An Artificial Neural Network - Genetic Based Approach for Time Series Forecasting *

José Neves and Paulo Cortez
Departamento de Informática
Universidade do Minho
Largo do Paço, 4719 Braga Codex, Portugal
{jneves,paulo}@di-ia.uminho.pt

Abstract

Genetic Algorithms (GAs) are a class of general optimization procedures, a class of randomized optimization heuristics based loosely on the biological paradigm of natural evolution. Artificial Neural Networks (ANNs) are well established optimization procedures in the domains of pattern recognition and function approximation, whose properties and training methods have been well studied. Recently, there has been some successful applications of ANNs in a new arena, that of sequential decision making under uncertainty (or stochastic control), where one's goal is the cost-to-go or cost function, which evaluates and guides management or control decisions in an organization. In this work one reports on the integration of GAs and ANNs, in terms of a new paradigm, the GANN's one, which will be applied to forecast sun spots, airlines and yields (eg. from batch chemical processes).

1 Introduction

Nowadays one assists to a globalization of the world economy, with the enterprises competing one another, aiming to deliver better products or services at competitive prices, where a multiplicity of different resources are to be considered. Only those with the best structures, processes and technologies may have a chance to survive, being innovation one of the most important advantages. But innovating in an environment of uncertainty can be a disaster. Thus, it is natural to assume that organizations will be interested on obtaining well-grounded forecasts, specially for decision making [9]. One way is to use *Time Series Forecasting (TSF)*, forecast of a chronological ordered variable. This approach tries to predict the behavior of a complex system and not how it works. The purpose in *TSF* is to discover patterns in its historical data in order to extrapolate.

Conventional TSF methods base their models on *Time Series (TS)* components such as *trend* or *seasonal* effects [11]. These methods give accurate forecasts on well behaved *TS*, but present some drawbacks on *TS* with noise or some unknown nonlinear relations among the *TS* data.

A promising ground for *TSF* is the use of *Artificial Neural Networks (ANNs)*. Indeed, *ANNs* have the ability to extract implicit dependencies among data, even when there is not an understanding of the nature of the dependencies or when the data presents a strong noise component, and; *ANNs* can adapt their behavior as new data comes in, without any need for complex redesigns to their structure [14]. Comparative studies [4][3][5] have shown that *ANNs* can perform as well as or even better than conventional methods such as the Holt-Winters [9] and the Box-Jenkins ones [2]. Other studies revealed success on a wide range of applications, such as the forecasting of bond markets [19], traffic [18] or electric power systems control [12]. But *ANNs* can not be seen as an universal solution for all problems. In fact, a problem with the use of *ANNs* resides on the searching time spent on the search for the best *ANN* architecture; here one can use random search, hill-climbing or a genetic programming approach [17]. The last one is appealing for problems of combinatorial nature, being effective where other methods seem to fail [7]. The work described in this paper reports on the integration of *Genetic Algorithms (GAs)* and *ANNs*, taking advantages of both approaches for *TSF* [10].

2 Time Series Analysis

A statistical instrument for *TS* analysis is the *autocorrelation* coefficient (r_k), which gives a measure of the statistical correlation between a *TS* and itself, lagged of k periods,

*This work was supported by JNICT throught the PRAXIS XXI grant

being given by the expression [11]:

$$r_k = \frac{\sum_{t=1}^{n-k}(X_t - \overline{X})(X_{t+k} - \overline{X})}{\sum_{t=1}^{n}(X_t - \overline{X})}$$

where X1,X2,...,Xn stand for the *TS*, and t for period. Autocorrelations are useful for decomposition of the *TS* main components (*trend* and *seasonal* effects) and for the detection of the *TS* randomness. A *trend* is characterized by a constant growth or decline of the data series. This may be due to factors like *inflation, technological improvements*, and so on. The autocorrelations of a *TS* with a *trend* present a clear distinctive pattern (Figure 1). This suggests that an *ANN* trained with the autocorrelations of various *TS* could learn how to detect the presence of a *trend*. Theoretically all autocorrelations should be zero for a random series. In practical terms a *TS* is random if, for all k lags, the autocorrelations are within the range:

$$-1.96 * \frac{1}{\sqrt{n}} \leq r_k \leq 1.96 * \frac{1}{\sqrt{n}}$$

considering a normal distribution with 95% degree of confidence. This test will be of use to the GANN's systems.

