For "AI in economics and business administration"

Neural Networks And Genetic Algorithm For Economic Forecasting

Francis Wong, PanYong Tan
Institute of Systems Science
National University of Singapore
Fax: (65)7782571

Abstract

This paper describes the applications of an enhanced neural network and genetic algorithm to economic forecasting. Our proposed approach has several significant advantages over conventional forecasting methods such as regression and the Box-Jenkins methods. Apart from being simple and fast in learning, a major advantage is that no assumptions need to be made about the underlying function or model, since the neural network is able to extract hidden information from the historical data. In addition, the enhanced neural network offers selective activation and training of neurons based on the instantaneous causal relationship between the current set of input training data and the output target. This causal relationship is represented by the Accumulated Input Error (AIE) indices, which are computed based on the accumulated errors back-propagated to the input layers during training. The AIE indices are used in the selection of neurons for activation and training. Training time can be reduced significantly, especially for large networks designed to capture temporal information. Although neural networks represent a promising alternative for forecasting, the problem of network design remains a bottleneck that could impair widespread applications in practice. The genetic algorithm is used to evolve optimal neural network architectures automatically, thus eliminating the many pitfalls associated with human-engineering approaches. The proposed concepts and design paradigm were tested on several real applications* , including the forecast of GDP, air passenger arrival and currency exchange rates.

1. Introduction

A forecast is an event that is going to happen in the future based on its present 'surrounding events'. Basically, forecasting assumes that future occurrences are based, at least in part, on presently observable or past events. Moreover, it assumes that some aspects of the past patterns will continue into the future (known as the assumption of continuity). Past relationships can then be discovered through study and observation. Forecasting techniques fall into two major categories: qualitative and quantitative methods. *Qualitative methods* may be subdivided into exploratory and normative methods while *quantitative methods* may be subdivided into time series and casual methods. *Exploratory methods* utilize the past as well as the present as their starting point and move toward the future in a heuristic manner, looking into all available possibilities. *Normative methods* begin by determining future goals and objectives, then working backwards to see if these can be achieved, given the constraints, resources and technologies available presently. *Casual models*, on the other hand, assume that the target to be forecasted exhibits a cause-effect relationship with one or more independent variables. The purpose is to discover the form of that relationship and use it to forecast future values of the dependent variable(s). *Time series* prediction is the forecast of the future based on past values of a variable and/or past errors with the objective

* The software used is available, please email thgoh@iss.nus.sg for a copy.

to discover the pattern in the historical data series and to extrapolate that pattern into the future.

In general, a time series can be broken down into 4 components: secular trend, cyclical variation, seasonal fluctuation and irregular fluctuation [Hanke 89]. Secular trends are long-term, slow moving, low frequency components. Cyclical variations usually take a few years to complete a cycle, while seasonal variations within a year. Irregular fluctuations are random in nature and usually difficult to predict. The irregular, random components can be further divided into two subcomponents: the deterministic chaotic behavior and the stochastic noise.

Classical time series analysis, based on the theory of stationary stochastic processes [Harvey 89], does not perform satisfactorily on economic time series. Economic data are not simple autoregressive-integrated-moving average (ARIMA) processes; they are not describable by simple linear structural models; they are not simple white noise or even random walks. Hence the major challenge ahead is the development of new methods, or the modification or integration of existing ones, that are capable of accurately forecasting series whose patterns or relationships change over time. Accurate stock market prediction and electricity usage forecasting are some examples of such problems.

Neural networks are currently very actively studied and applied to a wide range of applications, in addition to traditional areas such as pattern recognition and control. Its non-linear learning and interpolative smoothening capabilities have proven superior to conventional methods in solving certain categories of problems, one of which is time series forecasting. [Lapedes 87] not only demonstrated that neural networks are capable of decoding deterministic chaos, they also found that neural networks gave much superior predictions of certain types of non-linear, dynamical systems compared to several conventional methods based on adaptive filtering and polynomial curve-fitting; A 3-dimensional back-propagation model [Wong 90] was designed to simultaneously process multiple records of time series data, and has been effectively demonstrated for time series forecasting. Several comparative studies [Tang 91, Sharda 90] have also found that the neural network approach gave much superior predictions of certain non-linear, dynamical systems compared to several conventional methods. Specifically, for time series with long memory, neural network and the Box-Jenkins model produce comparable results. But for series with short memory, neural networks perform much better [Tang 91]. In another study [Sharda 90], it was found that simple neural network models could forecast as well as the much more complex Box-Jenkins model . Other studies have also concluded that the neural networks perform as well as or better than the Box-Jenkins or conventional methods [Varfis, Atlas, Windsor 90].

