Chaotic Time Series Prediction with Neural Networks - Comparison of Several Architectures

Rafał Mikołajczak [1] and **Jacek Mańdziuk** [2]

Abstract. This paper presents experimental comparison between selected neural architectures for chaotic time series prediction problem. Several feed-forward architectures (Multilayer Perceptrons) are compared with partially recurrent nets (Elman, extended Elman, and Jordan) based on *convergence rate, prediction accuracy, training time requirements and stability of results*.

Results for chaotic logistic map series presented in the paper indicate that prediction accuracy of MLPs with two hidden layers is superior to other tested architectures. Although potential superiority of MLPs needs to be confirmed on other chaotic time series before any general conclusions can be drawn, it is conjectured here that on the contrary to the common beliefs in several cases feed-forward nets may be better suited for short-term prediction task than partially recurrent nets.

It is worth noting that significant improvement in prediction accuracy for all tested networks was achieved by rescaling the data from interval (0,1) to (0.2, 0.8). Moreover, it is experimentally shown that with a proper choice of learning parameters all tested architectures produce stable (repeatable) results.

1 INTRODUCTION

Neural networks are widely used for time series prediction and analysis [7, 8, 4, 1]. Except for the most popular applications to financial (usually stock market related) problems they are also applied to several other nonlinear prediction problems, e.g. weather forecasting [6] or bankruptcy prediction [5, 2].

One of important application areas of neural nets is *chaotic time series prediction* often treated as an indicator of the method's quality before its application to the real (usually financial or business) data. There are two main advantages of such approach: first of all - chaotic series are usually less demanding in terms of complexity and time requirements, and therefore well suited for preliminary tests; secondly - there exist several benchmark chaotic series (logistic map, sunspot series, Mackey-Glass, etc.), which provide a base for comparison between different methods.

The main objective of this paper is experimental comparison between selected feed-forward and partially recurrent neural architectures for chaotic time series prediction based on *logistic map* series. The underlying questions are the following:

- how accurate predictions are possible with particular neural architectures ?
- are partially recurrent nets really better suited for prediction tasks than feed-forward ones ?
- what are the main advantages and disadvantages of both approaches ?

The paper is organized as follows: section 2 presents logistic map series used as experimental evaluation data, section 3 outlines the concept of the experiment (tested architectures, data files and training algorithms), section 4 presents experimental results. Conclusions and possible direction for future work are presented in section 5.

2 LOGISTIC MAP SERIES

Logistic map is one of classical benchmarks for time series prediction. The series is defined by the following recursive equation:

$$x_{k+1} = r x_k (1 - x_k), \qquad x_0 \in (0, 1), \tag{1}$$

where $r \in [1, 4]$ is a predefined parameter. The choice of r determines the "degree of chaos" in eq. (1).

For any given time series the amount of chaos in the data is usually measured by the so-called *Lyapunov Exponent* [9], which is a scalar value (denoted by λ). $\lambda > 0$ characterize chaotic series, whereas $\lambda < 0$ denote non-chaotic systems. The plot of λ versus r for $x_0 = 0.2027$ is presented in Figure 1. From Figure 1 it can be seen that chaotic series are generated for $r_c < r \leq 4$, for some r_c[3]. The highest value of λ - which corresponds to the highest amount of chaos in logistic equation - is obtained for $r = 4$. Following [3] we have used

$$r = 4 \qquad \text{and} \qquad x_0 = 0.2027 \tag{2}$$

in *all tests presented in this paper*[4]. The first 100 values of the series (1)-(2) are presented in Figure 2.

3 EXPERIMENT DESCRIPTION

A set of *a priori* selected architectures, described in more detailed in the next subsection was tested in the experiment. For each architecture suboptimal selection of internal network's parameters (number and size of hidden layers, time window size, self-loop weights in partially recurrent nets) and training coefficients (learning rate, momentum) was made based on preliminary tests. Once these parame-

[1] Indigo. Internet Technologies, e-mail: rafal.mikolajczak@indigo.pl
[2] Corresponding author: Faculty of Mathematics and Information Science, Warsaw University of Technology, Plac Politechniki 1, 00-661 Warsaw, POLAND, e-mail: mandziuk@mini.pw.edu.pl

[3] It can also be seen that $\lambda < 0$ also for some choices of $r_c < r \leq 4$. These "exceptions" are called *r-windows* [9].
[4] Our results are compared with [3] in section 4.2.