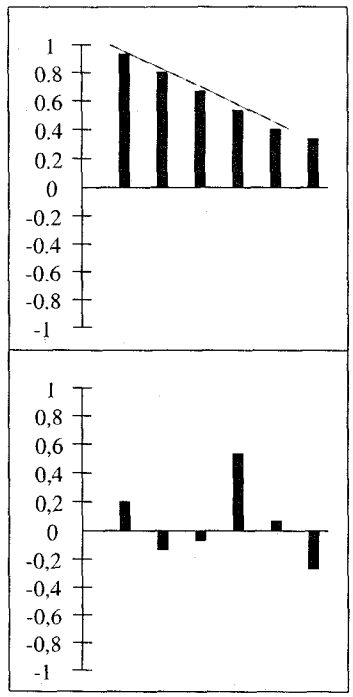

Figure 1: Autocorrelation coefficients for a *TS* with and without a *trend*.

3 Artificial Neural Networks

When using *ANNs* some parameters have to be set since the very beginning. This choice is done by looking at the data and to the needs of the problem under consideration [6][15]. Although some studies point to the benefits of using recurrent *ANNs* [18][12], most of the research under way on this area has focused its attention on feed-forward nets [4][14][3][5]. To cut the searching space it was decided to used fully connected feed-forward *ANNs* with bias, without shortcut connections and with one hidden layer. The structure of layers will be given by productions in the form $L_i - L_h - L_o$ for an *ANN* with Li input nodes, L_h hidden nodes and L_o output nodes. The initial weights are random, within the range of $[\frac{-2}{z}; \frac{2}{z}]$ for a node with z inputs [6]. The ***Resilient BackPROPagation (RPROP)*** algorithm was used to perform the training [16], and as activation functions were used those of Table 1 [1].

Table 1: Activation Functions.

Name	Function $f(x)$	Codomain
linear	x	$]-\infty, +\infty[$
sigmoid	$\frac{1}{1+\exp(-x)}$	$[0,1]$
sigmoid1	$\frac{2}{1+\exp(-x)} - 1$	$[-1,1]$
sigmoid2	$\frac{x}{1+\lvert x\rvert}$	$[-1,1]$
tanh	$\tanh(x)$	$[-1,1]$
cos	$\sin(x \bmod 2\pi)$	$[-1,1]$
sin	$\cos(x \bmod 2\pi)$	$[-1,1]$
gaussian	$\exp(\frac{-x^2}{2})$	$[-1,1]$

One step ahead forecast is done by using a moving time window of n lags from the *TS*, for an *ANN* of n inputs. The goal is to have the output, the forecasted value, as a function of the n previous ones. This approach simplifies the computational process once fewer *ANN's* weights need to be estimated [5]. Thus the generic *ANN* topology may be given in the form $n - nh - 1$.

Some activation functions require the data to be in a certain range (Table 1); the data must be rescaled within the range $[0, 1]$, if the *TS* is stationary, or within the range $[0.2, 0.8]$, if there is a *trend* component. The detection of the *trend* component is done using another *ANN*. The inputs to this network are the autocorrelation coefficients, for 1-10 lags, of 18 *TS*, having an output of $+1$, if there is a *trend*, and -1 if not. After some runs it was decided to use an *ANN* with the topology $10 - 5 - 1$, and as the activation function the *sigmoid1* one.

To avoid problems of overfitting, early stopping [15] was implemented. The *TS* data was divided into two categories: a training set (to assimilate the patterns) and a validation set (to test if the *ANN* had generalization capacity). Since recent data affects strongly the forecast, the system will use only 10% of the training data in the validation set.