This paper gives a description of the enhanced neural network for time series forecasting, followed by an overview of using the genetic algorithm to further improve forecast accuracy. Three real forecasting applications: GDP, air passenger arrival and currency exchange rates, are discussed.

2. The FastProp Algorithm For Time Series Forecasting

Back propagation is a technique for solving the credit assignment problem posed by Minsky and Papert in Perceptrons. David Rumelhart [Rumelhart 86] is usually the person associated with the invention of the back-propagation model. David Parker introduced a similar algorithm around the same time [Parker 85] and others have studied similar kinds of networks. The main mechanism in a back-propagation network is to forward the input data through the hidden layers to the output layers, determine the error between the actual output and the desired output (target), and then back propagate the error from the output to

the input through the hidden layers. During the learning phase, the objective is to minimize the global error by modifying the weights associated with the links. The generally good performance found for the back-propagation algorithm is somewhat surprising considering that it is a gradient search technique that may find a local minimum in the error term instead of the desired global minimum.

The main problem noted with the back-propagation algorithm is the large amount of training time required for convergence. Although a number of training algorithms have been proposed to speed up the convergence, it seems unlikely that a significant reduction in the number of training can be achieved. Moreover, with too few hidden nodes, the network may not be trained at all. If just barely enough, the network may learn but it might not be robust when faced with noisy data, and may not recognize patterns that it has not seen before. Furthermore, in order to capture the temporal information contained in time series, one may have to use a 3-D structure similar to that of figure 1 for better forecasting accuracy [Wong 90]. The number of layers and neurons used in such designs are much larger than other back-propagation models and will require a much longer training time.

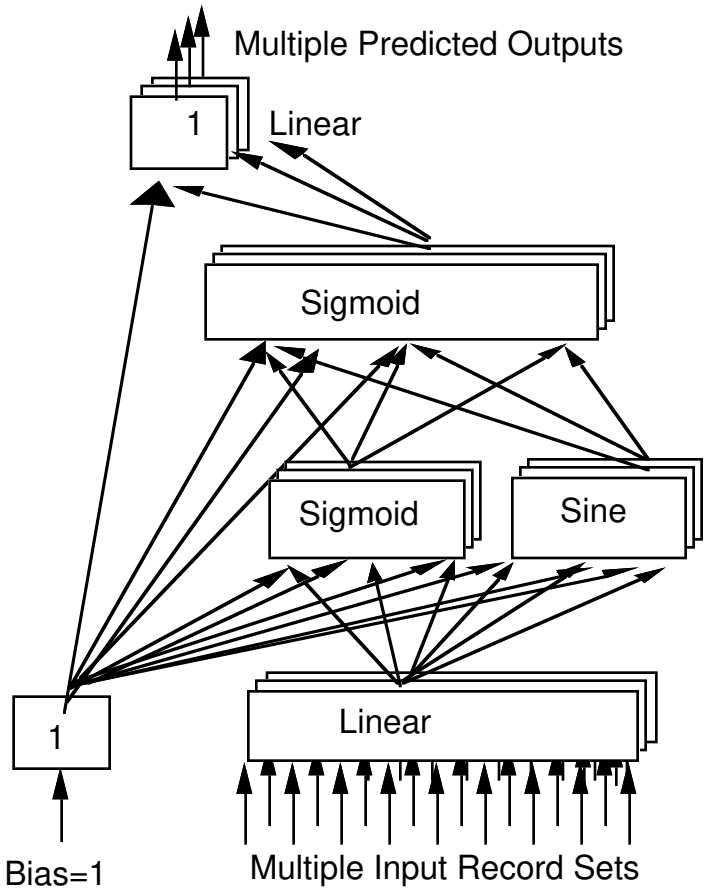


Fig. 1 -- The architecture of the 3-D NeuroForecaster [Wong 90].

One way to reduce the training time is to activate and train the neurons selectively according to some criteria [Wong 91]. One such criteria is the current "situation" of the set of input time series data. For example, suppose that the current economic situation is very good, then only certain groups of neurons in the hidden layers (which are associated with the concept of "very good economic situation") need to be involved in the forward computation of the signals, as well as the back-propagation of error. One may use a clustering method

to cluster the time series into several situations prior to the training. The algorithm we are going to present perform the clustering of situations dynamically during the training process.

