Single-Step-Ahead and Multi-Step-Ahead Prediction with Evolutionary Artificial Neural Networks

Víctor Manuel Landassuri-Moreno[1], Carmen L. Bustillo-Hernández[2], José Juan Carbajal-Hernández[2], and Luis P. Sánchez Fernández[2]

[1] Mexico Valley University Center (CUUAEM-VM) – Autonomous University of the State of Mexico, Boulevard Universitario, Predio San Javier, Atizapán de Zaragoza, Estado de México, C.P. 54500, México

[2] Center of Computer Research – National Polytechnic Institute. Av. Juan de Dios Bátiz s/n, Nueva Industrial Vallejo, Gustavo A. Madero, México D.F., C.P. 07738, México

vmlandassurim@uaemex.mx,
{chbustillo004,jcarbajalh,lsanchez}@cic.ipn.mx

Abstract. In recent years, Evolutionary Algorithms (EAs) have been remarkably useful to improve the robustness of Artificial Neural Networks (ANNs). This study introduces an experimental analysis using an EAs aimed to evolve ANNs architectures (the FS-EPNet algorithm) to understand how neural networks are evolved with a steady-state algorithm and compare the Single-step-ahead (SSP) and Multiple-step-ahead (MSP) methods for prediction tasks over two test sets. It was decided to test an inside-set during evolution and an outside-set after the whole evolutionary process has been completed to validate the generalization performance with the same method (SSP or MSP). Thus, the networks may not be correctly evaluated (misleading fitness) if the single SSP is used during evolution (inside-set) and then the MSP at the end of it (outside-set). The results show that the same prediction method should be used in both evaluation sets providing smaller errors on average.

Keywords: evolutionary algorithms, artificial neural networks, EANNs, single-step-ahead prediction, multi-step-ahead prediction.

1 Introduction

Artificial Neural Networks (ANNs) are mathematical models inspired by the structural and functional organization of biological neural networks. They are characterized by having input, hidden and output units with interconnection between them, where each connection has an associated weight which is updated during the training phase to allow the network to learn a given task. Since their origin, they have been used to solve control [1], classification [2, 3] and prediction [4] tasks, showing a performance and adaptability superior to those of conventional mathematical models. Even though neural networks have proved to be a robust method for solving different kinds of problem, they involve several different parameters that need to be chosen appropriately to obtain a functional network. Early studies used to select many of those parameters by trial and error [5]. Another difficulty is that some of these parameters may change over time, and thus more elaborate methods are needed to adjust them. On the other hand, ANNs and Evolutionary Algorithms (EAs) have been widely inspired by biological organisms, usually giving them superior performance when both are applied together to solve a problem than when they are applied in separate stages. Thus, Evolutionary Artificial Neural Networks (EANN), have been remarkably useful at adapting the ANNs' parameters during evolution [2, 6, 7]. This work uses the FS-EPNet algorithm [8], which is based on the EPNet algorithm [3], with the difference that the input Feature Selection is performed, i.e. the FS-EPNet algorithm evolves the inputs of ANNs.

The usage of EAs over ANNs requires an extra error-evaluation (inside-set to test), because during evolution several ANNs are evaluated (fitness assignment). Note that hand design ANNs (HDANNs), usually require one test set to measure the generalization performance. Therefore, there may be different evaluation sets within an EANN: validation set to discover overtraining; inside-set to obtain the fitness of an individual during evolution, and a final test set called outside-set to evaluate the generalization performance after the evolution has finished. In this way, inside and outside terms are used to make reference to performance evaluation, during and after the evolutionary process has been completed. Besides test sets, a prediction method is needed; e.g. the Single-step-ahead (SSP) and Multiple-step-ahead (MSP) prediction methods. It may be worth to remark that those methods have been previously used in econometrics [17]; nevertheless, they have not been used before with EANNs, as in this work.

