Prediction Error Feedback for Time Series Prediction: a way to improve the accuracy of predictions

RYAD ZEMOURI, PAUL CIPRIAN PATIC
Laboratoire d'Automatique du CNAM,
Conservatoire des Arts et Métiers, Valahia University of Targoviste
2 Rue Conté, 75003 Paris, France, 18-22 Unirii Boulevard, Targoviste, Romania
FRANCE, ROMANIA
ryad.zemouri@cnam.fr, patic@valahia.ro

Abstract: - Many applications using recurrent Artificial Neural Network (ANN) for system output prediction do not feedback the previous prediction error to perform the calculation of a new prediction output. The feedback of this prediction error is re-used in two different ways in the neural network. The prediction tests have been carried out on the Mackey–Glass chaotic time series, Logistic Map time series and Box–Jenkins furnace data.

Key-Words: - Static Neural Models, Radial Basis Function, Recurrent Radial Functions Networks

1 Introduction

To consider the temporal aspect of the input data, ANNs require some modifications of the static neural models [4]. We can find two different approaches of time representation in neural network architectures: in the first case, the time is represented as an external mechanism [6]. In the second case, the neural network is able to treat the time dimension without any external mechanism. These Recurrent neural networks are fundamentally different from feed-forward architectures in the sense that they not only operate on an input space but also on an internal state space.

Time series are pervasive in data acquisition, hence a significant amount of work has been done in the realm of time series analysis, modelling and prediction to support analysis and interpretation of such data [1]-[2]-[7]-[9]-[10]-[11]. The commonly encountered models of time series include auto-regressive models [1]-[10], recurrent neural networks [1]-[2]-[5]-[9], and fuzzy rule-based models [2]-[7]-[11]. In such cases it is useful to apply non-linear prediction architectures such as neural networks in order to improve prediction performance [9]. In our previous work, we have introduced the Recurrent Radial Basis Functions networks (RRBF network) [12]. The dynamic aspect is obtained by the use of an additional self-connection on the input neurons with a sigmoid activation function. The RRBF network can thus take into account a certain past of the input signal. To improve time series prediction, we have tested the influence of the prediction error and how we can use it in the RRBF architecture. We have tested two different ways to use this prediction error. One of these two ways gives very interesting results.

This paper is divided into the following sections: sections 2 provides respectively a brief overview of RBF and RRBF network, section 3 describes the data sets, section 4 describes how the experiments have been conducted, section 5 shows the experimental results and Section 6 provides the conclusions and recommendations from this study.

2 RBF Network and the Recurrent RBF

The Radial Basis Function (RBF) consists of two layers (Fig. 1.a) with architecture similar to that of a two-layer MLP. The distance between an input vector and a prototype vector determines the activation of the hidden layer with the nonlinearity provided by the basis function. The nodes in the output layer usually perform an ordinary linear weighted sum of these activations. Mathematically, the network output for linear output nodes is expressed as follows:

$$y_k = \sum_{j=1}^{M} w_{kj}\phi_j\left(\left\|\mathbf{x}-\mathbf{u}_j\right\|\right) \qquad (1)$$

where $\mathbf{x}$ is the input vector with elements x_i (where i is the dimension of the input vector); $\mathbf{u}_j$ is the vector determining the centre of the basis function ϕ_j with elements u_{ji}; w_{kj} are the final layer weights.

The Gaussian basis function $\phi_j(.)$ provides the nonlinearity of the neural network. Training a RBF with linear outputs is very fast and is accomplished through two stages. The first stage is unsupervised and accomplished by obtaining cluster centres of the training set input vectors. A popular method is k-means clustering. The second stage consists in solving a set of linear equations, the solution of which can be obtained by a matrix inversion technique such as singular value decomposition or by least squares.

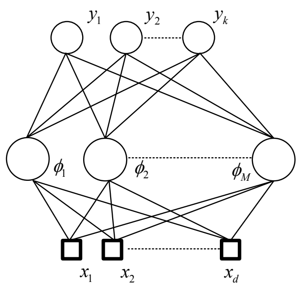

Fig. 1a Different architectures of the RRBF for time series prediction

The Recurrent RBF neural network (Fig. 1.b) considers the time as an internal representation [4], [6].