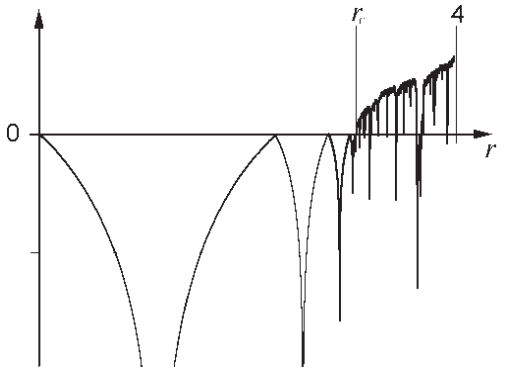

Figure 1. The plot of Lyapunov Exponent versus r for logistic map series with $x_0 = 0.2027$.

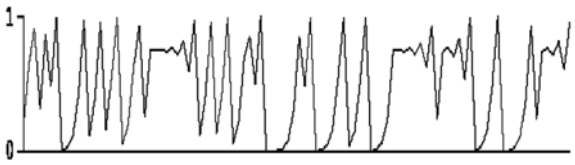

Figure 2. The first 100 values of logistic map series generated for $r = 4$ and $x_0 = 0.2027$.

ters were chosen, they remained fixed during final experiment[5]. In all tested networks sigmoidal neurons were used.

3.1 Tested architectures

Four types of neural architectures were tested:

- **multilayer perceptrons** with one hidden layer ($1-20-1$, $1-30-1$, $1-60-1$, $1-100-1$) and two hidden layers ($1-15-15-1$, $1-22-22-1$, $1-30-30-1$ and $1-60-60-1$),
- **Jordan networks** with one hidden layer, feedback one-to-one connections between output layer and additional state layer and fixed self-loop connections in state layer neurons. The following architectures were tested: $1-10-1$, $1-20-1$, $1-30-1$ and $1-55-1$, each with one element state layer,
- **Elman networks** with one hidden layer and one context layer. Outputs of hidden layer neurons were connected one-to-one with inputs to context layer neurons and each context layer neuron had a fixed self-loop connection. Context neurons were fully connected with hidden layer neurons. The following architectures were tested: $1-10-1$, $1-20-1$ and $1-30-1$ with respectively $10, 20$ and 30 neurons in the context layer,
- **extended Elman networks** with two hidden layers and one-to-one connected with them two context layers (one context layer for the first hidden layer and the other one for the second hidden layer). Each context layer neuron had a fixed self-loop connection. Context neurons were fully connected with respective hidden layer neurons. The following architectures were tested: $1-5-5-1$, $1-10-10-1$ and $1-15-15-1$ each with two context layers of size 5, 10 and 15, respectively.

[5] Except for some tests with partially recurrent nets were adaptive methods were used.

3.2 Data files

Several data files were generated based on logistic equation (1) with parameters (2). Each data file was composed of 1000 subsequent values divided into three subsets: training set (the first 700 points), validation set (next 100 points) and test set (the last 200 points).

All data was rescaled from interval $(0, 1)$ to interval $(0.2, 0.8)$ according to suggestion stated in [11], which was fully confirmed in preliminary simulations (see Figures 3-5).

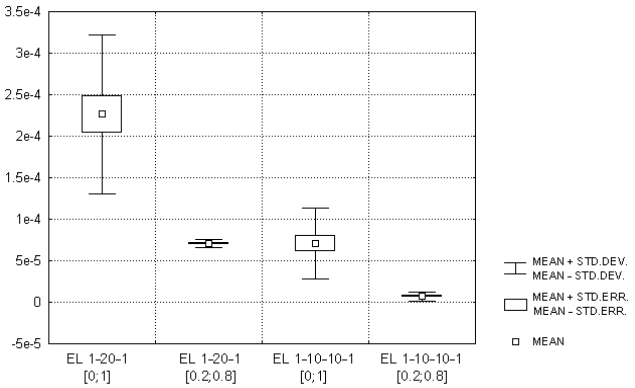

Figure 3. Comparison of preliminary results for the tested Elman-type architectures on the original (0,1) dataset and the one rescaled to (0.2, 0.8).

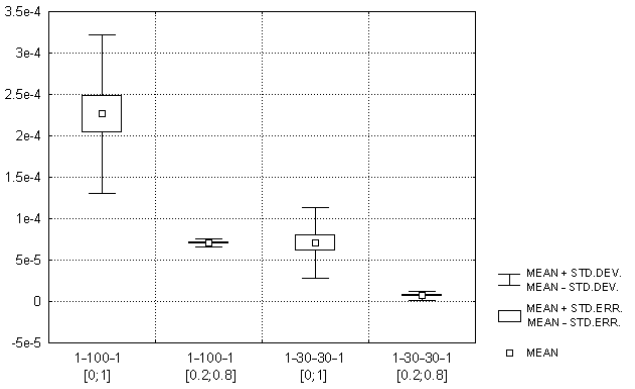

Figure 4. Comparison of preliminary results for the tested feed-forward architectures on the original (0,1) dataset and the one rescaled to (0.2, 0.8).

