# Automatic Design of Artificial Neural Networks to Forecast Time Series

Juan Peralta
Computer Science Department
University Carlos III of Madrid
28911 Madrid
jpetalta@inf.uc3m.es

German Gutierrez
Computer Science Department
University Carlos III of Madrid
28911 Madrid
ggutierr@inf.uc3m.es

Araceli Sanchis
Computer Science Department
University Carlos III of Madrid
28911 Madrid
masm@inf.uc3m.es

**Abstract**

In this work an approach to design Artificial Neural Networks (ANN) to forecast Time Series is tackled. The approach is an automatic method that is carried out by. an Evolutionary Algorithm (as a search algorithm) to design ANN. A key issue for these kinds of approaches is what information is included into the chromosome that represents an ANN There are two principal ideas about this question: first, the chromosome contains information about parameters of the topology, architecture, learning parameters, etc. of the ANN. The results using a parameter Encoding Scheme to design ANN for a Time Series Competition are shown.

## 1. Introduction

In order to acquire knowledge, it is so interesting to know what the future will look like, i.e. forecast the future from the observed past. Forecasting can be explained as the process of estimation in unknown situations. In the case of Time Series it consist of using a model to forecast future events based on known past events, i.e. to forecast future data points before they are measured. The forecasting task can be performed by several techniques as Statistical methods, Immune Systems, and Artificial Neural Networks (ANN).

This contribution reports the methodology to carry out the automatic design of ANN that tackles the data mining task (i.e. Forecasting Time Series).

The initial aim of this work was to carry out the forecast for the NN3, NN5 and NN GC1 Time Series competitions. used in this document have been taken from CEDI Forecasting Time Series Competition [1], the future values to be forecasted of these Time Series are still not known.

In previous works for those competitions the automatic design of ANN was done as a search process carried out by a Genetic Algorithm (GA). Nevertheless, in this work the search process will be carry out by an outgrowth of GA, i.e a kind of Estimation of Distribution Algorithms (EDA).

This paper is organized as follows. Sec 2 reviews questions about forecasting task with ANNs and ANN design with evolutionary Algorithms. Sec 3 approaches design of ANN to forecast with Estimation Distribution Algorithms. In Sec 4 experimental setup and results are shown. And finally, conclusions and future works are described in Sec 5.

## 2. Related Work

Several works have tackled the forecasting Time Series task with ANN, not only computer science researchers, but statistics as well [3]. This reveals the full consideration of ANN (as a data driven learning machine) into forecasting theory [4].

### 2.1. **Time** Series **and** ANN

The problem of forecasting time series with ANN is considered as obtaining the relationship of the value of period "t" (i.e. the net has only one output neuron) and the values of $k$ previous periods $(t\text{-}1, t\text{-}2, ..., t\text{-}k)$ ec. (1)

$$a_t = f(a_{t-1}, a_{t-2}, \ldots, a_{t-k}) \qquad (1)$$

Therefore the ANN that maps $a_{t-1}, a_{t-2}, \ldots, a_{t-k}$ and $a_t$ will have $k$ inputs.

The time series data will be transformed into a pattern set used for learning. That pattern set will have *(nts - k)* patterns, one pattern for each known time series value. Each pattern consists in:

- "k" attributes, that correspond to *"k"* normalized previous values of period t: $a_{t-1}, a_{t-2}, \ldots, a_{t-k}$
- One attribute (output) value, which corresponds to normalized Time Series value of period t that will be forecasted.

The complete patterns set are ordered into the same way the time series is. This patterns set will be used to train and validate the ANN, then it will be split into two sets, train and validation subsets. The train subset will be obtained from the first m% (e.g 70%) and the validation subset will be obtained from the rest of the complete patterns set.

The train subset is used to modify the weights of the ANN by the learning algorithm (e.g Backpropagation), and validation subset is used only to stop the learning algorithm.

The test pattern subset (apart from train and validation subsets) will be the values to forecast.

### 2.2. Evolutionary Computation for ANN design

Before using an ANN to forecast, it has to be designed, i.e. establishing the suitable value for each freedom degree of the ANN [5] (kind of net, number of input nodes, number of outputs neurons, number of hidden layer, number of hidden neurons, the connections from one node to another , connection weights, etc ).

If hand design of ANN is carry out, several topologies (i.e. different number of inputs nodes and different number of hidden neurons into the only hidden layer), with different learning rates are trained. For each of them, train and test error are obtained, and one with better generalization capability (i.e. less test error and a good train error) is selected to forecast the future values (i.e. unknown values). The design process is more an "art" based on test and error and the experience of human designer, than an algorithm. In [6] Sven Crone show evaluation framework to ANN modelling in forecasting

The problem of designing ANN could be seen as a search problem into the space of all possible ANN. And that search can be done by a GA [7] using exploitation and exploration.

