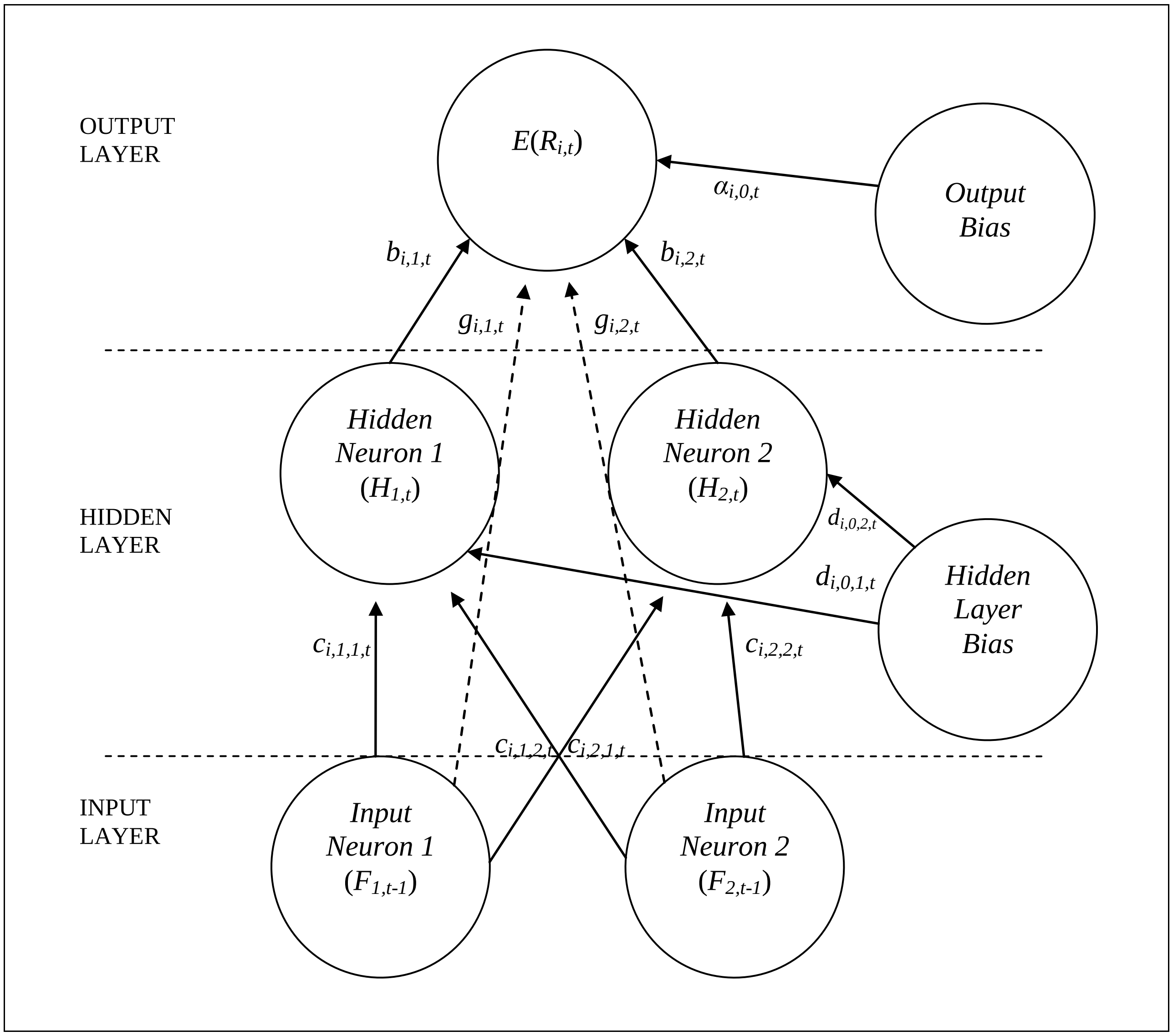

Figure 1: Fixed Architecture Neural Network (adapted from Hodnett (2010))

As previously mentioned fixed-architecture networks entail the predetermination of the appropriate input neurons, hidden layers and hidden neurons. Determining the exact number of hidden neurons to occupy a hidden layer varies from application to application and is often determined using trial and error, heuristics, or experimentation. This could severely impact the accuracy, robustness and generalization ability of the model.

Specifying too many hidden neurons could result in network overfitting, while too few hidden neurons could impact the ability of the network to adequately learn the training data. In order to overcome the problems associated with traditional fixed-architecture networks and the hidden layer problem, the cascade-correlation algorithm, was proposed by Fahlman and Lebiere (1990).

Cascade-Correlation Architecture

The cascade-correlation algorithm is a topology-modifying algorithm responsible for both network architecture and network learning. The objective is to establish the network architecture by adding hidden neurons one by one to the network as it learns. Once these hidden neurons have been added to the network they do not change. In installing a hidden neuron, many different networks are trained until the best one is selected. Unlike fixed architecture networks, no *a priori* network architecture needs to be specified in advance, since the architecture dynamically adjusts as more hidden neurons are installed. With cascade-correlation networks, unlike fixed architecture networks, depending on the complexity of the input data, more (or less) hidden neurons are added to the network, as the architecture evolves. The advantage is that the network learns quickly and it determines its own size and topology (Fahlman and Lebiere (1990), Diamtopoulou (2005), Hodnett (2010)). This cascaded architecture ensures that the neural network determines the actual number of hidden nodes and connection weights. The mechanics of the cascade-correlation algorithm is further discussed in the methodology section of this paper.

Neural Network Training

The goal of supervised learning is to minimize the discrepancy between the output of the network and the target/desired output by adjusting the network weights. The extended Kalman filter rule is tested, in addition to backpropagation in this research since it is a faster converging alternative to backpropagation and specifically adjusts for the presence of noise in financial data (Nechyba and Xu (1997)). Singhal and Wu (1989) were the first to test the extended Kalman filter in training neural networks. They find the extended Kalman filter to be superior to backpropagation for training multilayered networks. An alternate interpretation proposed by Ruck, Rogers, Kabrisky, Maybeck and Oxley (1992) is that backpropagation is simply a degenerate form of the extended Kalman filter. The backpropagation learning rule and the extended Kalman filter is embedded within the cascade-correlation architecture to train the network weights during cascade learning. Below is a description of each learning method.