Thus, this paper is aimed to compare SSP and MSP procedures over both test sets (inside and outside) to forecast two chaotic time series (TS): Lorenz and Mackey-Glass, usually tested in prediction tasks. Thus, the networks (evolved with the FS-EPNet algorithm) may not be correctly evaluated (misleading fitness) if the single SSP is used during evolution (inside-set) and then the MSP at the end of it (outside-set), i.e. both evaluations may be performed in the same terms. Moreover, no previous studies have been found explaining such scenario, which should be tested empirically.

2 FS-EPNet Algorithm

The FS-EPNet algorithm [8] is based upon the standard Evolutionary Programming (EP) approach, aimed at evolving ANN architectures and weights at the same time as obtaining smaller network topologies. The original algorithm (EPNet) [3] does not tackle the feature evolution; i.e. input adaptation in the same evolutionary process. However, further improvements consider their evolution [8]; i.e. Feature Selection EPNet algorithm (FS-EPNet), being the algorithm used during this empirical study. The FS-EPNet algorithm emphasizes the evolution of ANN behaviors by EP, like node-splitting, which maintains the behavioral (i.e. functional) link between the parent and its offspring. It does not have a crossover operator, nor a genotype to represent the individuals. Instead it carries out the evolutionary process by performing only nine different mutation operations directly on the phenotype as shown in Fig. 1: (1) hybrid training composed of training with the Modified Back Propagation (MBP) algorithm and Simulated Annealing (SA); (2) node deletion; (3) connection deletion; (4) input deletion; (5) delay deletion; (6) connection addition; (7) node addition; (8)

input addition; and (9) delay addition. The algorithm performs only one such mutation on the selected individual in each generation. The training in the EPNet algorithm is only a partial training; i.e. the networks are not trained until they converge. This is motivated by computational efficiency, which lets the evolution advance faster, with the individuals improving their fitness through the generations. For a more detailed description of the EPNet algorithm see [3, 8].

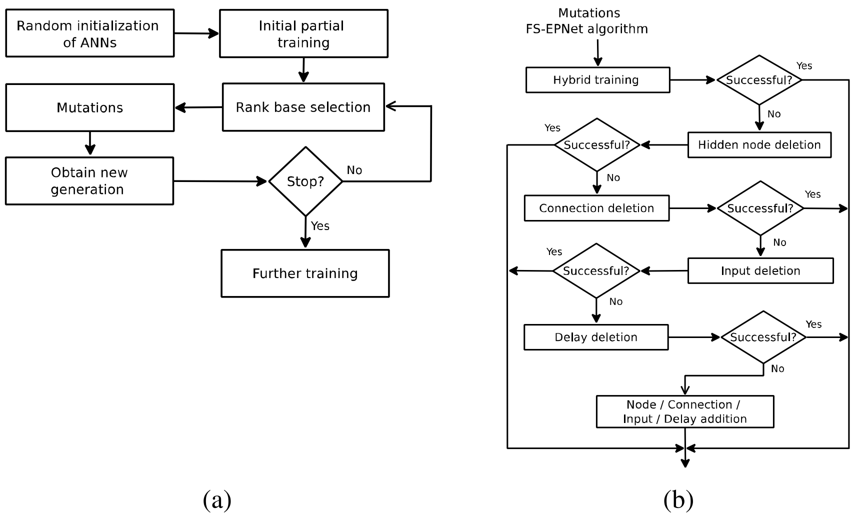

Fig. 1. Feature Selection EPNet algorithm (FS-EPNet); a) general procedure and b) FS-EPNet mutations