One could explain why the neural network associates a specific output signal with the input factors, by transforming the weights stored in it into an explicit form using the Causal Index method [Baba 90]. But the causal indices can only be calculated at the end of the training process when the system error has converged to an acceptable level, whereas the clustering of input data has to be performed during training. The FastProp algorithm [Wong 91] uses the Accumulated Input Error (AIE) index to obtain instantaneous causal relationship between the input and the output, which is useful for finding the cluster which the current set of input data belongs. The AIE index of the i-th input at time t is calculated as follows:

$$AIE_{ik}\ (t=T) = \sum_{t=0 \text{ to } T} | e_{it}[s=0] |$$

where i indexes over the neurons in the input layer (s=0). The total influence of X_i on Y_k at time t can be expressed by $AIE_{ik(t)}$ which is the causal relationship between the i-th input and the k-th output signals, and is calculated based on the absolute errors back-propagated to the input layers for all the input patterns presented, over the entire training cycles (t=0 to t=T).

The AIE indices are then used to rank the input signals according to their magnitudes. The input signal which has the highest AIE index value is referred to as the "reference input" and is represented by using the index "i*". The entire set of time series data can thus be clustered into several "situations" based on the magnitude of the reference input signal with respect to its value range $Max(x_{i*}[0])$-$Min(x_{i*}[0])$. The calculation of $x_j[s]$, $e_j[s]$, $\Delta w_{ji}[s]$ for the s-th layer are restricted to those neurons falling within the following range:

$$\frac{x_{i*t}[0]*N[s]}{(Max(x_{i*}[0])-Min(x_{i*}[0]))} - \frac{\Delta N x_{i*t}[0]*N[s]}{(Max(x_{i*}[0])-Min(x_{i*}[0]))} + \Delta N$$

where $x_{i*t}[0]$ is the reference input signal at time t, and ΔN is a small positive integer, usually 1.

The basic principle of this method is to modify the back-propagation learning algorithm to suppress non-dominant causal relationships between the input factors and the neural network outputs. A neuron transfers its inputs to its output and back-propagates its output error to its inputs in a way similar to traditional back-propagation algorithm:

$$x_{j*}[s] = f(\sum_i (w_{j*i}[s] * x_i[s-1])) = f(\ I_{j*}[s])$$

where $x_{j*}[s]$ = current output state of the j-th neuron which is selected for activation in layer s, $w_{j*i}[s]$ = weight of connection joining i-th neuron in layer (s-1) to the j-th neuron in layer s, and $I_{j*}[s]$ = weighted summation of inputs to j-th neuron in layer s, and f is the transfer function. Suppose that the network has a global error function E associated with it: a differentiable function of all the connection weights in the network. Then the error propagated backwards through the network layers is defined by:

$$e_{j*}[s] = \frac{\partial E}{\partial I_{j*}[s]}$$

The local error at a particular neuron at level s and all the local errors at the layers s+1 :

$$e_{j*}[s] = f'(I_{j*}[s]) * \Sigma_k (e_k[s+1] * w_{kj*}[s+1])$$

Given the current set of weights $w_{j*i}[s]$, one needs to know how to adjust them in order to reduce the global error. This can be done using a gradient descent rule as follows:

$$\Delta w_{j*i}[s] = ß * e_{j*}[s] * x_i[s-1] + a * \Delta w_{j*i}[s]$$

where ß is the learning coefficient, and *a* is the momentum rate. The forward propagation of signals is illustrated in figure 2. With this selection scheme, only a small portion of all the neurons are involved in the computation at any instant. Therefore training can be made faster.