Backpropagation in ANN Training

Backpropagation is a gradient descent algorithm in which the network weights move along the negative of the gradient of the performance function. Organized in a multilayer from input to output layer, the signal is propagated “forward”. The network error is then propagated “backwards” into the network via the backpropagation learning algorithm. This mechanism continues until the error is minimized. The training process starts by assigning random values (between -1 and +1) to the weights. The network computes an output which is dependent on the hidden neurons, activation function and network weights. The activation function (net_k) of the artificial neurons in ANNs can be represented as the sum of the product of the inputs (x_j) and their respective weights (w_{kj}):

$$net_k\,(\bar{x},\bar{w}) = \sum_{j=1}^{n} x_j w_{kj} \tag{3}$$

This is then passed through a transfer function. The output function is the sigmoid function represented as:

$$y_k = \frac{1}{e^{-net_k}} \tag{4}$$

The partial derivative of the sigmoid function with respect to its argument is derived and represented as follows:

$$\frac{\partial sigmoid\ (z)}{\partial z} = sigmoid\ (z)\big(1 - sigmoid(z)\big) \tag{5}$$

Because the activation function is differentiable, so is the function computed by the neural network, as well as the error function differentiable. The weights are thus adjusted at a rate reflective of the curvature of the sigmoid function. The network is trained to determine a target output from a set of given inputs. The error signal is the difference between the actual activation of the output (y_k) and the target ('desired') activation (t_k) for the neuron. The error is dependent on the weights, thus the weights need to be adjusted in order to minimize the error. This cost function is represented as follows:

$$E_k = \frac{1}{2}\sum_k (y_k - t_k)^2 \tag{6}$$

The error is reduced by adjusting the weights. Thus the partial derivative of the error with respect to the weights is computed. This gradient of *E* in respect of the weights *w* is:

$$\nabla E_k(w_{kj}) = \frac{\partial E}{\partial w_{kj}} \tag{7}$$

The network weights are updated as follows:

$$w_{kj+1} = w_{kj} + \Delta w_{kj} \tag{8}$$

Δw_{kj} (representing the training process/adjustment of each weight) is the vector of all weights of the network. The backpropagation algorithm adjusts the weights using the method of *gradient descent:*

$$\Delta w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}} \tag{9}$$

The learning rate (η) determines how the weights change at each step length of each iteration. The goal is to find the derivative of E with respect tow_{kj}, backwards. Thus referring to Equation 6, the partial derivative of the error with respect to the output can be represented as follows:

$$\frac{\partial E}{\partial y_k} = (y_k - t_k) \tag{10}$$

Next, expanding the partial derivative $\frac{\partial y_k}{\partial w_{kj}}$ by the chain rule, determine how much the output (y_k) depends on the activation (net_k) and how this activation depends on the weights. Referring to Equation 3 and Equation 4:

$$\frac{\partial y_k}{\partial w_{kj}} = \frac{\partial y_k}{\partial net_k}\frac{\partial net_k}{\partial w_{kj}} = y_k(1 - y_k)x_j \tag{11}$$

Thus, combining Equation 10 and Equation 11, the partial derivative of the error with respect to the weights can be represented as follows:

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k}\frac{\partial y_k}{\partial w_{kj}} = (y_k - t_k)y_k(1 - y_k)x_j \tag{12}$$

Rewriting Equation 10, the adjustment to each weight is now derived as follows:

$$\Delta w_{kj} = -\eta(y_k - t_k)y_k(1 - y_k)x_j \tag{13}$$

The Extended Kalman Filter in ANN Training

State estimation is a procedure in which the state of a process needs to be estimated, given the presence of noisy and inaccurate measurements from the process. In ANN modeling, the network weights represent the state that needs to be estimated in the training process. The extended Kalman filter is comprised of two phases: predict next state (before the measurements are taken) and measurement update (update state after the measurements are taken). In state estimation form, the training of the neural network can be formulated as a filtering problem, assuming that the data are generated by the following noisy signal model (Hodnett (2010), adapted from Haykin (2001)):

$$w_k = w_{k-1} + u_k \tag{14}$$

$$t_k = h(w_k, x_k) + v_k \tag{15}$$

Where:

$h(w_k, x_k)$	=	the value of the function is the ANN output;
t_k	=	is the known output target (actual) output vector;
v_k	=	is the measurement noise vector that is assumed to a Gaussian-distributed random variable with zero mean and constant covariance;
x_k	=	is the input vector;
w_k	=	the weights of the ANN model at k; and
u_k	=	is the process noise vector that is assumed to a Gaussian-distributed random variable with zero mean and constant covariance.

The following assumptions are made:

u_k is a white process noise with $E[u_j u_k^T]$ $= \delta_{jk} Q_k$ covariance matrix
v_k is a white measurement noise with $E\left[v_j v_k^T\right]$ $= \delta_{jk} R_k$ covariance matrix
$E[u_j v_k^T] = 0$, for allj, k.

w_k, the network's weight parameter represents the state of the system. Equation (14) represents the process (predict phase) model, which uses estimates from the previous step to produce and estimate of the current state. Equation (15) represents the network's desired response vector t_k as a nonlinear function of the input vector x_k and the weight parameter w_k (Haykin (2001)). Observation of the weights (w_k), is made according to Equation (16), the measurement phase. The nonlinear function h thus relates the state to the measurement t_k**.** Thus, the EKF solution to estimating $\widehat{w}$ that minimizes the network error is obtained via the following recursion (Haykin (2001); Lary and Mussa (2004); Hodnett (2010):

$$\widehat{w}_k = \widehat{w}_{k-1} + K_k(t_k - \hat{y}_k) \tag{16}$$

Where:

$\widehat{w}$	=	weights (the state vector) to be estimated;
$\widehat{w}_{k-1}$	=	previous step weight estimate;
$t_k - \hat{y}_k$	=	vector contains the prediction errors (innovations); and
$\hat{y}_k$	=	is the prediction (=$h(w_{k-1}, x_k)$).

K_k represents a weighting factor known as the Kalman gain matrix at step k. It is the gain or blending factor that minimizes the error covariance, where P_k is defined as the estimate error covariance (conditional mean covariance matrix). P_k represents a measure of the estimated accuracy of the state estimate. The Kalman gain matrix can thus be represented as follows:

$$K_k \quad = \quad P_{k-1} H_k \left(R_k + H_k^T P_{k-1} H_k\right)^{-1} \tag{17}$$

Where:

P_{k-1}	=	the estimate error covariance at the previous iteration; and
R_k	=	the measurement noise covariance at iteration k.

Observing Equation 17, it is evident that as P_{k-1}, approaches zero, the Kalman gain weights the residual error/innovation less heavily, but as R_k approaches zero, the Kalman gain weights the residual error more heavily.

P_k is thus represented as follows:

$$P_k = P_{k-1} - K_k H_k^T P_{k-1} + Q_k \tag{18}$$

Where:

H_k is the matrix of the partial derivatives h_k with respect to all elements of $\hat{w}_{k-1}$, that is:

$$H_k = \frac{[\partial h(\hat{w}_{k-1}, x_k)]}{\partial \hat{w}_{k-1}} \tag{19}$$

$Q_k =$ the process noise covariance, assumed to be zero

An important feature of the extended Kalman filter is that the function of H_k (the Jacobian matrix) is to correctly propagate the relevant component of the measurement component.

DEVELOPING ANN STOCK SELECTION MODELS

Based on the cascade-correlation architecture described in the previous section, two ANN models are built to perform stock selection for global equities, namely, a backpropagation model (BACKPROP) and an extended Kalman filter model (KALMAN). Using the Dow Jones (DJ) Sector Titans Composite as the research database, the ANN models predict forward monthly stock returns and subsequently divide sample stocks into quartiles based on the rankings of their predicted returns.[4] Such fractile analysis enhances the comparison between the predictability of BACKPROP and KALMAN. The performance of the top quartile (Q1) and the bottom quartile (Q4) of the two models are evaluated against the collective performance of sample stocks (that is, the market portfolio) over a 60-month examination period from 01 January 2005 to 31 December 2009. The top quartile (Q1) of a successful stock selection model is expected to generate higher risk-adjusted returns relative to the market portfolio. Conversely, the bottom quartile (Q4) is expected to underperform the market portfolio. We consider a series of financial ratios and firm-specific attributes as candidate model inputs to predict forward stock returns. This includes the beta coefficient that measures the sensitivity of stock returns to market returns, past return momentum (stocks that yield the largest gains over the past 1, 3, 6, 12, 24 and 36 months respectively), yields on fundamental values such as book value, earnings, dividends, sales and cash flow and the year-on-year growth of these fundamental values. The attributes for each sample stock are obtained from DataStream International over the period from 01 January 2004 to 31 December 2009. Although survivorship bias is inherent in the database, research results provide useful insights into investment strategies built on well-established (large market capitalization) global stock portfolios with sufficient sector diversification. The look-ahead bias is avoided by using model inputs that lag model outputs by at least 6 months. All downloaded data are subsequently converted into U.S. dollars. Table 1 demonstrates the detailed descriptions for the computation of these attributes.