3 Time Series Prediction

For the Time Series (TS) prediction problem with ANNs, it is common to try to use a small subset of recent TS information to perform the prediction. Therefore, we are aiming to obtain accurate predictions using only a finite segment of previous values up to the point to be predicted. Thus we have:

$$x_{t+1} = f\left[x_t, x_{t-k}, x_{t-2k}, \ldots, x_{t-(d-1)k}\right] \qquad (1)$$

where d is the number of inputs, k is the time delay and f is the method or algorithm that performs the prediction (the network for this work). There is one condition that needs to be satisfied: given an attractor of dimension D, we must have $d \geq 2D + 1$ [9]. There are two general ways to perform the prediction of TS in terms of the desired number of values to be forecast. Thus, assume the TS X is $[x_1, x_2, \ldots, x_t]$, the number of points ahead to predict is n, the test set is $[x_{t+1}, x_{t+2}, \ldots, x_{t+n}]$, and the forecast in the same interval is $[y_{t+1}, y_{t+2}, \ldots, y_{t+n}]$. In the following examples (Table 1), we are assuming that the number of inputs (past information) is 3, delays are set at 1 and the prediction step is $\Delta t = 1$.

3.1 Single-Step-Ahead Prediction (SSP)

The simplest method is just to predict a value in the future, and we may call this method One-step or Open-loop or Single-step-ahead prediction (SSP). It is called Open-loop forecasting because a pattern is used to predict a value and no feedback is used to continue the predictions as in an autoregressive method. Table 1b shows the single-step prediction method. A sample of previous works that have used (SSP) are [10–13], where [10, 11] predict the Lorenz TS and [12, 13] the Mackey-Glass TS.

3.2 Multi-Step-Ahead Prediction (MSP)

Another interesting prediction method is the Multi-step-ahead prediction (MSP) which uses closed-loop forecasting through an autoregressive method as shown in Table 1a.

Table 1. a) Multiple-step-ahead and b) Single step-ahead prediction methods

a) Forecasting	*inputs*	*b) Forecasting*	*inputs*
y_{t+1}	x_t, x_{t-1}, x_{t-2}	y_{t+1}	x_t, x_{t-1}, x_{t-2}
y_{t+2}	y_{t+1}, x_t, x_{t-1}	y_{t+2}	x_{t+1}, x_t, x_{t-1}
y_{t+3}	y_{t+2}, y_{t+1}, x_t	y_{t+3}	x_{t+2}, x_{t+1}, x_t
y_{t+4}	$y_{t+3}, y_{t+2}, y_{t+1}$	y_{t+4}	$x_{t+3}, x_{t+2}, x_{t+1}$

Note that in Table 1 the predictions are used as input values in subsequent predictions; i.e. it is repeated one-step prediction several times, using the actual prediction to predict the next value. The input vector from the SSP (Table 1b) and MSP (Table 1a) methods may be seen as a window of d values with k delays that is moved one position ahead every time a value is predicted, to be ready to predict the next value. The real difference between both methods is that the SSP moves the window input vector over the original data available, meanwhile the MSP starts with the original data, overlap original and predicted data, and finish with predicted values in the window input vector. Previous publications that used the MSP method are [3, 14, 15], where [14, 15] are focused on the Lorenz TS and [3] predicting the Mackey-Glass TS.

3.3 MSP and SSP Comparison

For the previous section, it can be said that prediction tasks with SSP are similar to classification tasks as one input vector produce one output vector and there is no feedback in the output as in MSP. Having said that, a standard procedure in the literature is to evaluate an inside-set with the SSP method for classification and prediction tasks to obtain the itness of individuals. Moreover, any publication has been found so far that uses MSP over the inside-set, that kind of evaluation is never said and it may be assumed to be SSP as it is the standard.