Several works approach the design of ANN as a search process using Evolutionary Techniques like Genetic Algorithms [8],[9]. An important issue about GA is what information is included into the chromosome because the more information is included into the chromosome (i.e. number of genes) the more detail we will have about the architecture that is being looked but at the same time the bigger will be the search space so the process will be slower. Regards this issue, there are some works that use Direct Encoding Schemes (DES). For DES the chromosome contains information about parameters of the topology, architecture, learning parameters, etc. of the Artificial Neural Network. In IES the chromosome contains the necessary information so that a constructive method gives rise to an Artificial Neural Network topology (or architecture) [10]

## 3. ANN design with Estimation of Distribution Algorithms

Estimation of Distribution Algorithms (EDA), also called Probabilistic Model-Building Genetic Algorithms (PMBGA), is an outcome of GA.

It has been observed in literature that few hybrids systems have been carried out applying ANN and EDA, and they have been applied to classification domains [19].

Due to this, here it is proposed a new hybrid method using advantages of EDA to design ANN applied to forecast all kind of time series. Besides, as it is a totally automatic method, it is no necessary the user to be an expert at all in time series analysis or time series forecasting.

There are different kinds of EDA, but for our approach it has been chosen UMDA, with no dependencies between variables, according to [11], due to it is faster and easier to work with them.

Here we have the process for a general EDA:

1. Generate and evaluate an initial population of solutions -> $D_0$
2. Repeat a,b,c,d,e steps for $K$ =0,1,2, until a stopping criterion is met.
   a. Select a subset of solutions from Dl -> $D^{sel}_K$
   b. Estimate the empirical probability distribution of $D^{sel}_K$ -> $P_k(X)$
   c. Sample solutions from Pk(X)-> $D^{NEW}_K$
   d. Evaluate solutions in $D^{NEW}_K$
   e. Create the new population with solutions from $D^{NEW}_K$ and

Algoritmo 1.

For our approach [12] to design ANN to forecast time series, a Parameter Encoding Scheme for a Multilayer Perceptron (MLP) (i.e. ANN Full Connected) has been considered. For this scheme the information placed into the chromosome will be: two decimal digits, i.e. two genes, to codify the number of inputs nodes (i); other two for the number of hidden nodes (h); two more for the learning factor (a); and the last ten genes for the initialization seed value of the connection weights (s) (seed in the ANN simulator, SNNS is "long int" type, that is why it has been used 10 genes (decimal digits) to encode "s"). This way, the values of "i", "h", "a" and "s" are obtained from the chromosome as it can be seen in Figura 1:

Chrom:

$$g_{i1}\, g_{i2}\, g_{h1}\, g_{h2}\, g_{\alpha 1}\, g_{\alpha 2}\, g_{s1}\, g_{s2}\, g_{s3}\, g_{s4}\, g_{s5}\, g_{s6}\, g_{s7}\, g_{s8}\, g_{s9}\, g_{s10}$$

$$0 \le g_{xy} \le 9,\ x = i, h, \alpha,\ y = 1..10$$

$$i = \text{max\_inputs}((g_{i1} \cdot 10 + g_{i2})/100)$$

$$h = \text{max\_hidden} \cdot ((g_{h1} \cdot 10 + g_{h2})/100)$$

$$\alpha = ((g_{\alpha 1} \cdot 10 + g_{\alpha 2}))/100$$

$$s = g_{s1}\, g_{s2}\, g_{s3}\, g_{s4}\, g_{s5}\, g_{s6}\, g_{s7}\, g_{s8}\, g_{s9}\, g_{s10}$$

Figura 1. Chromosome

The search process (EDA) begins with a random population, i.e set of randomly generated chromosomes (with uniform distribution). Later, the fitness value for each one of the individual of the population is obtained. The fitness value is the MSE (minimum square error) on validation set. Once that the fitness for the whole population is already obtained the EDA operators, see Algoritmo 1, are applied in order to generate the population of next generation, i.e. set of chromosomes. The steps *a, b, c, d, e* in Algorithm 1 are iteratively executed till a maximum number of generations is reached.

To obtain the fitness value of a chromosome:

1. The phenotypes (i.e. ANN) of all the individuals of the actual generation are obtained.
2. The train patterns and validation patterns subsets are obtained for each ANN, depending on the number of inputs nodes of the net, as it was said above (sec.2.1).
3. Then, the connection weights are randomly initialized for each ANN, and the net is trained with Back-Propagation.