4 The Genetic Algorithm

Within the *GANNs'* systems, *ANNs* evolve by natural selection and learn by backpropagation [10]. One's system uses the Baldwin learning approach [8] where the *ANNs* learning guides the evolutionary search. If an encoded *ANN* is near to the optimum, then by backpropagation will reach that optimum, resulting in a good fitness value [10]. Solutions (chromosomes) near the optimum will share a big amount of bit patterns, and therefore the *GA* will be able to exploit them by hyperplane sampling. Since the *GA* works in this case as a second order optimization procedure, and it is guided by observation rather than by theory, it was decided to use a population of m individuals (in this case m was set to 30), rank-based *selection* [10], one point *crossover* with a crossover rate of 1 and a *mutation* rate of 0.02. As the *fitness* function the system uses the *MSE (Mean Squared Error* calculated for the validation cases), namely:

$$fitness = \frac{1}{MSE}$$

In Figure 2 one can see how the process works. At the beginning a set of m individuals are randomly generated, being the *ANN* designed according to the genome information. The fitness value is determined after training the *ANN*. After evaluation all individuals are ranked. *crossover* and *mutation* operations will create a new population of individuals, which will be also evaluated. Finally, the rank-based *selection* operation will select the best individuals from both populations, leading to a new generation. This process goes on until some stopping criterium is achieved (in this case after g generations).

After some runs of the *ANNs* it became evident that forecasting is a function of: the number of input nodes (in), the RPROP algorithm parameters (Δ_0 and Δ_{max}), the activation function (f), the number of hidden nodes (hn), and the random weights initialization seed (s). This can be easily explained: the first factor in sets the size of the *TS* time window, while the others set how the *ANN* learns. The number of nodes in and hn are set within the range $3, ..., 14$ allowing a binary encoding of 4 bits; there is some empirical evidence that this is the correct range and that a generous one will affect badly the searching space [4]. Δ_{max} was set to 50 (value advised in [16]) and Δ_0 was encoded into 3 bits using discrete values within the range $0.1, ..., 0.8$. The activation functions were also encoded into 3 bits.

The GANNs' parameters were encoded using base 2 gray codes, with the most influent ones located at the left of the chromosome (Figure 3).

5 The System Architecture

The system was developed using the UNIX operating system, the SICStus Prolog and the C languages, with a X-

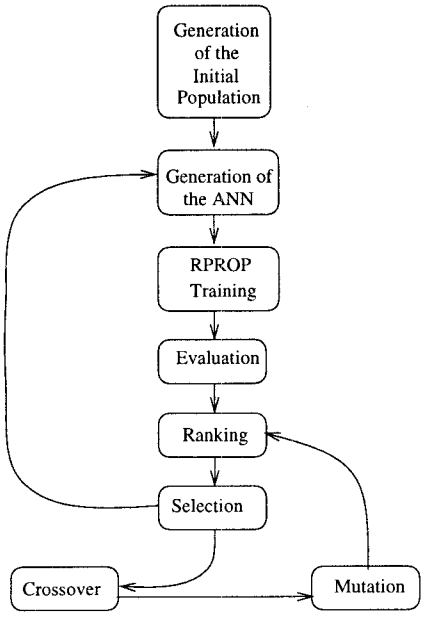

Figure 2: The structure of the *GANN* system.

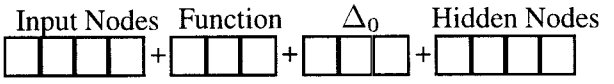

Figure 3: Neural Network Encoding.

Windows interface [13]. The whole system presents three main modules (Figure 4): the *preprocessor*, the *GANN* and the *interface* ones. The former one calculates the autocorrelations factors and checks for *TS* randomness. In this process, a special value, *avg*, the average of the *TS* when forecasting, it is returned. If the series is not random, it checks for a *trend*, in order to set the scaling domain, and the training cases will be created according to the size of the time window. The second one handles the search for the best *ANN* for *TSF*. Finally the interface one presents the user with the desired forecasts.

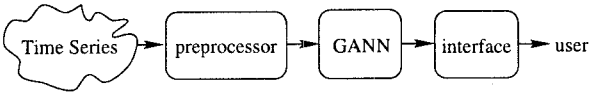

Figure 4: The system main modules.