4 Experimental Set-Up and Data Sets

As previously remarked, it was decided to use an extra test set during evolution as tasks solved with MSP require to test the fitness of individuals with the same method (MSP) when the evolution finish. Thus, the networks may not be correctly evaluated (misleading fitness) if the single SSP is used during evolution and then the MSP at the end of it. For example, it can be assumed that SSP method is easier than MSP by the feedback in the later, therefore, if a prediction task requiring MSP is evaluated with SSP during evolution (inside-set), it will probably produce a different fitness than if the MSP is used in the same inside-set, which could produce a bias in the selection process with networks not so fit. For that reason it was needed to use an extra test set in prediction tasks, so the validation set is used mainly to evolve the learning rate and the inside-set to measure the fitness as it were a real prediction. However, it may be expected to obtain a smaller fitness error during evolution with SSP than MSP. Besides these two sets, the training set was subdivided again (30 patterns) to have a validation set to avoid overtraining (Early Stopping) and provides the maximum generalization performance for the selected architecture. In this way the training may be stopped if over fitting is occurring. As validation set is independent of the task at hand, or the prediction method, it was used the SSP approach to introduce it after each epoch of the training process, as the standard in the literature.

There are some common parameters that were fixed for the experiments throughout this study: population size 30, generations of evolution 3000, initial connection density 30%, initial learning rate 0.15, minimum learning rate 0.01, epochs for learning rate adaptation 5, number of mutated hidden nodes 1, number of mutated connections 1-3, temperatures in SA 5, iterations per temperature in SA 100, 1500 epochs of training inside the FS-EPNet, and 2000 of further training at the end of the algorithm. The only stopping criterion was the number of generations. For all the experiments, 30 independent runs were performed to ensure statistical validity of the results. The inside-set was setup with 30 patterns to perform the prediction and the MSP is performed on the outside-set in all TS tested. Thus, from al data available, the last 100 patterns were taken for the final test set (outside-set) and the next 30 patterns for the inside-set to obtain the fitness of individuals, the rest of the data (from the beginning) was taken for the training set. All these parameters were set at convenient traditional values and are not intended to be optimal. Therefore, it is worth to say that those parameters were set up after some preliminary experiments and they have not been studied thoroughly to say they are the best. Further studies may optimize them.

Two chaotic TS were used to test the insights presented in this study: a) the first one is the Lorenz TS [16] generated with the fourth order Runge-Kutta method as done in [14], i.e. the following values are used to generate the TS: $\Delta t = 1$, $\sigma = 10$, $r = 28$, $\beta = 8/3$ and time step = 0.05 using 1000 values to train and 100 to test (outside-test); and b) the Mackey– Glass TS usually generated with fourth order Runge-Kutta method as Lorenz TS, where the parameters used here to generate it are: $x(0) = 1.2$, $\tau = 17$, $\alpha = 0.2$ and $\beta = -0.1$ as done in [3] using 500 values to train and 500 to test (outside-set). Therefore, the outside-set for Lorenz TS was set to 100 and for Mackey– Glass to 500 patterns, where the last part of the training set was subdivided into the inside-set as commented above. Note that in this work, the same parameters as the literature were replicated to have a fair comparison in the experimental results. Moreover, the FS-EPNet algorithm uses less patterns that in the literature to train the networks (as the inside-set is used), thus a drawback is induced, instead of giving an advantage. The Normalized Root Mean Squared Error (NRMSE) is used to measure the error in the inside and outside sets.

5 Experimental Results

This section presents the results from a set of experiments developed to determine if the usage of SSP in the inside-set may degrade the performance of task requiring MSP on the outside-set. To illustrate this, consider Fig. 2 for the best predictions found for the Lorenz with MSP (Fig. 2a and 2b) and SSP (Fig. 2c and 2d) on the inside-set during evolution and using MSP on the outside-set. Interestingly to note (and as previously expected), the average fitness of the networks evaluated with SSP on the inside-set have a lower error during all the evolutionary process than the fitness obtained from the MSP as can be seen in Fig. 3. The best prediction error on the inside-set at the end of the 3000 generations of evolution were smaller with the SSP than with the MSP as expected too (Table 2, NRMSE Inside-set row). It was also obtained a smaller error with the SSP on the average fitness over all independent trials as shown at the end of generations in Fig. 3. At the end of the evolution, even the network that uses MSP have a bigger fitness error in the inside-set, it obtained the smallest generalization error, with statistical significant having a *p-value* < 0.01 (Fig. 2a), because the selection mechanism during evolution was in the same terms as the generalization measurement.