[4] The DJ Sector Titans Composite provides a fair sector representation as it consists of the largest 30 global stocks by market capitalization from each of the 19 second tier sectors of the Supersector structure defined by the Industry Classification Benchmark (ICB). This index methodology results in 570 stocks in the research sample. The constituents of this composite are chosen from both developing and developed economies.

Table 1: List of Firm-Specific Attribute

Symbol	Name	Descriptions
Mom36	Prior 36-month total return	Total return index$_t$/Total return index$_{t-36}$ - 1
Mom24	Prior 24-month total return	Total return index$_t$/Total return index$_{t-24}$ - 1
Mom12	Prior 12-month total return	Total return index$_t$/Total return index$_{t-12}$ - 1
Mom6	Prior 6-month total return	Total return index$_t$/Total return index$_{t-6}$ - 1
Mom3	Prior 3-month total return	Total return index$_t$/Total return index$_{t-3}$ - 1
Mom1	Prior 1-month total return	Total return index$_t$/Total return index$_{t-1}$ - 1
YOYC%	Year-on-year cash flow growth	Net cash flow$_t$/Net cash flow$_{t-12}$ - 1
YOYS%	Year-on-year sales growth	Total Sales$_t$/Total Sales$_{t-12}$ - 1
YOYD%	Year-on-year dividend growth	Total dividend$_t$/Total dividend$_{t-12}$ - 1
YOYE%	Year-on-year earnings growth	Earnings after tax$_t$/Earnings after tax$_{t-12}$ - 1
YOYB%	Year-on-year book value growth	Book value$_t$/Book value$_{t-12}$ - 1
CY	Cash flow-to-price ratio	Net Cash flow per share/Share price
SY	Sales-to-price ratio	Total Sales per share/Share price
DY	Dividend yield	Dividend per share/Share price
EY	Earnings yield	Earnings per share after tax/Share price
BY	Book-to-market value ratio	Book value per share/Share price
Beta	Beta coefficient	*Regression coefficient obtained by regressing prior 36-month stock excess returns on prior 36-month market excess returns.

*The market proxy is a monthly-rebalanced equally-weighted portfolio of all constituents from the DJ Sector Titans Composite Index; the risk-free proxy used to calculate excess returns is the 3-month U.S. Treasury bill.

This research first takes on all 17 candidate attributes as potential inputs in the training set. The cross-sectional factor weights are estimated by training the MLP of the ANN models. Four steps are involved in building the respective ANN models, namely: specifying training, test and validation sets; selecting input variables; model construction and training and evaluating model performance:

Step 1 Specifying Training, Test and Validation Sets

The training set maps the model parameters by updating the network weights and hence determines the generalization ability of the learning process. A test set, independent from the training set, is used to reassure the robustness of the weights obtained in the training set in the model building process. The training set and the test set together comprise the in-sample period based on which the respective ANN models are built. A round robin approach is implemented whereby 70% of the in-sample data is used as the training set and 30% as the test set at regular intervals. As opposed to the training set and the test set, the validation set is not involved in the model building process, as it serves to evaluate the performance of the quartiles of the respective ANN models in the out-of-sample period. This research employs a moving-window procedure whereby 12 months are used as the in-sample period to forecast the immediate 1-month forward return in the out-of-sample period.[5] Under this procedure, the data set spanning the period 01 January 2004 to 31 December 2004 represents the first training set to forecast the January 2005 returns. The second training set spans the period 01 February 2004 until 31 January 2005 to forecast the February 2005 returns. This overlapping procedure is repeated a further 58 times until all monthly returns over the out-of-sample period from 01 January 2005 to 31 December 2009 is predicted.

Step 2 Selecting Input Variables

Different variable selection procedures have been employed in research entailing stock return prediction.

[5] Kaastra and Boyd (1996), Qi (1999), Quah and Srinivasan (1999) and Eakins and Stansell (2003) apply this procedure when modelling time-series data. An advantage of the moving-window procedure as opposed to a static procedure is that new information is updated and older information discarded as time proceeds.

For example, studies have employed stepwise and backward stepwise regression (Motiwalla and Wahab, 2000), principal component analysis, or selecting variables based on a set of *a priori* assumptions, to name a few. Some authors argue that although several available methods are applicable for linear modeling, these approaches cannot be simply extended to nonlinear modeling. For example, Mao and Billings (1999:352) attribute the inapplicability of forward selection and backward elimination algorithms to the weight structure of an MLP neural network and explain "*these algorithms are in fact term selection and/or deletion methods rather than variable selection and/or deletion algorithms in the nonlinear system case"*. A genetic algorithm (Holland (1975) and Goldberg (1989)) is used in this research to select the optimal subset of input variables for the neural network.[6] This evolutionary search procedure has the capability of modeling complex systems with multiple solutions. Inspired by Charles' Darwin's principle of evolution and genetics, the laws of natural selection apply where only the fittest members of a group will survive. These fit members genetically recombine with other fit members to produce offspring thereby ensuring that good characteristics pass on to the next generation, while those less fit members will be eliminated. The standard GA starts with the simulation of a random population consisting of subsets of input variables referred to as "individuals" or "chromosomes" in the evolutionary process. Each member of the population is then evaluated and graded according to a fitness function. This ensures that only the proportion of the fittest individuals from the current ("parent") population is extracted. Through the process of reproduction (crossover and mutation) individuals (subset of input variables) are selected for the next generation. This continues until a particular stopping criterion (50 generations for BACKPROP and 30 generations for KALMAN) is achieved. With regard to KALMAN, a necessary requirement for the extended Kalman filter rule is that the number of input variables presented to the network for training is reduced.

Step 3 Construction and Training of ANN Model

Referring to Figure 2, the cascade-correlation algorithm is depicted as follows:

1. Cascade-correlation starts with a network consisting of a fully connected input layer and output layer with no hidden layers.
2. Train the output-side weights via the backpropagation learning rule for BACKPROP and via the extended Kalman filter learning rule for KALMAN until the mean squared error (MSE) is minimized.
3. Initialize candidate hidden neuron. Candidate neurons are fully connected to input neurons as well as any hidden neuron. There are no connection weights between the candidate neurons and output neurons at this stage.
4. Attempt to maximize the correlation between the candidate hidden neuron's output and the network output. Learning stops when there is no further improvement in correlation.
5. The candidate neuron with the maximum correlation between its output and the network output is prepared for installation in the network.
6. Selected hidden neuron's input-side weights are then frozen; it is installed in network with links between the hidden neuron and the output neuron created (output-side weights).
7. Go back to 2.