3.3 Training algorithms

Feed-forward networks were trained with standard *backpropagation with momentum* algorithm and partially recurrent ones with *backpropagation with momentum for partially recurrent nets*. Simulations were performed in the Stuttgart Neural Network Simulator environment [10]. For training feed-forward nets fixed learning rate $\eta = 0.8$ and fixed momentum $\alpha = 0.4$ were used in all final tests. Jordan

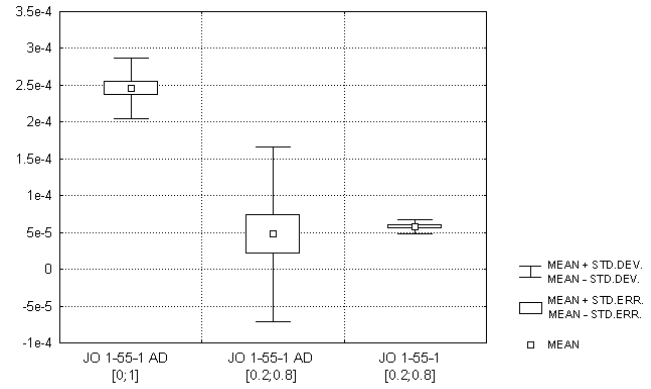

Figure 5. Comparison of preliminary results for the tested Jordan architectures on the original (0,1) dataset and the one rescaled to (0.2, 0.8). In the figure AD denotes the case of adaptively chosen learning rate - as oppose to all other cases where the fixed learning coefficient was applied.

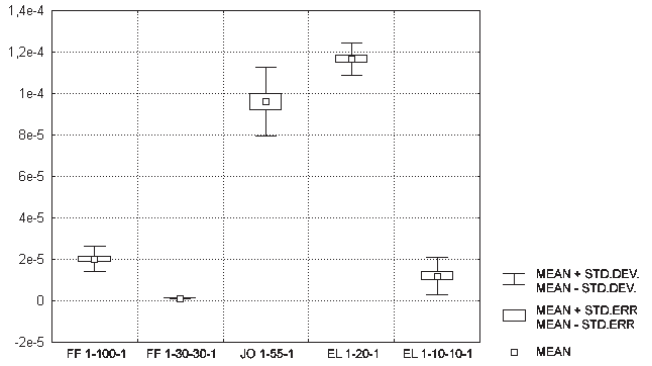

Figure 6. Comparison of the best architectures in each of five types of networks.

nets were trained with $\eta = 0.15$ and $\alpha = 0.05$, and parameters for Elman nets were $\eta = 0.2$ and $\alpha = 0.05$. In each case the choice of η and α was based on initial tests. Due to very unstable results of these tests for extended Elman architectures adaptive methods for chosing learning rate and momentum were used in this case.

Several values for self-loop weights in state layer neurons of Jordan networks and in context layer neurons of Elman-type networks ranging from 0.0 to 0.8 were tested.

In the training procedure validation was made after every 200 learning epochs. Training was stopped either after three subsequent raises of validation error or after $10\ 000$ training epochs.

3.4 Supplementary tests

Some limited number of experiments was also performed for longer term prediction. In these tests the goal was to predict x_{k+t} for $t > 1$ based on x_k. Reasonable results were obtained for $t = 2, 3$, with visible degradation for $t \geq 4$.

4 RESULTS

Each architecture was tested on the same 20 data sets generated from logistic map equation. In some cases the learning procedure became stuck in a very early stage (e.g. after several hundred epochs) and the results were therefore far from satisfactory. These tests were discarded and final result for each network was calculated only among successful tests. The number of unsuccessful tests varied between 0 for MLPs and 5 for Jordan networks with $2 - 3$ for Elman nets.

4.1 Comparison of architectures

Comparison of the best architectures of each type is presented in Figure 6. Closer look at results for all tested architectures in the "winning" type, i.e. feed-forward nets with 2 hidden layers is presented in Figure 7.

On both figures $MEAN$ denotes the mean value of the set of *mean square root errors* generated in individual tests, $STD.DEV$ denotes the *standard deviation* and $STD.ERR$ the *standard error* calculated as $STD.ERR = STD.DEV/\sqrt{(n-1)}$ where n is the number of tests.