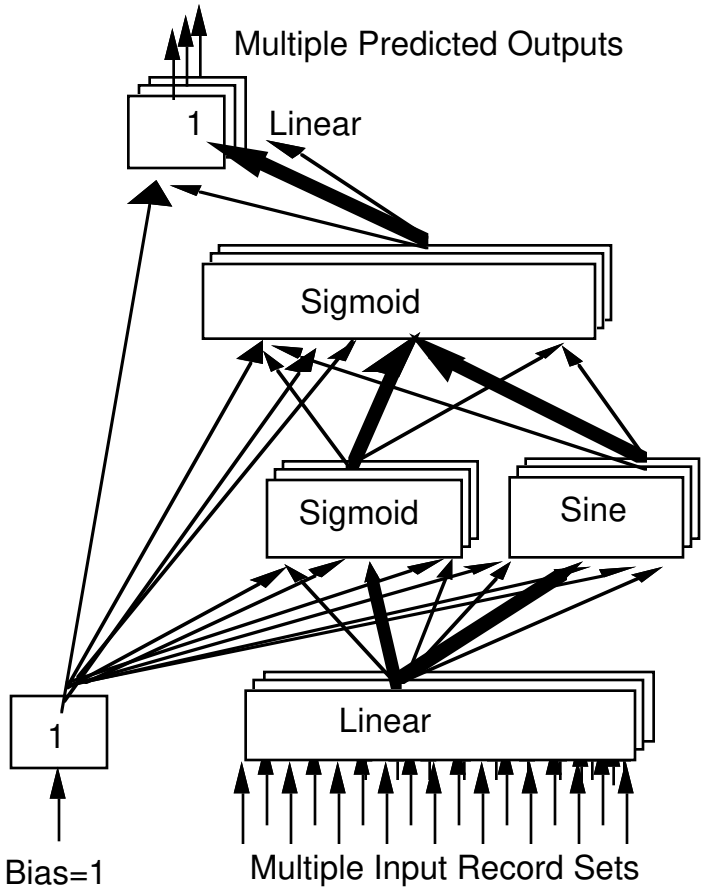


Fig. 2 -- The learning time can be reduced significantly by using the Fastprop alogrithm to activate and train neurons selectively.

2.1. Application Example 1: GDP Forecast

The usefulness of the FastProp method has been investigated using several real world applications. For the Gross Domestic Product (GDP) index forecast, the network was trained with 18 economic indicators for a period of 70 months, up to October, 1991 (the range Learn-Learn in Fig. 3), and produced the forecast for the next 6 months (the range Learn-Forecast in Fig. 3). It was found that the training time for FastProp is merely 15% of that required by the traditional back-propagation algorithm, and it produced a better forecast accuracy. The indicators used include several foreign currency exchange rates, import and export, national reserves, CPI, money supply, unemployment rate, trade balance, current account balance, industrial productivity index. The forecast horizon was 6 months ahead, and the forecast was done in October, 1991.

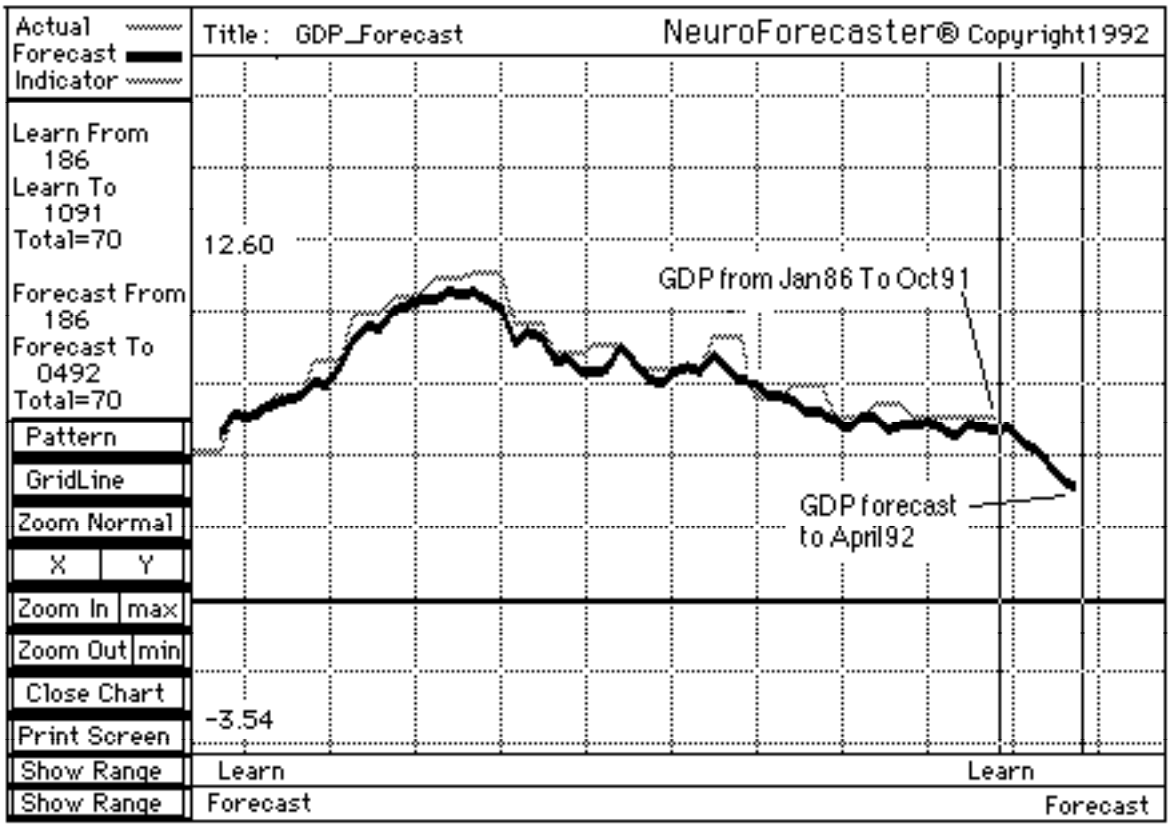
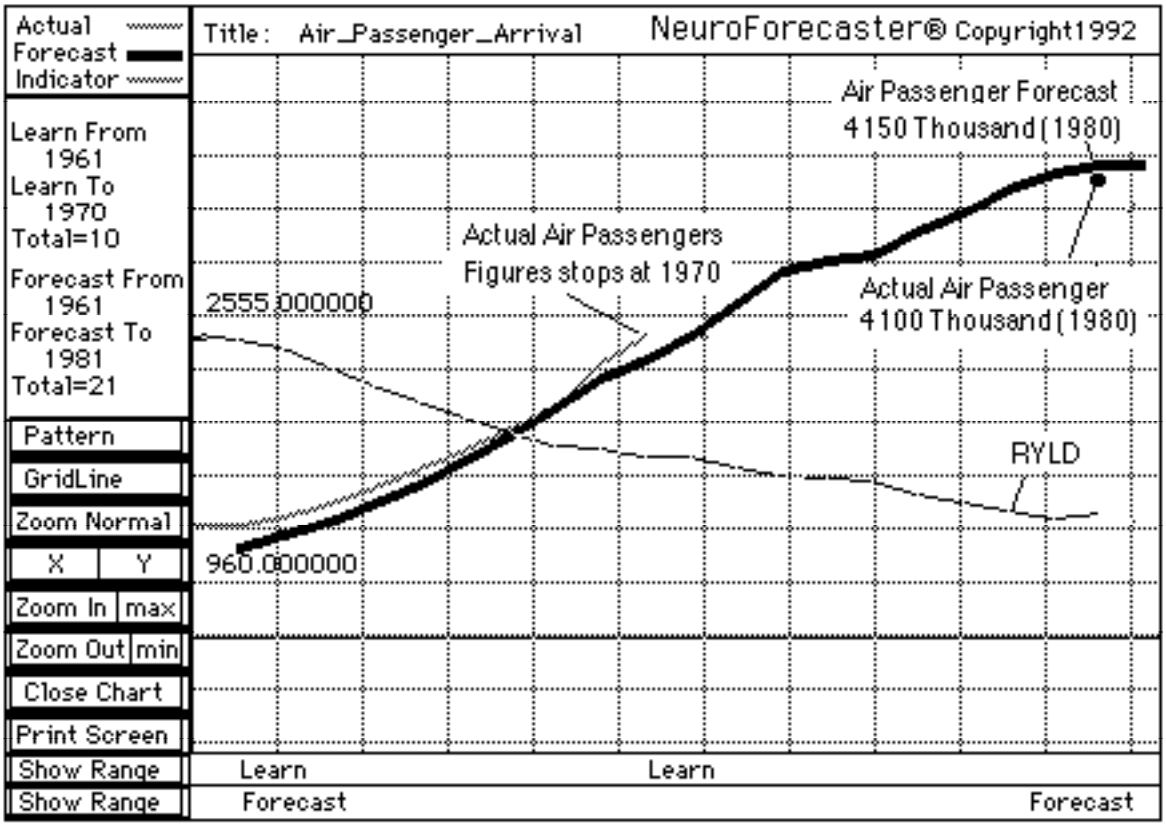