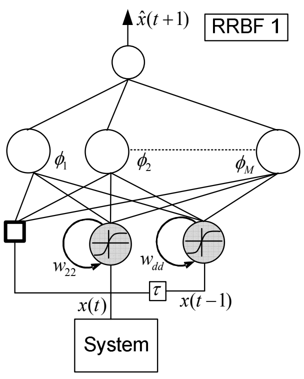

Fig. 2b Different architectures of the RRBF for time series prediction

The dynamic aspect is obtained by the use of an additional self-connection to the input neurons with a sigmoid activation function. The RRBF network can thus take into account a certain past of the input signal. Every neuron of the input layer gives a summation at the instant t between its input x_i and its previous output weighted by a self-connection w_{ii}. The output of its activation function is:

$$a_i(t) = w_{ii}\xi_i(t-1) + x_i(t)$$
$$\xi_i(t) = f\left(a_i(t)\right) \tag{2}$$

where $a_i(t)$ and $\xi_i(t)$ represent respectively the neuron activation and its output at the instant t, f is the sigmoid activation function:

$$f\left(x\right) = \frac{1-\exp(-kx)}{1+\exp(-kx)} \tag{3}$$

For the three benchmarks (described in next section), we have compared three ways to predict the output system $\hat{x}(t+1)$:

a. For the first RRBF (Fig. 1.b) we do not use any prediction error (this first neural configuration is called RRBF1 in the paper).

b. For the second way (RRBF2, Fig. 1.c) we have calculated the prediction error $\varepsilon(t)$ between the output system $x(t)$ and the final predicted output $\hat{x}(t)$. RRBF2 output is then described by these equations (4):

$$\hat{x}'(t+1) = \sum_{j=1}^{M}\left(w_{kj}\phi_j(\left\|\mathbf{x}-\mathbf{u}_j\right\|)\right)$$
$$\hat{x}(t+1) = \hat{x}'(t+1) + \varepsilon(t) \tag{4}$$
$$\varepsilon(t) = x(t) - \hat{x}(t)$$

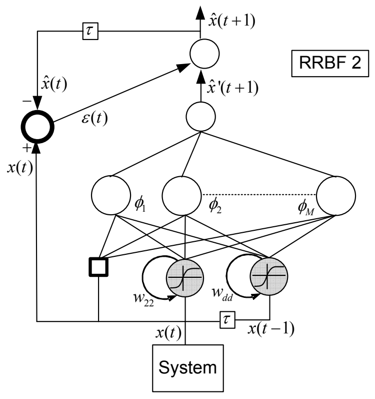

Fig. 3c Different architectures of the RRBF for time series prediction

c. For the third way (RRBF3, Fig. 1.d) one have calculated the prediction error $\eta(t)$ between the output system $x(t)$ and the first stage predicted output $\hat{x}'(t)$. The final predicted output $\hat{x}(t)$ of the RRBF3 is then described by these equations (5):

$$\hat{x}'(t+1) = \sum_{j=1}^{M}\left(w_{kj}\phi_j(\left\|\mathbf{x}-\mathbf{u}_j\right\|)\right)$$
$$\eta(t) = x(t) - \hat{x}'(t) \tag{5}$$
$$\hat{x}(t+1) = \hat{x}'(t+1) + \eta(t)$$

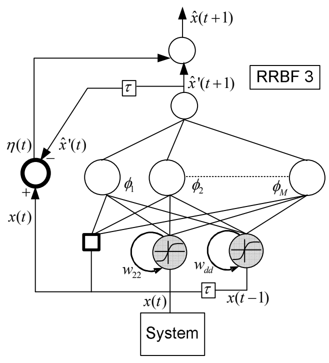

Fig. 4d Different architectures of the RRBF for time series prediction

The main difference between the RRBF2 and the RRBF3 is the way to calculate the feedback error. In the RRBF2, the error $\varepsilon(t)$ is calculated between the output system $x(t)$ and the final predicted output $\hat{x}(t)$, while in the RRBF3, the error $\eta(t)$ is calculated with $\hat{x}'(t)$ (before the final predicted output system).