This process is repeated until the error is sufficiently reduced when the specified hidden layers reaches a maximum. In the research a maximum of 30 hidden layers are specified, with a minimum increment of 1 and a maximum increment of 2 hidden neurons added at a time.

[6] Genetic algorithms have been previously employed in financial applications, for example, by Mahfoud and Mani (1995) for stock selection, Rutan (1993) for portfolio selection and Kingdom and Feldman (1995) for bankruptcy prediction, to name a few.

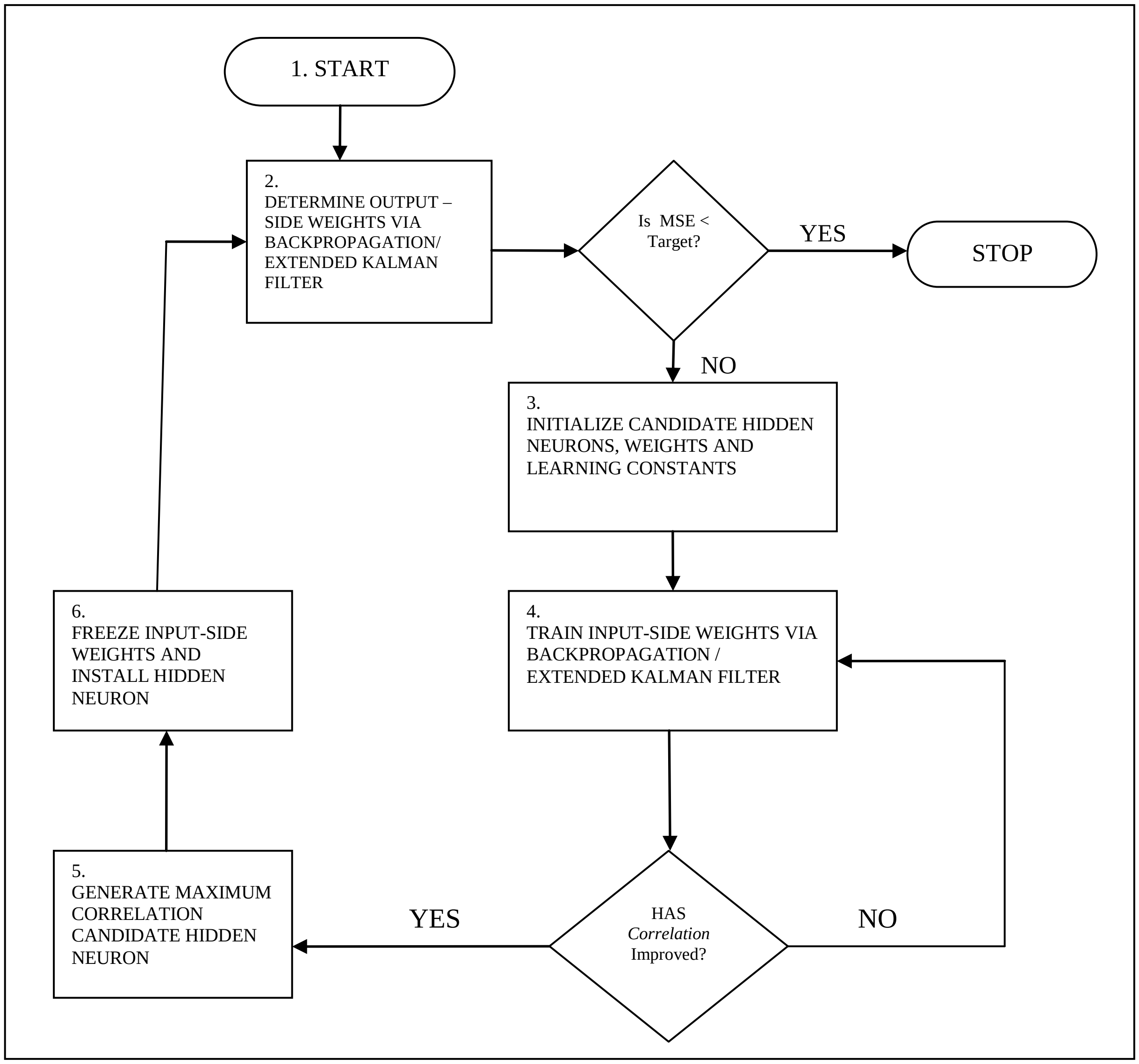
1. START
2. DETERMINE OUTPUT – SIDE WEIGHTS VIA BACKPROPAGATION/ EXTENDED KALMAN FILTER
Is MSE < Target?
YES
STOP
NO
3. INITIALIZE CANDIDATE HIDDEN NEURONS, WEIGHTS AND LEARNING CONSTANTS
4. TRAIN INPUT-SIDE WEIGHTS VIA BACKPROPAGATION / EXTENDED KALMAN FILTER
HAS Correlation Improved?
YES
NO
5. GENERATE MAXIMUM CORRELATION CANDIDATE HIDDEN NEURON
6. FREEZE INPUT-SIDE WEIGHTS AND INSTALL HIDDEN NEURON

Figure 2: Cascade-Correlation Algorithm

Step 4 Evaluation of ANN Models

The overall accuracy of model predictions is evaluated by calculating the time-series correlation between the model's forecasted returns and the actual returns over the training, testing and the out-of-sample periods. Fractile analysis is conducted to evaluate the practicality of the ANN models by assessing the relative out-of-sample quartile performance based on risk-adjusted performance measures such as the Sharpe ratio, the Treynor measure and Jensen's alpha. The Sharpe ratio discounts the excess portfolio return by the standard deviation (a measure of total volatility) of portfolio returns to measure excess return per unit of volatility as shown in Equation 19:

$$Sharpe\ Ratio = \frac{R_{Xp.a.} - U.S.TB3M_{p.a}}{\sigma_{Xp.a}} \tag{19}$$

Where:

$U.S.TB3M_{p.a}$	=	the yield on U.S. 3-month Treasury Bill;

$R_{Xp.a}$	=	the annualized return for portfolio X,
$\sigma_{Xp.a}$	=	the annualized standard deviation of return for portfolio X

The Treynor measure estimates excess return per unit of systematic risk proxied by the fractile's beta coefficient as shown in Equation 20:

$$Treynor\ Measure = \frac{R_{Xp.a.} - USTB3M_{p.a}}{\beta_{X,m}} \tag{20}$$

Using equally-weighted portfolio of all constituents from the DJ Sector Titans Composite Index and the 3-month U.S. Treasury yield as the risk-free rate, the beta coefficient of portfolio *X* is estimated using Equation 21:

$$R_{Xp.a} - USTB3M_{p.a} = \propto_X - \beta_X\,(R_{m,p.a} - USTB3M_{p.a}) \tag{21}$$

Where:

β_X	=	the beta coefficient for portfolio X,
$R_{m,p.a}$	=	the annualized return for the market proxy

The regression intercept, $\propto_X$ is known as Jensen's alpha and measures the abnormal return of portfolio *X*.

RESULTS

Variable Selection

The attributes selected as inputs for BACKPROP and KALMAN using a genetic algorithm to forecast monthly stock returns in the out-of-sample period are displayed in Figure 3 and Figure 4 respectively. With regard to BACKPROP (refer to Figure 3), most of the momentum attributes are important inputs for forecasting returns, with the exception of Mom3. DY is the most important attribute in the value fundamental category for BACKPROP. Beta, a measure of sensitivity of stock returns to market risk, is also an important model input employed by BACKPROP. Attributes in the growth category are less frequently selected, with the exception of YOYB%.