Fig. 3 -- The 6-month ahead forecast of Singapore's National GDP. The thin line represents the actual quarterly GDP, and the thick line is the forecast. The forecaster was trained using a 6-month window to perform 6-month ahead forecast.

2.2. Application Example 2: Air Passenger Arrival Forecast

In another study, the network was trained with 3 indicators for a period of 10 years, up to 1970 (the range Learn-Learn in Fig. 3), and produced the forecast for the next 11 years (the range Learn-Forecast in Fig. 3). The indicators used include RPK (World scheduled revenue passenger kilometers), RGDP (World Gross Domestic Product in real terms) and RYLD (World scheduled airline passenger yield in real terms). The forecast horizon was 1

year ahead. Compared to the linear regression method, the neural network produced much more accurate results.



2.3. Discussions

There have been several claims made with reference to the accuracy of the neural networks in economic forecasting: 65% in the forecasting of S&P500 weekly closing prices, around 80% in the predicting of the bond rating [Dutta 89], 96% in stock market forecasting [Schoneburg 90]. However, there are several key factors, besides the neural network design, which influence the accuracy:

(1) the nature of the applications
(2) the indicators used (leading/lagging/coincidental/noise, complete/incomplete)
(3) the forecast horizon (number of steps ahead)
(4) definition of "hit" and "miss" (tolerance of error)
(5) the sampling rate (daily/weekly/monthly)
(6) the training strategy (network structure, parameters)

The generally good performance found for the FastProp algorithm may sometimes have to be traded off with local minimum problems. Heuristics to avoid this problem include:
(1) re-training the network many times each with different initial weights;
(2) lowering the Learning Coefficient; and,
(3) adding extra hidden layers.