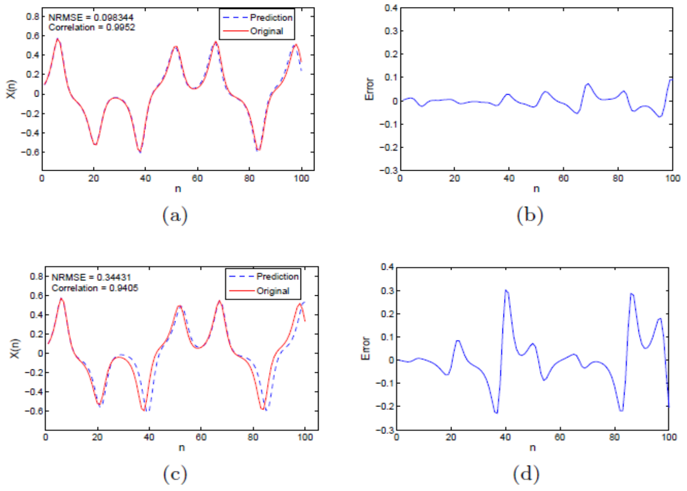

Fig. 2. Best predictions for Lorenz TS after 3000 generations, inside-set with MSP (Fig. 2a and 2b) and inside-set with SSP (Fig. 2c and 2d). Figs. 2b and 2d present the error in terms of $Y_i(t)-Z_i(t)$, where $Y_i(t)$ is the prediction at time t and $Z_i(t)$ is the original data to be predicted (outside-set)

Table 2 presents the individual parameters evolved for the Lorenz TS, showing how the NRMSE over the inside-set is smaller when the SSP is used during evolution than MSP, but the generalization performance is smaller when the MSP is used in both test sets. Also note that using SSP produces the convergence of delays as it is easier to predict with SSP than MSP for the feedback in the latter as previously remarked.

Table 2. Lorenz time series individual results

Parameter	MSP – MSP				SSP – MSP			
	Mean	*Std Dev*	*Min*	*Max*	*Mean*	*Std Dev*	*Min*	*Max*
Number of Inputs	6.70000	1.91455	3	10	6.13333	1.79526	3	9
Number of Delays	2.46666	0.93710	1	4	1	0	1	1
Number of Hidden Nodes	13.3333	4.25346	5	26	12.8	3.28423	6	18
Number of Connections	108.466	56.1989	39	351	92.9333	35.5614	43	172
NRMSE Validation Error	0.02475	0.02848	0.00062	0.12563	0.00145	0.00104	0.00035	0.00447
NRMSE Inside-set	0.01657	0.00775	0.00676	0.03480	0.00189	0.00084	0.00123	0.00477
NRMSE Outside-set	0.52301	0.26162	0.09834	0.92639	0.73256	0.23340	0.34431	1.20320

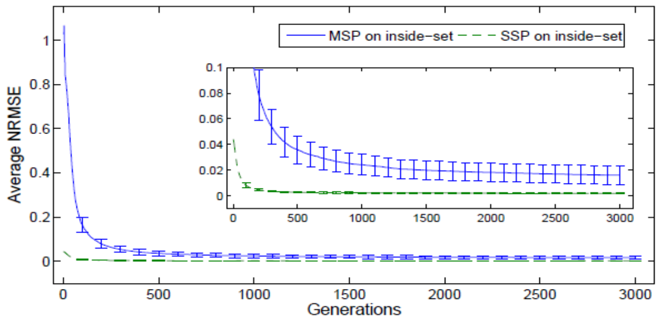

Fig. 3. Average fitness value of Lorenz TS with MSP and SSP over the inside-set

Comparing these results against results found in the literature, Dudul [14] obtain a NRMSE for the best individual with a State-space 8th order of NRMSE = 0.1822 and a NRMSE = 0.7325 with a Regularized ANN-ARMA, while the FS-EPNet obtain a NRMSE = 0.09834 (Table3) for the best individual found. To finalize the results of this work for the Mackey-Glass TS are similar for those presented for the previous case, nevertheless there was no statistically significance in the results. They are no presented here for space reasons.