For both BACKPROP and KALMAN, Mom12 is the most frequently selected model input, among other attributes. Mom6 and Mom24 are selected as inputs for KALMAN, mostly prior to 2007 (refer to Figure 4). On the other hand, YOYB% is mostly selected for monthly forecasting in 2007 by KALMAN. The presence of other attributes from the growth category for KALMAN is not obvious. With the exception of DY, attributes from the value category are not frequently selected for KALMAN. As opposed to BACKPROP, beta is not regarded as an important input for KALMAN.

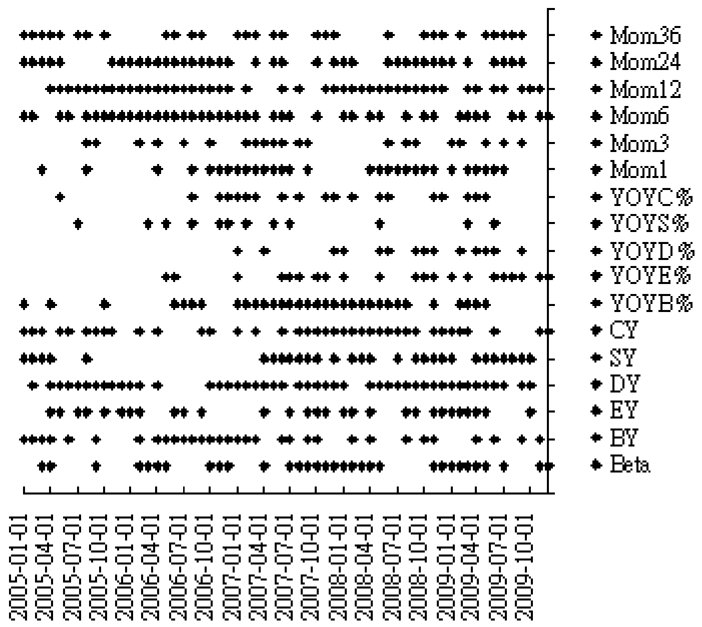

Figure 3: This figure displays the attributes selected for BACKPROP using a genetic algorithm over the examination period form 01 January 2005 to 31 December 2009. The variables examined include the beta coefficient (Beta), yields (Y) on fundamental values such as book value (B), earnings (E), dividends (D), sales (S) and cash flow (C), the year-on-year (YOY) growth of the above fundamental values, prior return momentum (Mom) and the beta coefficient (Beta).

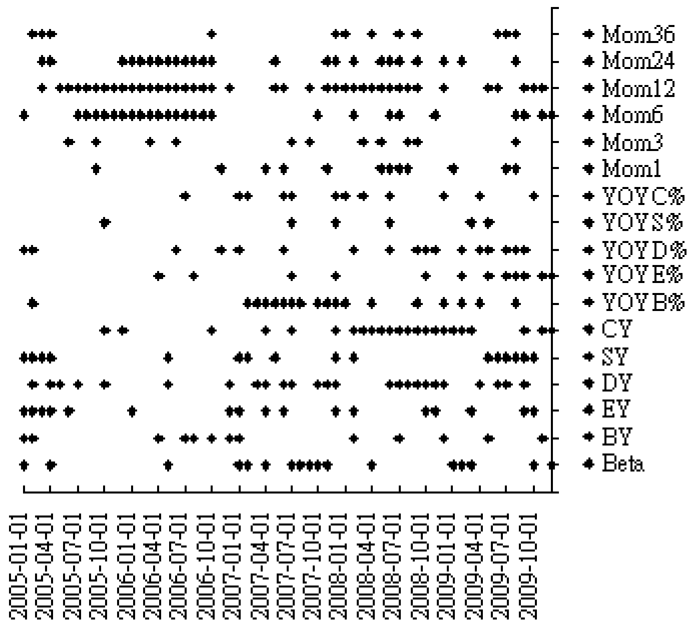

Figure 4: This figure displays the attributes selected for KALMAN using a genetic algorithm over the examination period form 01 January 2005 to 31 December 2009. The variables examined include the beta coefficient (Beta), yields (Y) on fundamental values such as book value (B), earnings (E), dividends (D), sales (S) and cash flow (C), the year-on-year (YOY) growth of the above fundamental values, prior return momentum (Mom) and the beta coefficient (Beta).

Correlation Analysis

The log cumulative monthly correlations between the forecasted returns and the realized returns for the out-of-sample period and their corresponding in-sample period (train and test) results for BACKPROP and KALMAN are illustrated in Figure 5 and Figure 6 respectively. The monthly out-of-sample non-cumulative correlations between the forecasted returns and the realized returns are shown in histograms. A kink in the out-of-sample log cumulative correlation (refer to the trend line marked by cross legends) around 01 January 2006 is observed for both BACKPROP (refer to Figure 5) and KALMAN (refer to Figure 6). The gap between the in-sample (train and test periods) and out-of-sample log cumulative correlations widens since the beginning of 2006 and is apparent for both BACKPROP and KALMAN. The out-of-sample log cumulative correlation for BACKPROP completely flattens out after 2006. By contrast, the log cumulative out-of-sample correlation for KALMAN, on the other hand, continues to increase after 2006.

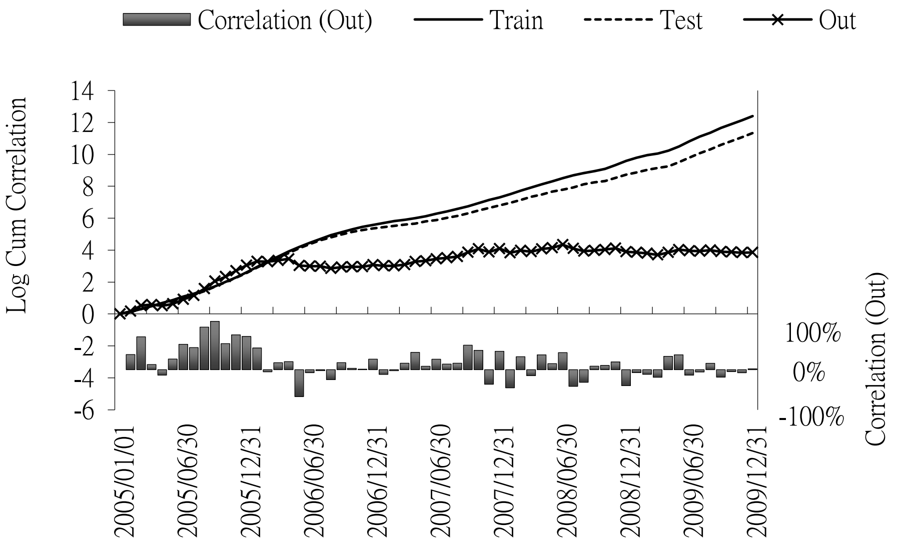

Figure 5: This figure displays the log cumulative monthly correlations between the forecasted returns and the realised returns for the out-of-sample period and the corresponding in-sample period (train and test) for BACKPROP. The monthly out-of-sample correlations (non-cumulative) are illustrated in the form of histograms.