3. Genetic Algorithm for automatic neural network design

Genetic algorithms have been increasingly applied in conjunction with neural networks [Belew 91]; two primary areas of activity being topology optimization and genetic training algorithms. In topology optimization, genetic algorithm is used to select a topology (pattern of connectivity) for the neural network which in turn is trained using some fixed training scheme, most commonly back-propagation. In genetic training algorithms, the learning of a neural network is formulated as a weights optimization problem, usually using the inverse mean squared error as a fitness measure.

Although the interest in genetic algorithms from the connectionist community is fairly recent, this class of adaptive search procedures was initially introduced in John Holland's seminal work back in 1975 [Holland 75]. His analysis of natural adaptive systems shows that natural evolution embodies an elegant generate-and-test strategy that can rapidly identify and exploit regularities in the environment. Most importantly, he produced a formal result which states that even in large and complicated search spaces, given certain conditions on the problem domain, genetic algorithms would tend to converge on solutions that were globally optimal or nearly so.

There are four stages in the genetic search process : initialization, selection, crossover and mutation. In the initialization stage, a population of genetic structures which are randomly distributed in the solution space is selected as the starting point of the search. After the initialization stage, each structure is evaluated using a user-defined fitness function and assigned a utility value. On the basis of their relative utility values, structures in the current population are selected for reproduction. A stochastic procedure ensures that the expected number of offspring associated with a given structure s is u(s)/u(P,t), where u(s) is the observed performance of s and u(P,t) is the average performance of all structures in the current population. Thus structures with high performance may be chosen for replication several times while poor-performing structures may not be chosen at all. In the absence of other mechanisms, such a selective pressure would cause the best performing structures in the initial population to occupy an increasingly larger proportion of the population over time.

The selected structures from the previous phase are recombined using crossover, which operates by swapping corresponding segments of a string representation of the parents. Crossover serves two complementary search functions. First, it provides new points for further testing of schemata[1] already present in the population and secondly, it introduces instances of new schema into the population.

However, crossover draws only on the information present in the solutions of the current population in generating new solutions for evaluation. If specific information is missing (due to storage limitation or loss incurred during the selection process of a previous generation), then crossover is unable to produce new structures that contain this piece of information. A mutation operator, which arbitrarily alters one or more components of a selected structure, provides the means for introducing new information into the population. However, mutation functions as a background operator with a very low probability of application. The presence of mutation ensures that the probability of reaching any point in the search space is never zero.

---

[1]Holland has defined a schema as describing a subset of strings (in a population) with similarities at certain string positions. For example, consider strings of length three over the alphabet {0, 1}. The two strings 011 and 111 are similar in the sense that they are identical when the first position is ignored. Regarding # as a symbol which may be instantiated to either 0 or 1, these two strings may therefore be represented by #11. Strings over the alphabet { #, 0, 1 } are the schemata (plural of schema) of Holland, and play a central role in the generally accepted explanation of why genetic algorithms work.

3.1. Application Example: Currency exchange rates forecasting

We investigate the usefulness of genetic algorithm and neural network hybrid on a real application: currency exchange rates forecast. Our currency exchange rates database consists of US-Singapore currency exchange rates and 7 financial ratios reflecting the fundamental characteristics of the time series. The database was prepared by analysts at a major financial institution. The data are of monthly nature, collected from the period of Jan 1986 to May 1991. The currency exchange rates time series is depicted in figure 4.