In the output of the RRBF2, the prediction error $\varepsilon(t)$ at step *t* is propagated at step *t+1, t+2,* …. The following equations present this relation between each time prediction error:

$$\begin{aligned}\varepsilon(t) &= x(t)-\hat{x}(t) = x(t)-\left(\hat{x}'(t)+\varepsilon(t-1)\right)\\ &= x(t)-\hat{x}'(t)-\left(x(t-1)-\hat{x}(t-1)\right)\\ &= x(t)-\hat{x}'(t)-x(t-1)+\hat{x}'(t-1)+\varepsilon(t-2)=....\end{aligned} \tag{6}$$

While in the RRBF3, the prediction error $\eta(t)$ between the output system and the first stage predicted output $\hat{x}'(t)$ is only used to calculate the finale prediction $\hat{x}(t+1)$, and there is any relation between $\eta(t)$ and $\hat{x}(t+2)$ $\hat{x}(t+3)$, …

3 Data sets

The first time series is the Logistic Map. It is defined by the expression $x(t+1)=4x(t)(1-x(t))$. This series is chaotic in the interval of [0, 1], with $x(0)=0.2$. The goal of this application is to predict the target value of $x(t+1)$. The goal of the second time series (the Mackey–Glass differential equation [8]) is to predict the value of time series at some point in the future $x(t+P)$ by using past values. For the purpose of this work, a value of $P=1$, has been used. The third benchmark is Box–Jenkins [3] furnace data. It consists of 296 data points *{y(t), u(t)}*, from *t=1* to 296 where *y(t)* is the output CO2 concentration and *u(t)* is the input gas flow rate. Here we are trying to predict *y(t)* based on: *{y(t-1), u(t-1)}*. Consequently, the effective number of data points is reduced to 290 providing 145 for training and 145 for testing.

4 Comparison tests

In all cases, the error measure used for evaluation is the mean square error (MSE), since it is the most commonly used measure found in literature. All data have been normalized by range [-1, +1]. With all three data sets described earlier, the initial tests attempted to find the best RRBF model when evaluated with the corresponding test sets. In every case, RRBFs have been created with varying numbers of basis functions from 2 to 100 nodes and with varying basis width parameters. While it is appreciated that values used for this scaling variable are extreme, these values have been chosen to encapsulate all possibilities. This study could not be totally exhaustive since the basis width parameter is real. However, increments sufficiently small in width have been used to illustrate trends. For the three RRBFs the basis width parameter started at 0.01 and was increased to a maximum of 1.0 in increments of 0.01.

5 Results

Results in table 1 are the best overall MSEs obtained for the Logistic Map test data using the three RRBFs previously described together with the range of hidden nodes and basis widths discussed in the previous section. For this test, the RRBF1 gives 'the best' result with 100 nodes.

Neural Network	σ	No. nodes	MSE test set
RRBF 1	***0.95***	***100***	***1.2234381e-013***
RRBF 2	0.96	100	3.5824799e-013
RRBF 3	0.99	100	2.0830382e-013

Table 1 Logistic map results

Results in table 2 show the best overall MSEs obtained for Box–Jenkins for various RRBF models produced. The RRBF3 gives the best prediction results. Fig. 5 gives some prediction results of Box–Jenkins benchmark for the three RRBFs network. In this figure we can see that RRBF2 is very unstable. The prediction given by RRBF3 is very close to the output system.

Neural Network	σ	No. nodes	MSE test set
RRBF 1	0.95	25	6.7648253e-003
RRBF 2	0.56	69	2.6361584e-002
RRBF 3	***0.92***	***34***	***3.4003523e-003***