6 Results

In this work were used series of real data from different types of sources (including the annual number of sun

spots, airlines and yields [2][11][9]. The results were compared with the ones obtained through two well known conventional methods: the Holt-Winters [11] and the Arima [2] ones. The former one works well on seasonal *TS*, being the last one more general purpose. Figure 5 shows the values of the sun spots series while Figure 6 shows sun spots forecasts. Table 2 shows the *ANNs* modeled by the system for each series, while Table 3 compares the system's results with those obtained from using the Arima and the Holt-Winters methods.

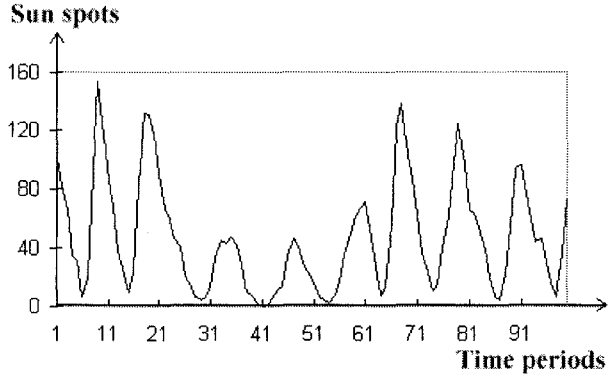

Figure 5: Sun spots values.

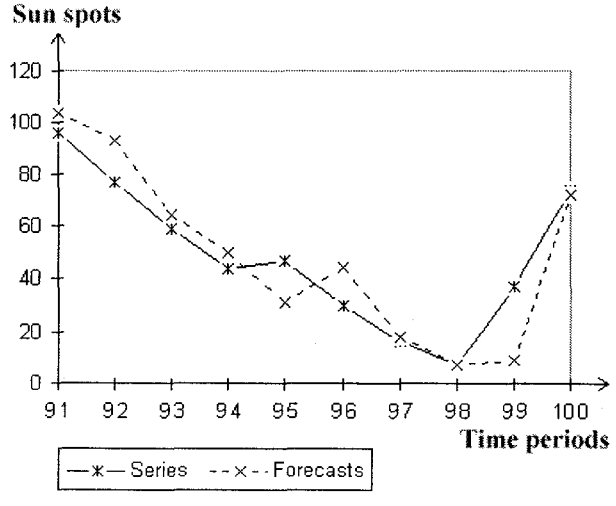

Figure 6: One-step ahead sun spots forecasts for 91 to 100 time periods.

The Holt-Winters method outperforms other methods for seasonal series such as series 2, 3 and 4. This is easily explained since this method was developed for this particular kind of well behaved *TS*. Series 4 and 6 have a strong *trend* component. The system outperforms the Arima one for the last series. Finally series 1, 7, 8 and 9 are series without *sea-*

Table 2: The best *ANN* for each serie.

Series	Size	Trend	Topology	Function	Δ_0
1	100	No	$4-4-1$	*sin*	0.7
2	144	Yes	$13-11-1$	*linear*	0.4
3	130	Yes	$13-3-1$	*linear*	0.8
4	261	Yes	$7-9-1$	*linear*	0.2
5	115	Yes	$12-8-1$	*gaussian*	0.5
6	355	Yes	$4-3-1$	*cos*	0.8
7	65	No	$8-8-1$	*sigmoid2*	0.8
8	70	No	$7-6-1$	*sin*	0.8
9	75	No	$6-6-1$	*sigmoid2*	0.2

Table 3: Comparing the system results with other methods (using the MSE as the metric for the forecasts).

Series	Forecasts	System	Arima	Holt-Winters
1	10	139.6	154	-
2	12	340.3	452	271
3	12	834578	-	530654
4	10	63.9	36.3	-
5	12	20672	15290	11435
6	10	62.87	72.1	-
7	10	3666	2939	-
8	10	66.8	141.5	-
9	10	265	240.4	-

sonal and *trend* components. The systems results are in the same order of similarity to those obtained by Arima (for series 1 and 8 the system's results are even better).