The fitness function will be related with the validation error. The fitness value for each individual will be the minimum error test reached during the training process; it doesn't have to be in the last training cycle. The architecture of the net (topology + connections weights) when the error test is minimum during the training process is saved to be used later for forecasting Time Series values.

Once that EDA reaches the last generation, the best individual from all generations is used to forecast the time series.

## 4. Experimental setup and Results

The experiments related to our approach have been carried out for the two CEDI-SICO-TADIMA Time Series Competition [2], called "Cuenca Hidrografica" and "Ozone". The Time Series values to be forecasted (test subset) have not been published yet. Therefore, the forecasted values obtained by our system cannot be evaluated or compared with real values.

In Figura 2 and Figura 3 the results of forecasting process can be shown.

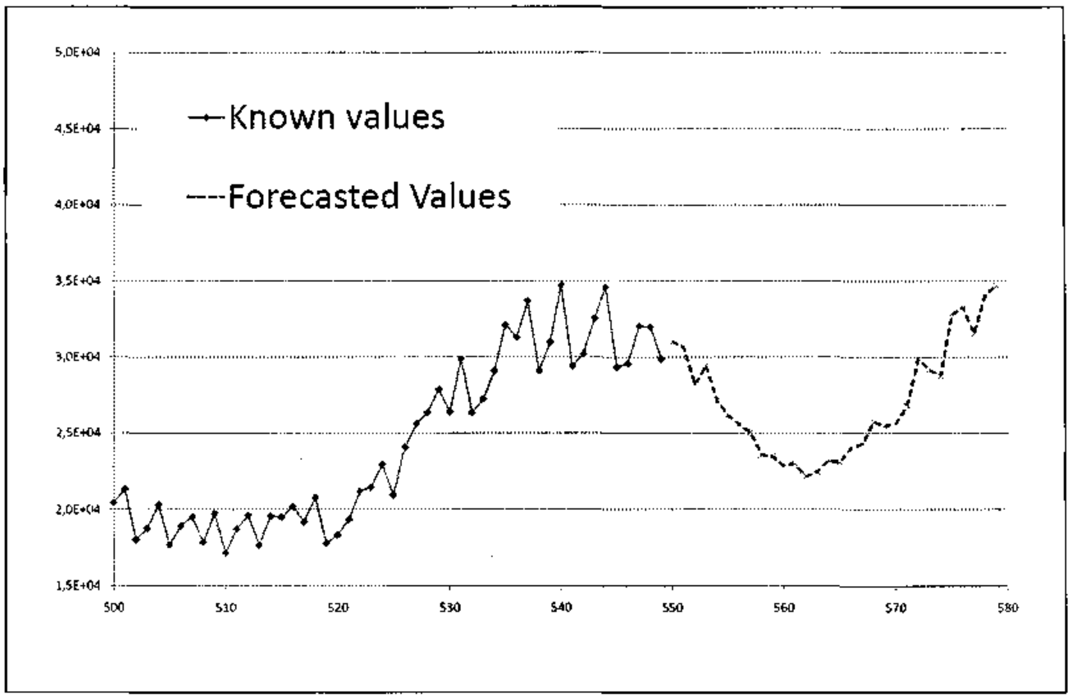


Figura 2. Zoom "Cuenca Hidrografica" Forecast

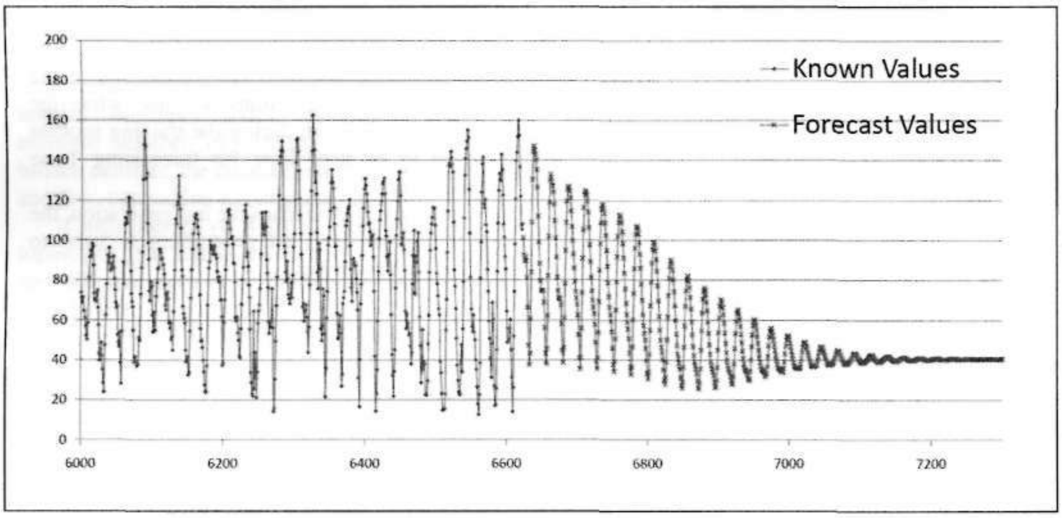


Figura 3. Zoom "Ozone" Forecast.

The parameters for the ANNs obtained from search process are shown in

| Time Series | Inputs | Hidden Nodes | Learning rate |
|---|---|---|---|
| Cuenca Hid. | 64 | 27 | 0.17 |
| Ozone | 73 | 27 | 0.30 |

Tabla 1. Parameter for ANN, obtained from EDA, and used to Forecasting

## 5. Conclusions and Future Works

The approach shown is a totally automatic method, it will not be necessary any previous knowledge from the user so the user will not have to be an expert in time series, statistics, mathematics or computational intelligence. The user just have to give the time series he wants to forecast and the number of future elements he wants to be forecasted to the system; and this method will give these forecasted values as result to the user.

This approach was presented, applying GA instead of EDA, as an automatic method to design ANN in NN-5 competition, getting the 6th position with SMAPE error of 21.9% in Neural Nets and Computational Intelligence methods (NNCI) ranking, for the reduced dataset (i.e. 11 time series). Best result on NNCI ranking and reduced data was a SMAPE error of 19.0%. Autobox tool [21] based on Box-Jenkins forecasting methodology got an error of 23.9%.