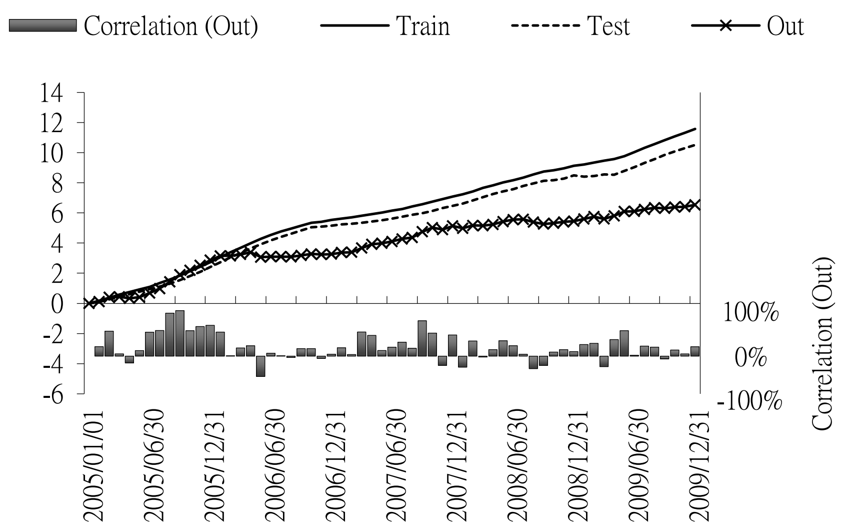

Figure 6: This figure displays the log cumulative monthly correlations between the forecasted returns and the actual returns for the out-of-sample period and the corresponding in-sample (train and test) period for KALMAN. The monthly out-of-sample correlations (non-cumulative) are illustrated in the form of histograms.

Fractile Analysis

Figure 7 illustrates the log cumulative returns for the top quartile (Q1) and bottom quartile (Q4) for BACKPROP and KALMAN relative to that of the market proxy over the out-of-sample period.

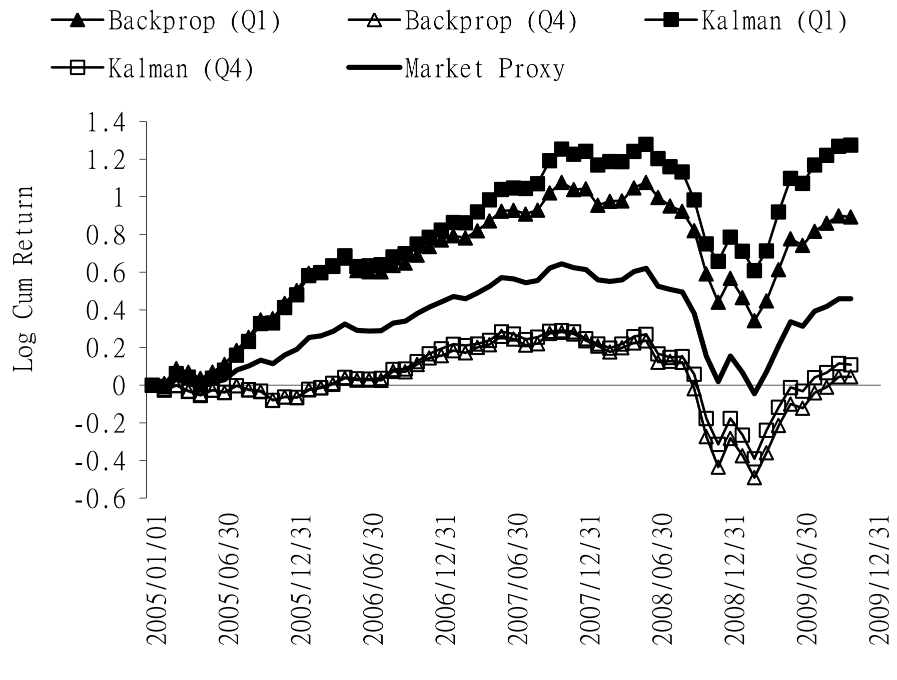

Figure 7: This figure displays the **log cumulative returns** for the top quartile (Q1) and the bottom quartile (Q4) of BACKPROP and KALMAN respectively over the out-of-sample period from 01 January 2005 to 31 December 2009. The cumulative return for the market proxy is represented by the solid trend line in the figure.

The top quartiles of both BACKPROP and KALMAN outperform the market proxy. On the other hand, the bottom quartiles of both models underperform the market proxy. The performance of the top quartiles constructed under the respective models only diverge after 2006 in that Q1 of KALMAN continues to accumulate returns faster than Q1 of BACKPROP. The performance of the bottom quartiles for both models were however not distinctively different.

Figure 8 illustrates the log cumulative returns for long-short zero investment portfolios constructed using the top and bottom quartiles of the ANN models and the market proxy over the out-of-sample period. The zero investment portfolios constructed using the top quartile and the bottom quartile (Q1 - Q4) of both BACKPROP and KALMAN outperform their own zero investment portfolios constructed using the top quartile and the market proxy (Q1 - MKT), which in turn outperforms the zero investment portfolios constructed using the bottom quartile and the market proxy (Q4 – MKT). Although KALMAN (Q1 – Q4) outperform BACKPROP (Q1 – Q4) and KALMAN (Q1 – MKT) outperform BACKPROP (Q1 – MKT), there is no clear distinction in the log cumulative performance between BACKPROP (Q4 – MKT) and KALMAN (Q4 – MKT).

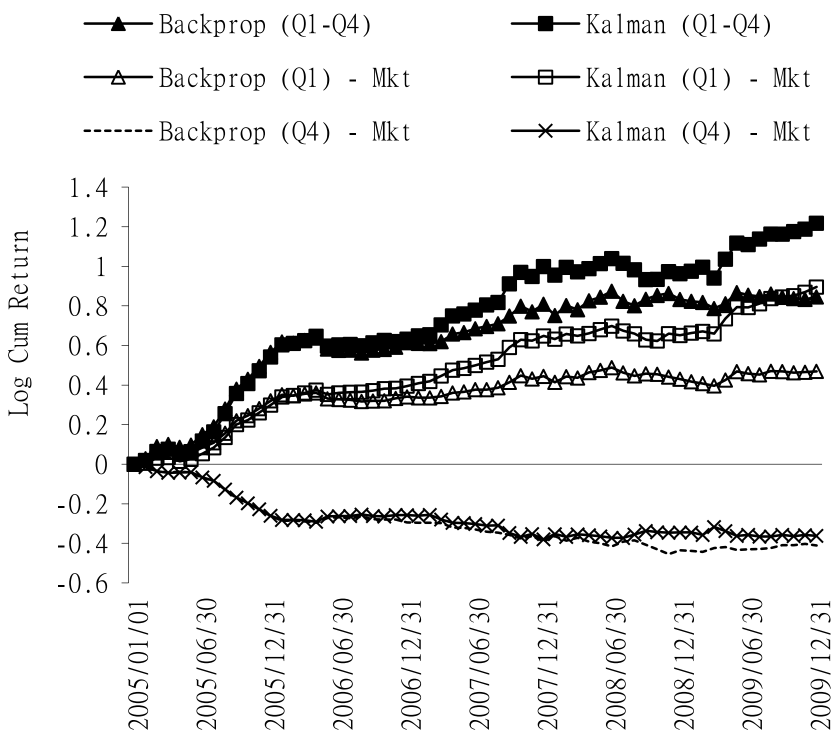

Figure 8: This figure displays the **log cumulative spreads** for the long-short zero investment portfolios constructed by longing (buying) the top quartile (Q1) and shorting (selling) the bottom quartile (Q4) of the respective BACKPROP and KALMAN models and the market proxy over the out-of-sample period from 01 January 2005 to 31 December 2009.