7 Conclusions

The prediction of the *TS* is a classical problem in the applied mathematics field, in particular in the economical and control arenas. The results obtained so far suggest that *ANNs* can be a promising alternative for *TSF* , specially on series with high degree of non-linearity. The *GANN's* systems have the disadvantage of being more demanding in computational terms than the Holt-Winters and the Arima ones; however, while the Arima's systems requires the use of an expert analyst, the *GANN's* systems work on its own, with a minimum of human intervention.

In the future it is one's intention to explore the use of data filtering and other approaches to *ANNs*, such as *ANNs* with shortcut connections or the self-organizing ones (eg. following the Kohonen approach to computing).

References

[1] E. Azoff. *Neural Network Time Series Forecasting of Financial Markets.* John Wiley & Sons, 1994.

[2] G. Box and G. Jenkins. *Time Series Analysis: Forecasting and Control.* Holden Day, San Francisco, USA, 1976.

[3] P. Cortez, J. Machado, and J. Neves. An Evolutionary Artificial Neural Network Time Series Forecasting System. In *IASTED International Conference on Artificial Intelligence, Expert Systems and Neural Networks*, Honolulu, Hawai, August 1996.

[4] P. Cortez, M. Rocha, J. Machado, and J. Neves. A Neural Network Based Forecasting System. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, Perth, Western Australia, November 1995.

[5] J. Faraday and C. Chatfield. Times Series Forecasting with Neural Networks: A Case Study. Research Report, Statistiscs Group, University of Bath, UK, 1995.

[6] S. Gallant. *Neural Network Learning and Expert Systems.* MIT Press, Cambridge, USA, 1993.

[7] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Publishing Company, Inc., NY, USA, 1989.

[8] F. Gruan and D. Whitley. Adding Learning to the Cellular Development of Neural Networks: Evolution and the Baldwin Effect, Volume 3. In *Evolutionary Evolution.* MIT Press, Nature, 1989.

[9] J. Hanke and A. Reitsch. *A Business Forecasting.* Allyn and Bancon Publishing Company Inc., Massachussetts, USA, 1989.

[10] P. Koenh. Combining Genetic Algorithms and Neural Networks. M.Sc. Thesis, University of Tennessee, Knoxville, USA., 1994.

[11] S. Makridakis, S. Weelwright, and V. McGee. *Forecasting: Methods and Applications.* John Wiley & Sons, New York, USA, 1978.

[12] J. Mandal, A. Sinha, and G. Parthasarathy. Application of Recurrent Neural Networks for Short Term Load Forecasting in Electric Power Systems. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, Perth, Western Australia, November 1995.

[13] J. Neves and J. Machado. The Application of Webbaed Multi-agent Technology to Simulation Modelling. In Proceedings of EUROMEDIA 96 - Telematics in a Multimedia Environment (a Publication of the Society for Cumputer Simulation International), London, UK, December 1996.

[14] G. Papadourakis, G. Spanoudakis, and A. Gotsias. Application of Neural Networks in Short-Term Stock Price Forecasting. In *First International Workshop on Neural Networks in the Capital Markets*, London, UK, November 1993.

[15] L. Prechelt. PROBEN1 - A Set of Neural Network Benchmark Problems and Benchmarking Rules. Research Report, Fakultät für Informatik, Universität Karlsruhe Germany, 1994.

[16] M. Riedmiller and H. Braun. A Direct Adaptative Method for Faster Backpropagation Learning: The RPROP Algorithm. In *Proceeedings of the IEEE International Conference on Neural Networks*, San Francisco, CA, USA, 1993.

[17] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach.* Prentice-Hall, New Jersey, USA, 1995.

[18] C. Ulbricht. Multi-recurrent Networks for Traffic Forecasting. In *Proceedings of AAAI'94 Conference*, Seatle, Washington, USA, July 1994.

[19] J. YAO and H. POH. Forecasting the KLSE Index Using Neural Networks. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, Perth, Western Australia, November 1995.