Table 2 Box and Jenkins results

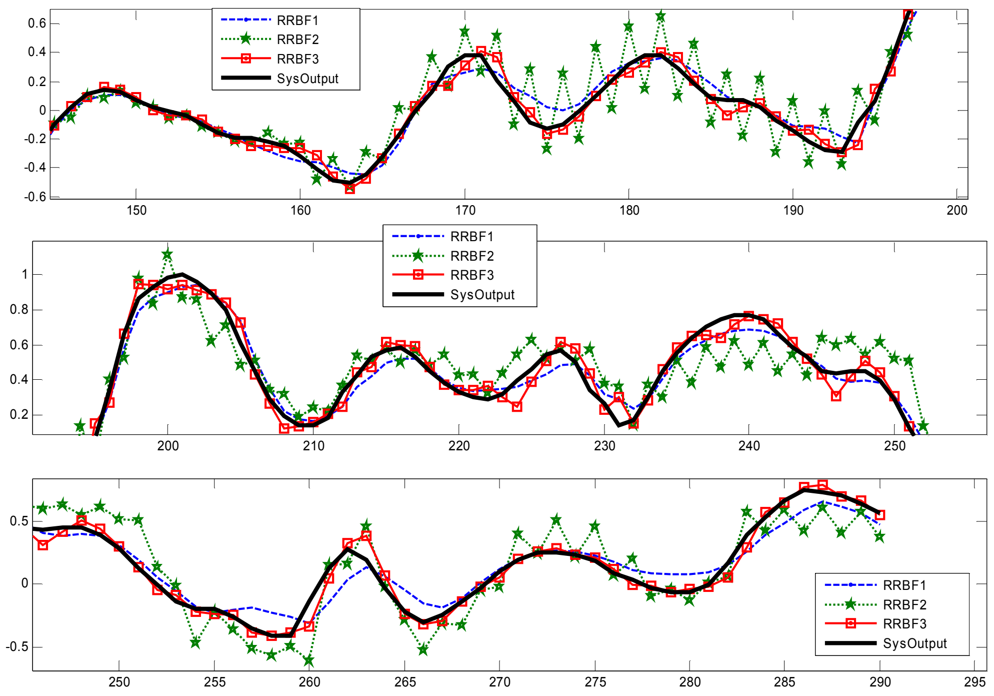

Fig. 5 Box and Jenkins prediction results

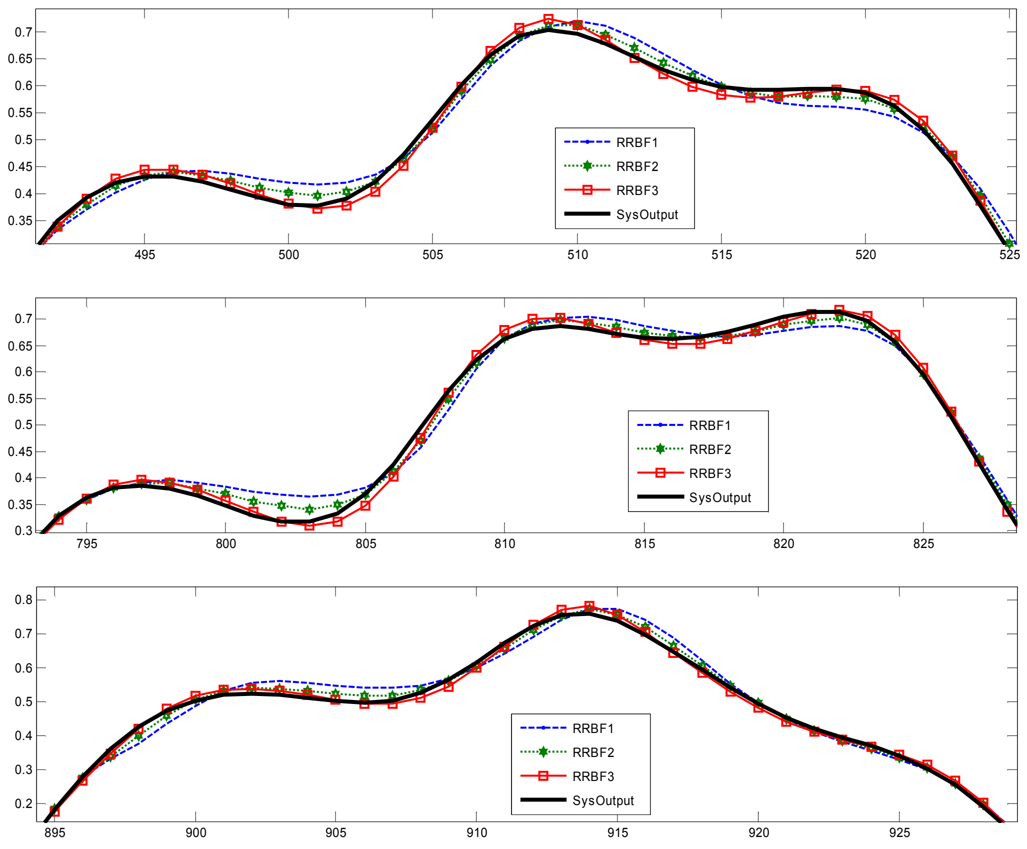

Fig. 6 Mackey Glass prediction result

The results in table 3 are the best overall MSEs obtained for Mackey–Glass test data using the three RRBF previously described together with the range of hidden nodes and basis widths discussed in the previous section. The RRBF3 gives the best result with 61 Gaussian nodes. Fig. 6 gives some prediction results of Mackey–Glass for the three RRBFs network.

Neural Network	σ	No. nodes	MSE test set
RRBF 1	0.38	98	3.1185013e-004
RRBF 2	0.58	97	8.6695360e-005
RRBF 3	***0.75***	***61***	***7.3566953e-005***

Table 3 Mackey-Glass results

6 Conclusion

In this paper the results indicate that the use of the previous prediction error to calculate the future prediction output is very interesting. The results of different tests indicate that the 'best' way to feedback the prediction error is the second way (RRBF3, Fig. 1 a, b, c, d). Additional research should be made to investigate more sophisticated ways to use this feedback error. For example, the linear (or nonlinear) control approaches can be tested as a proportional–integral–derivative controller (PID controller). In this study, only the proportional one has been tested. Other techniques used for control application as fuzzy-controller or neuro-controller can be explored in our future works.

References:

[1] Aliev R.A., B. Fazlollahi, R.R. Aliev, B. Guirimov, *Linguistic time series forecasting using fuzzy recurrent neural networks*, Soft Computing 12(2) (2007) 183-190.

[2] Barbounis T.G., J.B. Theocharis, *A locally recurrent fuzzy neural network with application to wind speed prediction using spatial correlation*, Neurocomputing (2006), doi:10.1016/j.neucom.2006.01.032