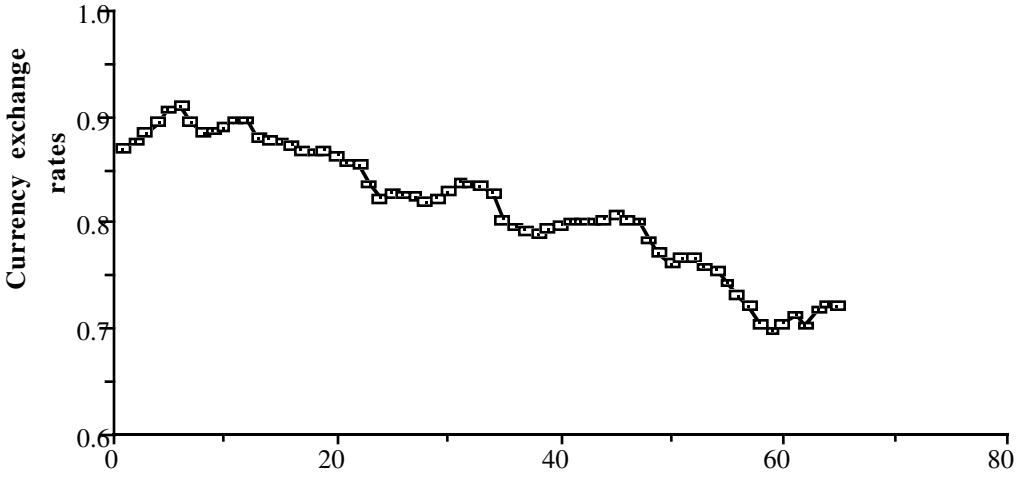


Fig. 4 -- Currency exchange rates time series

Our method of automating neural network architecture design combines two adaptive processes: genetic search through network architecture space, and backpropagation learning in individual networks to evaluate the selected architectures. This generic process is mapped out in figure 5.

In our method, as in real biological systems, cycles of learning in individuals are nested within cycles of evolution in populations. Each learning cycle presents an individual neural network - an instantiation of a particular genetic description - with the set of input-output pairs from the currency exchange rates database defining the task. The backpropagation learning algorithm then compares the network's actual outputs with the desired outputs, and modifies the network's connection weights so that it performs the desired input-output mapping task more accurately. Each evolution cycle processes one population of network designs according to their associated fitness values (computed during the learning cycles) to yield an offspring population of more highly adapted network designs.

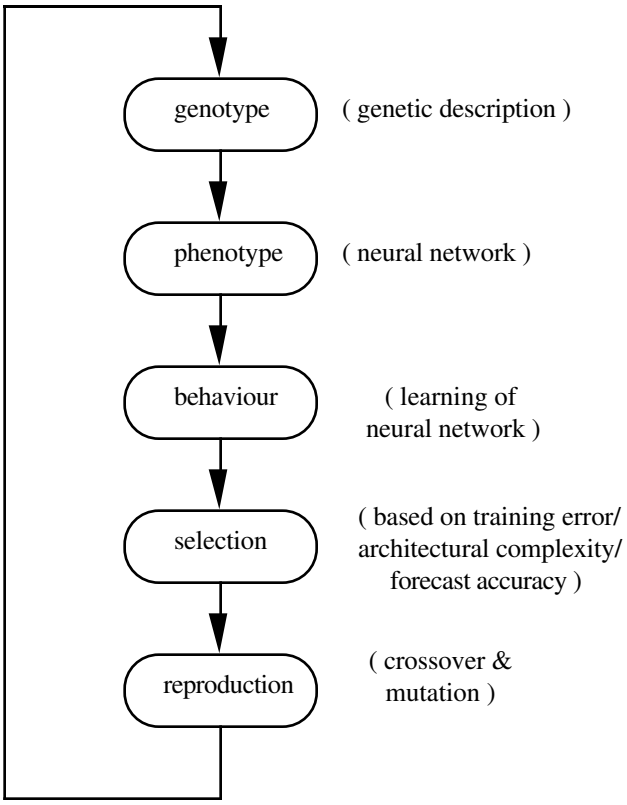


Fig. 5 -- Evolutionary process. The genes lead via development to a structured network, including interconnections. The network then responds to the environment and is selected based on the resulting fitness score, which depends upon the posited task. The fittest individual reproduces to yield the genotypes in the next generation, and the evolutionary process continues.

There are many aspects of a neural network which can be optimized using genetic algorithm [Tan 92]. In this paper we will focus on the issue of selecting input attributes. A prevalent problem faced by neural network designers in forecasting economic time series is limited sample size. To compound this problem, the number of factors bearing on the task at hand is usually myriad, and even experts have a hard time agreeing on the relative importance of these factors. The natural inclination is to use as many factors as possible, but this gives rise to the problem of overfitting.

An illuminating analogy can be drawn between backpropagation learning and curve fitting. When using a curve model (say a polynomial) with lots of parameters compared to the number of points, the fitted curve will closely model the training data but will not be likely to accurately represent the new data. On the other hand, if the number of parameters in the model is small, the model will not necessarily represent the data but will be more likely to capture the regularity of the data and extrapolate (or interpolate) correctly. When the data is not too noisy, the optimal choice is the minimum size model that represents the data.

We formulate the problem of searching for minimal representative subsets of input attributes as a bit-string optimization problem in genetic algorithm. Each input attribute set is represented by a binary bit string with N bits, where N is the total number of input attributes. Each bit determines whether an input attribute is used in the training phase of the neural network. A bit "1" indicates that an input attribute is present in a subset, and bit "0" indicates that the input attribute does not belong to it.