Several conclusions can be drawn from Figure 6 and the rest of experimental results not presented here. The main ones are the following:

[1] **The most *accurate predictions* were obtained for feed-forward networks with two hidden layers.** The best prediction was generated for $1 - 30 - 30 - 1$ architecture with the $MEAN$ approximately equal to $1.2 \cdot 10^{-6}$.
When the network was run without the upper limit for the number of training epochs the results were enhanced to $MEAN \approx 3.3 \cdot 10^{-8}$ - after $200\ 000$ epochs.

[2] Very good results, but one order of magnitude worse than for 2 hidden layer feed-forward nets, were obtained for extended Elman architectures with adaptive learning and momentum rates. These results were in turn one order of magnitude better than those for partially recurrent architectures (Jordan and Elman) with 1 hidden layer.

[3] The best result among perceptrons with 1 hidden layer was 1.5 order of magnitude worse than the best result for 2 hidden layer perceptrons. Hence, it is suggested that **networks with 2 hidden layers (both feed-forward and partially recurrent) are more effective in short term prediction task for chaotic time series than their 1 hidden layer counterparts**.

[4] **Partially recurrent nets converged faster than feed-forward ones**, but on the other hand **were generally more sensitive to learning and momentum rate changes**. As mentioned before, in case of extended Elman networks, we were unable to select fixed learning coefficients that generated stable results.

[5] Significant improvement was achieved by **rescaling the data** from interval $(0, 1)$ to $(0.2, 0.8)$.

[6] Results were repeatable with relatively low variances for all architectures.

4.2 Comparison with literature

Experimental results were compared to those presented in [3] where similar experiment was reported. For all tested types of networks our results were superior to those presented in [3].

Seeking for possible explanations of these differences we exactly repeated Hallas and Dorfner's tests and obtained very close results

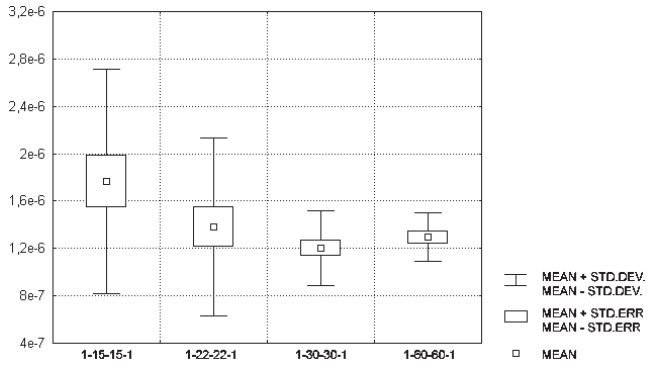

Figure 7. Comparison of 2 hidden layer feed-forward networks tested in the experiment.

to theirs. Further analysis of both experiments led to three aspects, which most probably have caused superiority of our results:

- first, in our experiment learning with momentum was applied,
- second, the data was rescaled to $(0.2, 0.8)$[6],
- third, individual learning and momentum coefficients (or adaptive scheme in case of extended Elman networks) were applied in our tests, whereas in [3] the same coefficients were used for all types of networks.

4.3 Further analysis of results

Certainly, the most surprising conclusion that can be drawn from the above set of experiments is the superiority of feed-forward architectures over the recurrent ones (which "by definition" are supposed to be more appropriate for predicting recurrent time series). Even though a similar qualitative conclusion was reported in [3] it is still interesting to search for a possible explanation of this phenomenon.

One of the potential explanations is the fact that in the recursive logistic equation value $x(t+1)$ at time $t+1$ depends only on the last time step value i.e. $x(t)$. Previous values of the series do not (explicitely) provide any additional information for predicting model. Certainly, due to the recursion mechanism $x(t+1)$ *does implicitely depend* also on previous values: $x(t-1), x(t-2), \ldots, x(0)$, though this dependence is potentially much weaker compared to $x(t)$.

In order to verify the above claim additional tests were performed. First feed-forward networks (MPLs) tested in this work were compared to respective feed-forward networks trained based on the 10-element input (composed of $x(t), x(t-1), \ldots, x(t-9)$). Next, partially recurrent architectures were tested against the choice of self-recurrent connection strengths. The following conclusions from the numerical results support the claim that $x(t+1)$ depends significantly higher on $x(t)$ than on previous elements of the logistic series:

- MPLs with one input performed better than those with ten inputs (the last ten values of the series),
- the best results for partially recurrent networks were obtained when self recurrent loops in state (or context) layers were disabled, i.e. the strenghts of these weights were set to 0.