The out-of-sample risk-return performance statistics for the top quartile and bottom quartile of BACKPROP and KALMAN relative to the market proxy are demonstrated in Table 2. KALMAN (Q1) achieves the highest annualized return while BACKPROP (Q4) achieves the lowest annualized return over the examination period. The top quartiles of the ANN models are riskier than their bottom quartile counterparts in terms of standard deviation and beta coefficient. With regard to the risk-adjusted performance measures, KALMAN (Q1) achieves the highest Sharpe ratio, Treynor measure and Jensen's alpha while BACKPROP (Q4) achieves the lowest Sharpe ratio, Treynor measure and Jensen's alpha, over the examination period. Examining the *p*-value of Jensen's alpha reveals that all four long-only portfolios constructed under both ANN models have generated statistically significant risk-adjusted abnormal returns at a 5% significance level. This serves as strong evidence as to the strength of both models in classifying future winners and losers.

Table 2: Performance Evaluation of the Long-Only Portfolios

	BACKPROP (Q1)	KALMAN (Q1)	BACKPROP (Q4)	KALMAN (Q4)	Market Proxy
Return	19.56%	29.04%	0.90%	2.19%	9.60%
Stdev	24.80%	25.65%	22.80%	21.21%	21.49%
Beta	1.11	1.15	1.03	0.95	1.00
Sharpe	0.675	1.022	-0.084	-0.030	0.315
Treynor	0.150	0.229	-0.019	-0.007	0.068
Jensen	0.007	0.014	-0.007	-0.006	0.000
t-Stats	2.940	5.040	-3.374	-2.647	-------
p-Value	0.005	0.000	0.001	0.011	-------

The out-of-sample risk-return performance statistics of the zero-investment portfolios constructed under the respective ANN models are shown in Table 3. KALMAN (Q1 – Q4) and BACKPROP (Q4 – MKT) achieves the highest and the lowest returns respectively over the examination period. All zero-investment portfolios achieve beta coefficients close to zero. KALMAN (Q1 – Q4) achieves the highest risk-adjusted performance while BACKPROP (Q4 – MKT) has the weakest risk-adjusted performance in terms of the Sharpe ratio, Treynor measure and Jensen's alpha. All six zero-investment portfolios earn statistically significant Jensen's alpha over the examination period.

Table 3: Performance Evaluation of the Zero-Investment Portfolios

	BACKPROP (Q1-Q4)	KALMAN (Q1-Q4)	BACKPROP (Q1-MKT)	KALMAN (Q1-MKT)	BACKPROP (Q4-MKT)	KALMAN (Q4-MKT)
Return	18.24%	26.52%	9.70%	18.57%	-7.85%	-7.01%
Stdev	11.81%	12.93%	6.94%	7.84%	5.26%	5.74%
Beta	0.08	0.19	0.11	0.14	0.03	-0.05
Sharpe	1.305	1.833	0.991	2.010	-1.932	-1.714
Treynor	1.902	1.235	0.614	1.092	-3.639	1.998
Jensen	0.012	0.017	0.005	0.012	-0.010	-0.008
t-Stats	2.703	3.679	1.994	4.174	-4.414	-3.686
p-Value	0.009	0.001	0.049	0.000	0.000	0.001

CONCLUSION

With regard to variable selection, BACKPROP requires far more inputs to forecast stock returns in any given month compared to KALMAN. As was mentioned earlier in the paper, an extended Kalman filter model requires fewer inputs for network training. Attributes from the momentum category (especially Mom12) are the most frequently selected attributes by the genetic algorithm for both BACKPROP and KALMAN models. On the other hand, DY is the most recognized input in the value category for both ANN models. Although Beta and the majority of the value attributes are employed by BACKPROP, they are not regarded as important inputs for stock return forecasting for KALMAN. With the exception of YOYB%, the attributes from the growth category are not frequently included as inputs for both ANN models.

Positive correlations between the realised returns and model forecasted returns indicate the strength of the model in distinguishing outperforming and underperforming stocks in the forecasted periods. On the other hand, weak positive or even negative correlations reveal the model's weakness in ranking stocks according to their respective performance in the forecasted periods. Comparing the log cumulative correlations of the in-sample and out-of-sample periods for the ANN models reveal a structural decrease in the robustness of both ANN models in forecasting stock returns after 2006. This decrease was however less significant for KALMAN as the correlation between the realised returns and the forecasted returns after 2006 remains mostly positive. By contrast, the log cumulative correlation of BACKPROP flattens after 2006 until the end of the examination period.

The quartile analysis and risk-adjusted performance statistics provide further evidence that KALMAN exhibits greater strength in identifying future top performers (using the top quartile as the proxy) in the global equity market compared to BACKPROP. This superiority of KALMAN over BACKPROP is consistent with tests results of Singhal and Wu (1989) and Nechyba and Xu (1997). There is no distinguishable difference between the performances of the bottom quartiles formed by both ANN models. The zero-investment portfolios formed by longing the top quartiles and simultaneously shorting the bottom quartiles or the market proxy exhibit statistically significant Jensen's alpha and continues to accumulate positive returns over the out-of-sample period for both ANN models. On the other hand, the zero-investment portfolios formed by longing the bottom quartiles and simultaneously shorting the market proxy exhibit statistically significant Jensen's alpha and continues to accumulate losses over the out-of-sample period for both ANN models.

Our findings support the uses of artificial neural network for stock return forecasting as an active portfolio management tool. Further to this, our results motivate for the use of the extended Kalman filter in training artificial neural networks for applications involving noisy financial data. An area requiring further research includes using the extended Kalman filter to train fixed-architecture neural networks for financial forecasting.

ACKNOWLEDGEMENTS

This work is supported by the National Research Foundation of South Africa. The authors would like to thank the School of Business and Finance of the University of the Western Cape where the research was undertaken, as well as Professor Paul van Rensburg of the University of Cape Town.

AUTHOR INFORMATION

Dr. Kathleen Hodnett is currently a Research Fellow (funded by the National Research Foundation (NRF) of South Africa) in the School of Business and Finance at the University of the Western Cape, South Africa. She is a member of the International Institute of Forecasters (IIF) and an associate member of the South African Institute of Financial Markets (SAIFM).

Dr. Heng-Hsing Hsieh, CFA is the Head of Finance in the School of Business and Finance at the University of the Western Cape, South Africa. He is a CFA charterholder and a member of the South African Institute of Financial Markets (SAIFM). E-mail: ahsieh@uwc.ac.za. Corresponding Author.