6 Discussion and Conclusions

This work compares two prediction methods: Single step-ahead (SSP) and Multi-step-ahead, during the evolution of Artificial Neural Networks (ANNs) for time series (TS) prediction. The experiments were carried out using the FS-EPNet algorithm designed to evolve ANNs architectures and weights simultaneously through a steady-state procedure. From two chaotic TS tested (Lorenz and Mackey-Glass), it was determined that tasks that use SSP will use SSP for the fitness during evolution and to evaluate the generalization performance. Contrary, tasks that use MSP will use the same MSP method in both parts of the process. Further research is required to test a broad range of TS to generalize these results.

References

1. Stanley, K., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation 10(2), 99–127 (2002)
2. Bullinaria, J.: Understanding the emergence of modularity in neural systems. Cognitive Science 31(4), 673–695 (2007)
3. Yao, X., Liu, Y.: A new evolutionary system for evolving artificial neural networks. IEEE Transactions on Neural Networks 8(3), 694–713 (1997)
4. Cholewo, T., Zurada, J.: Sequential network construction for time series pre¬diction. International Conference on Neural Networks 4, 2034–2038 (1997)
5. Bishop, M.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
6. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 87(9), 1423–1447 (1999)
7. Bullinaria, A.: Evolving neural networks: Is it really worth the effort? In: Proceedings of the European Symposium on Artificial Neural Networks, Evere, Belgium: d-side, pp. 267–272 (2005)
8. Landassuri, V., Bullinaria, J.: Feature selection in evolved artificial neural networks using the evolutionary algorithm EPNet. In: Proceedings of the 2009 UK Workshop on Computational Intelligence, UKCI '2009. University of Nottingham, Nottingham (2009)
9. Belaire, J., Contreras, D.: Recurrence plots in non-linear time series analysis: Free software. Journal of Statistical Software 7(9) (2002)
10. Rojas, I., Pomares, H., Bernier, J., Ortega, J., Pino, B., Pelayo, F., Prieto, A.: Time series analysis using normalized pg-rbf network with regression weights. Neurocomputing 42(1-4), 267–285 (2002)
11. Gholipour, A., Araabi, B., Lucas, C.: Predicting chaotic time series using neural and neurofuzzy models: A comparative study. Neural Processing Letters 24(3), 217–239 (2006)
12. Müller, K., Smola, A., Rätsch, G., Schökopf, B., Kohlmorgen, J., Vapnik, V.: Using support vector machines for time series prediction, pp. 243–253 (1999)
13. Müller, K.R., Smola, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., Vapnik, V.: Predicting time series with support vector machines. In: Gerstner, W., Hasler, M., Germond, A., Nicoud, J.-D. (eds.) ICANN 1997. LNCS, vol. 1327, pp. 999–1004. Springer, Heidelberg (1997)
14. Dudul, S.: Prediction of a Lorenz chaotic attractor using two-layer perceptron neural network. Applied Soft Computing 5(4), 333–355 (2005)
15. Guerra, F., Dos, S.: Multi-step ahead nonlinear identification of Lorenz's chaotic system using radial basis neural network with learning by clus¬tering and particle swarm optimization. Chaos, Solitons & Fractals 35(5), 967–979 (2008)
16. Lorenz, E.: Deterministic non periodic flow. Journal of Atmospheric Science 20, 130–141 (1963)
17. Hansen, L.P., Hodrick, R.J.: Forward Exchange Rates as Optimal Predictors of Future Spot Rates: An Econometric Analysis. Journal of Political Economy 88(5), 829–853 (1980)