Future works with additional time series will allow us to obtain more accurate conclusions about the effect of using EDA instead of GA. On the other hand, it would be really interesting to use EDA with dependencies between its variables like MIMIC (i.e. variables with order one dependencies) or even "tree" EDA, with no restriction on the numbers of dependencies.

Other interesting future works are: to use "cross validation" into the GA for a better evaluation of each individual; using sparsely connected ANN to try to improve the forecast and getting an accurate system.

**Acknowledgment**

The research reported here has been supported by the Spanish Ministry of Science and Innovation under project TRA2007-67374-C02-02.

**References**

[1] Time Series Forecastig Competition for Neural Networks and Computational Intelligence. http://www.neural-forecasting-competition.com. Web. Update: 18.02.2010

[2] CEDI 2010. Competición de series temporales: http://cedi2005.ugr.es/2010/SICO/competicion html Web. 23 April 2010.

[3] Makridakis, Wheelwright and Hyndman (1998), Forecasting: methods and applications, Wiley: New York.

[4] Zhang, G.; Patuwo, B.E. & Hu, M.Y. Forecasting with artificial neural networks: The state of the art International Journal of Forecasting, 1998, 14,35-62.

[5] S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Edition, Prentice-Hall, 1999.

[6] Crone, S. F. Stepwise Selection of Artificial Neural Networks Models for Time Series Prediction Journal of Intelligent Systems, Department of Management Science Lancaster University Management School Lancaster, United Kingdom, 2005.

[7] Fogel, D. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. Wiley-IEEE Press, 1998.

[8] Ajith Abraham, Meta-Learning Evolutionary Artificial Neural Networks, Neurocomputing Journal, Elsevier Science, Netherlands, Vol. 56c, pp. 1-38, 2004.

[9] X. Yao, "Evolving artificial neural networks," Proceedings of the IEEE, 87(9):1423-1447, September 1999

[10] G. Gutierrez, A. Sanchis, P. Isasi, J.M. Molina and I. M. Galván. Non-direct Encoding method based on Cellular Automata to design Neural Network Architectures. Computer and Informatics, 24 2005, No3.

[11] C. Cotta, E. Alba, R. Sagarna, P. Larrañaga. 2002. Adjusting Weights in Artificial Neural Networks using Evolutionary Algorithms. In Estimation of Distribution Algorithms. A new tool for Evolutionary Computation, P. Larrañaga and J.A. Lozano (eds.). Kluwer Academic Publishers, pp. 357-373.

[12] J. Peralta, G. Gutierrez, A. Sanchis, "ADANN: Automatic Design of Artificial Neural Networks". ARC-FEC 2008 (GECCO 2008). ISBN 978-1-60558-131-6.