Another way to estimate the relevance of previous values in the series is to calculate the strenght (the sum of absolute values) of outgoing weights from individual inputs in the multi-input MLP network [7] (e.g. the 10-input network described above composed of $x(t), x(t-1), \ldots, x(t-9)$ values). Such analysis - shown on Figure 8 - additionally confirms the fact that significance of the last value of the series is superior to the others.

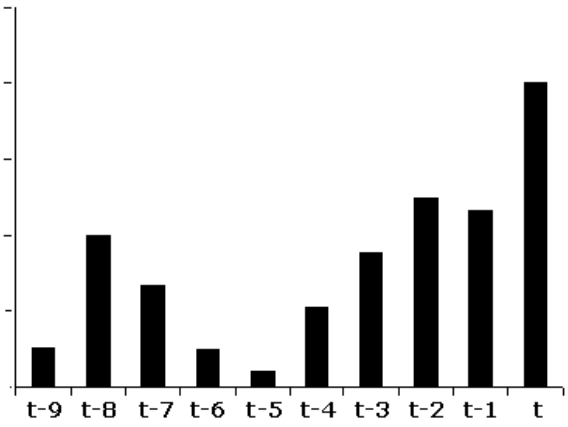

Figure 8. The plot of estimated influence of previous logistic map series values: $x(t-i), i = 9, \ldots, 1$ and $x(t)$ on prediction of the current value: $x(t+1)$ in case of 10-input feed-forward architecture. Bars represent the sums of absolute values of outgoing weights.

5 MAIN CONCLUSIONS AND DIRECTIONS FOR FUTURE WORK

Results presented in this paper indicate that neural nets with 2 hidden layers are superior to those with 1 hidden layer in short-term predictions of logistic map time series. Although potential superiority of MLPs needs to be confirmed on other chaotic time series before any general conclusions can be drawn, it is conjectured here that on the contrary to the common beliefs in several cases feed-forward nets may be better suited for short-term prediction task than partially recurrent nets. It is worth noting that similar qualitative conclusions were also presented in [3], however with generally poorer prediction results. An interesting observation is relatively high stability of results for all architectures.

Several improvements of this work may be considered in future. The first idea is to replace standard backpropagation method by more efficient training algorithms such as Quickprop or Rprop. Next, additional improvement is expected with application of adaptive schemes for learning and momentum rates.

There are also several open problems, e.g. verification of the above results on another chaotic series or application to longer term predictions. Finally, an interesting issue is to compare the above types of neural architectures based on financial or business data. Our current research is focused on that area.

[6] Certainly results were compared after reverse rescaling to interval $(0, 1)$.

[7] Certainly such intuitive approach is acceptable only for rough estimation purposes and under the condition that all inputs are rescaled to the same range (which is true in case of logistic map series).

REFERENCES

[1] D. Brownstone, Using percentage accuracy to measure neural network predictions in Stock Market movements, *Neurocomputing*, **10**, 237-250, (1996).

[2] P.R. Burrell and B.O. Folarin, The impact of neural networks in finance, *Neural Computing & Applications*, **6**, 193-200, (1997).

[3] M. Hallas and G. Dorffner, A comparative study on feedforward and recurrent neural networks in time series prediction using gradient descent learning, in R. Trappl (ed.) *Proc. of the 14th European Meeting on Cybernetics and Systems Research, Austrian Society for Cybernetics Studies*, Vienna, **2**, 644-647, (1998).

[4] K. Kohara, Y. Fukuhara and Y. Nakamura, Selective learning for neural network forecasting of stock markets, *Neural Computing & Appl.*, **4**, 143-148, (1996).

[5] M. Leshno and Y. Spector, Neural network prediction analysis: The bankruptcy case, *Neurocomputing*, **10(2-4)**, 125-147, (1996).

[6] D. Silverman and J. Dracup, Artificial Neural Networks and Long Range Precipitation Prediction in California, *J. of Applied Meteorology*, **31(1)**, 57-66, (2000).

[7] A-P. N. Refenes, A. N. Burgess and Y. Bentz, Neural Networks in Financial Engineering: A Study in Methodology, *IEEE Trans. on Neural Networks*, **8**, 1222-1267, (1997).

[8] E.W. Saad, D.V. Prokhorov and D.C. Wunsch, II, Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks, *IEEE Transactions on Neural Networks*, **9(6)**, 1456-1470, (1998).

[9] Schuster H. G., Deterministic Chaos, VCH Verlagsgesellschaft, 1988

[10] Stuttgart Neural Network Simulator, http://ra.informatik.uni-tuebingen.de/SNNS

[11] Z. Tang and P.A. Fishwick, Feed-forward neural nets as models for time series forecasting, *Technical Report tr91-008*, University of Florida, USA, 1991