[3] Box G.E.P., G.M. Jenkins, *Time Series Analysis Forecasting and Control*, rev. ed., Holden-Day, San Francisco, 1976.

[4] Chappelier J.C., Grumbach A., *« A Kohonen Map for Temporal Sequences »*, Proceeding of neural Networks and Their Application, NEURAP'96, IUSPIM, Marseille, mars 1996, p. 104-110.

[5] Dai Q., S. Chen, *Chained DLS-ICBP neural networks with multiple steps time series prediction*, Neural Processing Letters 21 (2005) 95-107.

[6] Elman J.L., *« Finding Structure in Time »*, Cognitive Science, vol. 14, juin 1990, p. 179-211.

[7] Jacquin A.P., A.Y. Shamseldin, *Development of rainfall-runoff models using Takagi-Sugeno fuzzy inference systems*, Journal of Hydrology 329 (2006) 154-173.

[8] Mackey M.C., L. Glass, *Oscillation and chaos in physiological control systems*, Science 197 (1977) 287–289.

[9] Mandic D.P., J.A. Chambers, *Recurrent Neural Networks for Prediction: Learning Algorithms and Architectures and Stability* (John Wiley & Sons, Chichester, 2001).

[10] Neurmaier A., T. Schneider, *Estimation of parameters and eigenmodes of multivariate autoregressive models*, ACM Transactions on Mathematical Software 27(1) (2001) 27-57.

[11] Vernieuwe H., N.E.C. Verhoest, B. De Baets, R. Hoeben, F.P. De Troch, *Cluster-based fuzzy models for groundwater flow in the unsaturated zone*, Advances in Water Resources 30 (2007) 701-714.

[12] Zemouri R., D. Racoceanu, N. Zerhouni, *Recurrent Radial Basis Function network for Time-Series Prediction, Engineering Applications of Artificial Intelligence*, The International Journal of Intelligent Real-Time Automation, journal IFAC - the International Federation of Automatic Control, Ed. Elsevier Science, vol. 16, Issue 5-6, 2003, pp.453-463.