REFERENCES

1. Abhyankar A, Copeland L and W Wong (1997), "Uncovering Nonlinear Structure in Real-Time Stock-Market Indexes: The S&P 500, the DAX, the Nikkei 225, and the FTSE-100", *Journal of Business and Economic Statistics*, American Statistical Association, vol. 15, no 1, 1-14.
2. Attew D (2002), "Artificial Neural Networks", In *Neural Networks and the Financial Markets*, Shadbolt J and J G Taylor (eds.), Springer, 87-93.
3. Banz R W (1981), "The Relationship between Return and Market Value of Common Stocks", *Journal of Financial Economics*, vol. 9, 3-18.
4. Cao Q, Leggio K and M Schiederjans (2005), "A Comparison between Fama and French's Model and Artificial Neural Networks in Predicting the Chinese Stock Market", *Computers and Operations Research*, vol. 32, 2499-2512.
5. Cao Q, Parry M and B Leggio (2009), "The Three-Factor Model and Artificial Neural Networks: Predicting Stock Price Movement in China", *Annals of Operations Research*, Springer.
6. Diamantopoulou M J (2006), "Tree-Bole Estimation on Standing Pine Trees Using Cascade-Correlation Artificial Neural Network Models", *Agricultural Engineering International: The GIGR Ejournal. Manuscript IT 06 002*, vol. 8, 1-14.

7. Eakins S and R Stansell (2003), "Can Value-Based Selection Criteria Yield Superior Risk-Adjusted Returns: An Application of Neural Networks", *International Review of Financial Analysis*, vol. 12, 83-97.
8. Eakins S, Stansell R and J Buck (1998), "Analyzing the Nature of Institutional Demand for Common Stocks", *Quarterly Journal of Business and Economics*, vol. 37.
9. Fahlman S E and C Liebere (1990/1991), "The Cascade-Correlation Learning Algorithm", *Technical Report, CMU-CS-90-100*, Carnegie Mellon University, 1-13.
10. Fama E F and K R French (1992), "The Cross-Section of Expected Stock Returns", *Journal of Finance*, vol. 47, 427-465.
11. Fama E F and K R French (1995), "Size and Book-to-Market Factors in Earnings and Returns", *Journal of Finance*, vol. 50, no 1, 131-156.
12. Goldberg D E (1989), "*Genetic Algorithms in Search, Optimisation and Machine Learning*", Addison-Wesley.
13. Gonzalez S (2000), "Neural Networks for Macroeconomic Forecasting: A Complementary Approach to Linear Regression Models", Finance Canada Working Paper.
14. Haykin S (2001): *Kalman Filtering and Neural Networks*, Wiley.
15. Hodnett K (2010), "Analysis of the Cross-Section of Equity Returns on the JSE Securities Exchange based on Linear and Nonlinear Modelling Techniques", *Unpublished Doctoral Thesis*, University of Cape Town
16. Holland J H (1975), "*Adaptation in Natural and Artificial Systems*", University of Michigan Press, Ann Arbor.
17. Hsieh H (2010), "Applications of Global Equity Style Indices in Active and Passive Portfolio Management", *Unpublished Doctoral Thesis*, University of Cape Town
18. Hung S, Liang T and V Liu (1996), "Integrating Arbitrage Pricing Theory and Artificial Neural Networks to Support Portfolio Management", *Decision Support Systems*, vol. 18, 301-316.
19. Kaastra I and M Boyd (1996), "Designing a Neural Network for Forecasting Financial and Economic Time Series", *Neurocomputing*, vol. 10, 215-236.
20. Kalman R (1960), "A New Approach to Linear Filtering and Prediction Problems", *Transactions of the ASME- Journal of Basic Engineering*, 35-45.
21. Kanas A and A Yannopoulos (2001), "Comparing Linear and Nonlinear Forecasts for Stock Returns", *International Review of Economics and Finance*, vol. 10, 383-398.
22. Kingdom J and K Feldman (1995), "Genetic Algorithms for Bankruptcy Prediction", *Search Space Report, No. 09-95*, Search Space Limited, London
23. Lary D J and H Y Mussa (2004), "Using An Extended Kalman Filter Learning Algorithm for Feed-Forward Neural Networks to Describe Tracer Correlations" *Atmospheric Chemistry and Physics*, vol. 4, no 3, 3653-3667
24. Lintner J (1965), "The Valuation of Risky Assets and the Selection of Risky Investments in Stock Portfolios and Capital Budgets", *Review of Economics and Statistics*, vol. 47, no 1, 13-37
25. Mahfoud S and G Mani (1995), "Genetic Algorithms for Predicting Individual Stock Performance", Proceedings of the 3rd International Conference on Artificial Intelligence Applications on Wall Street, 171-181
26. Mao K Z and S A Billings (1999), "Variable Selection in Non-Linear Systems Modeling", *Mechanical Systems and Signal Processing*, vol. 13, no 2, 351-366
27. McNelis P (2005), *Neural Networks in Finance: Gaining Predictive Edge in the Market*, Elsevier
28. Motiwalla L F and M Wahab (2000), "Predictable Variation and Profitable Trading of US Equities: A Trading Simulation Using Neural Networks" *Computers and Operations Research*, vol. 27, 1111-1129
29. Nechyba M and Y Xu (1997), "Cascade Neural Networks with Node-Decoupled Extended Kalman Filter", Proceedings on IEEE Intelligence Symposium on Computational Intelligence in Robotics and Automation, vol. 1, 214-219
30. Qi M (1999), "Nonlinear Predictability of Stock Returns Using Financial and Economic Variables", *Journal of Business and Economic Statistics*, vol. 17, no 4, 419-429
31. Qi M and G Maddala (1998), "Economic Factors and the Stock Market: A New Perspective", *Journal of Forecasting*, vol. 18, 151-166
32. Quah T and B Srinivasan (1999), "Improving Returns on Stock Investments through Neural Network Selection", *Expert Systems with Applications*, vol. 17, 295-301

33. Ross S A (1976), "The Arbitrage Theory of Capital Asset Pricing", *Journal of Economic Theory*, vol. 13, no 2, 341-360
34. Ruck W, Rogers S, Kabrisky M, Maybeck P and M Oxley (1992), "Comparative Analysis of Backpropagation and the Extended Kalman Filter for Training Multilayer Perceptrons", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no 6, 686-691
35. Rutan E (1993), "Experiments with Optimal Stock Screens", Proceedings of the 3rd International Conference on Artificial Intelligence Applications on Wall Street, 269-273
36. Sharpe W F (1964), "Capital Asset Prices: A Theory of Market Equilibrium under Conditions of Risk", *Journal of Finance,* vol. 19, no 3, 425-442
37. Singhal S and L Wu (1989), "Training Multilayer Perceptrons with the Extended Kalman Algorithm", *Advances in Neural Information Processing Systems*, Touretzky D (ed.), Morgan Kaufmann Publishers, 133-140
38. Singleton J C and A J Surkan (1990), "Bond Rating with Neural Networks", In *Neural Networks in the Capital Markets*, Refenes, (ed.), John Wiley & Sons, 301-307
39. Sofge D A (2002), "Using Genetic Algorithm Based Variable Selection to Improve Neural Network Models for Real-World Systems", Proceedings of the 2002 International Conference on Machine Learning and Applications
40. Swanson N R and H White (1995b), "A Model-Selection Approach to Real-Time Macroeconomic Forecasting Using Linear Models and Artificial Neural Networks", Working Paper, Department of Economics, Penn State University
41. Utans J and J Moody (1991), "Selecting Neural Network Architectures via the Prediction Risk: Application to Corporate Bond Rating Prediction", Proceedings of the 1st International Conference on Artificial Intelligence Applications on Wall Street, IEEE Computer Society Press, CA
42. Van Rensburg P and M Robertson (2003b), "Size, Price-to-Earnings and Beta on the JSE", *Investment Analysts Journal*, vol. 58, 1-11

NOTES