Table 1 shows the performance of 5 minimal representative subsets of input attributes generated by genetic algorithm, compared against a fully forward-connected network. Better prediction accuracy was achieved with all these subsets, with improvements of 31% to 69%. In addition, training time was reduced drastically by 47% to 80% and the space complexity of the neural network was decreased by 50% to 83%. The smallest mean square prediction error was obtained with a subset of input attributes 1 and 7. This combination produced forecast results that are 70% more accurate than when the entire set (of 8 input attributes) was used. In all the subsets obtained, input attribute 1, which is the current currency exchange rate, was always present. This emphasized the relevance of using the past value of a variable to forecast its future values.

Representative Subset	Mean square prediction error	Relative time complexity	Relative space complexity
1, 7	0.000085	0.2	0.17
1, 5, 6	0.000190	0.29	0.26
1, 3, 4, 6	0.000134	0.4	0.37
1, 2, 3, 5, 8	0.000180	0.53	0.50
1, 2, 4, 6, 7	0.000135	0.53	0.50
All 8 input attributes	0.000275	1.0	1.0

Table 1 -- Performance of minimal representative subsets obtained by genetic algorithm

3.2. Discussions

We showed an example where optimizing the network architecture improves both learning speed and generalization performance dramatically. This is really not surprising but it is more easily said than done. However, we have shown that it *can* be done in at least one case, currency exchange rates prediction, using a backpropagation model. These techniques can be easily extended to other time series forecasting applications, as well as to other domains.

The scope of this present work is very limited; indeed, it is intended primarily as a vehicle for presenting the relevant issues in a relatively uncomplicated setting, and for illustrating the relevant approaches. We are just beginning to collect techniques and to understand the principles which can help us design a network for a particular task.

An important limitation of the present approach is that the optimization methods used here does not guarantee an optimum solution , but that, under certain conditions, the solutions will converge to the optimum. Some enhancements, such as performing local searches after finding optimal regions, can be considered.

References:

[Atlas] Atlas L, Cole R., "Performance Comparisons Between Backpropagation Networks and Classification Trees on Three Real-World Applications," Dept. of EE, University of Washington, Seattle, Washington, 98195.

[Baba 90] K. Baba and et. al., "Explicit Representation of Knowledge Acquired From Plan Historical Data Using Neural Network," IJCNN'90, Wash-D.C., V2, pp II-579-582.

[Dutta 89] S. Dutta & S. Shekhar, "An Artificial Intelligence Approach To Predicting Bond Rating," Expert Systems In Economics, Banking and Management, L Pau (Ed), Elsevier Science Publisher, 1989.

[Hanke 89]J. Hanke & A. Reitsch, Business Forecasting, 3rd ed., 1989, published by Allyn & Bacon.

[Harvey 89] Harvey, A. Forecasting, structural time series models and the Kalman filter.

[Holland 75] Holland, J. (1975). Adaptation in natural and artificial systems.

[Lapedes 87] Lapedes, A., R. Farber (1987). "How neural nets work" Advances in Neural Information Systems.

[Parker 85] D Parker, "Learning-Logic", Report TR-47, MIT Center for Computational Research in Economics and Management Science.

[Rumelhart 86] D. Rumelhart, G. Hinton & R. Williams, Parallel Distributed Processing, 1986. Cambridge, MA, MIT Press.

[Schoneburg 90] E. Schoneburg, "Stock Price Prediction Using Neural Networks: A Project Report," Neurocomputing 1. Elsevier, 1990.

[Sharda 90] Sharda, R. & Patil, R.B., "Neural Networks as Forecasting Experts: An Empirical Test," Proceedings of the IJCNN, Washington, pp 491-494.

[Tan 92] PanYong Tan, "Automatic election of neural network architectures via genetic algorithm", MSc thesis in preparation.

[Tang 91] Tang, Z., et al, "Time Series Forecasting Using Neural Networks vs. Box-Jenkins Methodology", SIMULATION, 303-310, Nov. 1991.

[Varfis] Varfis, A, Versino, C, "Univariate Economic Time Series Forecasting by Connectionist Methods," Commission of the European Communities, Joint Research Center, Ispra Establishment, CITE, 21020, Italy.

[Windsor] Windsor C.G, & Harker A.H., "Multi-variate Financial Index Prediction - A Neural Network Study", B521.2, Harwell Lab, Oxon, OX11 ORA.

[Wong 90] F. Wong, "Time Series Forecasting Using BackPropagation Neural Networks," Neurocomputing, 2 (1990/91) 147-159, Elsevier.

[Wong 91] F. Wong, "FastProp: A Selective Training Algorithm For Fast Error Propagation," Proceedings of the IJCNN, 1991, Singapore, pp 2038-2044.