Article

Not peer-reviewed version

A Training Algorithm for Locally Recurrent Neural Networks based on the Explicit Gradient of the Loss Function

Augusto Montisci * and Sara Carcangiu

Posted Date: 9 January 2025

doi: 10.20944/preprints202501.0702.v1

Keywords: Closed-form error gradient; dynamic systems; gradient-based training; locally recurrent neural networks

Article

A Training Algorithm for Locally Recurrent Neural Networks based on the Explicit Gradient of the Loss Function

Sara Carcangiu and Augusto Montisci *

Electrical and Electronic Eng. Dept.; University of Cagliari; Italy
* Correspondence: augusto.montisci@unica.it;

Abstract: In this paper a new algorithm for the training of Locally Recurrent Neural Networks (LRNNs) is presented, which aims to reduce computational complexity and at the same time to guarantee the stability of the network during the training. The main feature of the proposed algorithm is the capability to represent the gradient of the error in an explicit form. The algorithm builds on the interpretation of the Fibonacci's sequence as the output of an IIR second-order filter, which makes it possible to use the Binet's formula that allows the generic terms of the sequence to be calculated directly. Thanks to this approach, the gradient of the loss function during the training can be explicitly calculated, and it can be expressed in terms of the parameters, which control the stability of the neural network.

Keywords: Closed-form error gradient; dynamic systems; gradient-based training; locally recurrent neural networks

---

1. Introduction

Machine learning techniques, when they are applied to dynamic systems, are preferred to have in their turn a dynamic structure, namely the variable time being included in the algebraic structure of the model. More specifically, in the case of Artificial Neural Networks, the dynamics of the model is obtained by including in the structure delay blocks [1]. The dynamics can be introduced in the neural network by maintaining its feedforward structure (Time Delay Neural Networks - TDNN [2–5]), or by introducing feedbacks [6]. In the latter case, the delays are mandatory, otherwise the calculation of the neurons output cannot be resolved. Both, feedforward and feedback neural networks are suitable for modeling dynamic systems, so that the choice of the paradigm to use resides on the requirements of the problem to deal with and on the available resources. Feedforward paradigm has the advantage of leveraging on the same algorithms for training static NNs. In particular, in the case the delay blocks are foreseen only in a delay line at the input (Focused Time Delay Neural Networks – FTDNN [7]), the downstream part of the NN is structured as a static one, and so any static paradigm can be implemented. A different training strategy is adopted depending on the stationary or non-stationary behavior of the system to be modeled. In the first case the whole evolution of the system assumed as training set can be used to train the NN iteratively in batch mode, in the same way the static NNs are trained. In case the physical system is not stationary, the batch mode is no longer suitable, and the NN model must be adapted dynamically during the evolution of the system [8]. To this purpose, the training set at each iteration is constituted only by the last few samples of the physical signal, while the sensitivity of the model with respect to the past samples tends to vanish with time. From this point of view, the training strategy is the same as adapting filters [9][10], with the added value of exploiting nonlinearity. From a topological point of view, FTDNNs can be seen as the cascade of a Finite Impulsive Response (FIR) linear filter with a Multi-Layer Perceptron [11]. As for the FIRs, this kind of NNs has a short-term memory, represented by the samples stored in the delay line, and a long-term memory, represented by the weights of the

 doi:10.20944/preprints202501.0702.v1

connections. The FIR filters take their name from the fact that the impulsive response has a duration equal to the number of delays in the delay line (Memory Depth), after which it is null. This implies that a proper number of delays must be defined a priori, to guarantee that the dynamics of the system under study will be properly modeled. As such dynamics are unknown a priori, a trial-and-error procedure is adopted to design of the delay line. As an alternative, feedbacks can be introduced in the structure of the NNs [12–14]. Different strategies are used depending on the fact that the feedbacks connect each pair of neurons in the network, or neurons belonging to different layers, or the output of the NN is feedback to the input, or finally if the feedback are localized within the neurons. This last category is called Locally Recurrent Neural Networks (LRNNs [15][16]), and for many applications it represents the best compromise between performance and computational burden. The advantage of feedback is that the memory depth can be arranged by modifying a parameter, rather than changing the topology of the NN. The drawbacks are a larger computational cost with respect to feedforward NNs, and the fact that they are subject to instabilities[4][17]. LRNNs allows one to limit the computational cost, but the stability issue remains. The global structure of these networks is the same of Multi Layer Perceptron (MLP), but internally they are structured as an Infinite Impulsive Response (IIR) linear filter, optionally combined with a FIR filter, while the nonlinearity is placed downstream the linear filter. The IIR filters have a delay line as the FIR filters, but the taps are connected to the input rather than to the output. They take their name from the fact that the duration of the impulsive response is theoretically infinite, even if after a period of time enough long the response is negligible. The main advantage of the IIR filters is that the vanishing time of the impulsive response can be set by changing the feedback parameters and keeping the topology unchanged. Unfortunately, the values of such parameters could make the impulsive response unstable, therefore some measures are needed to prevent this event.

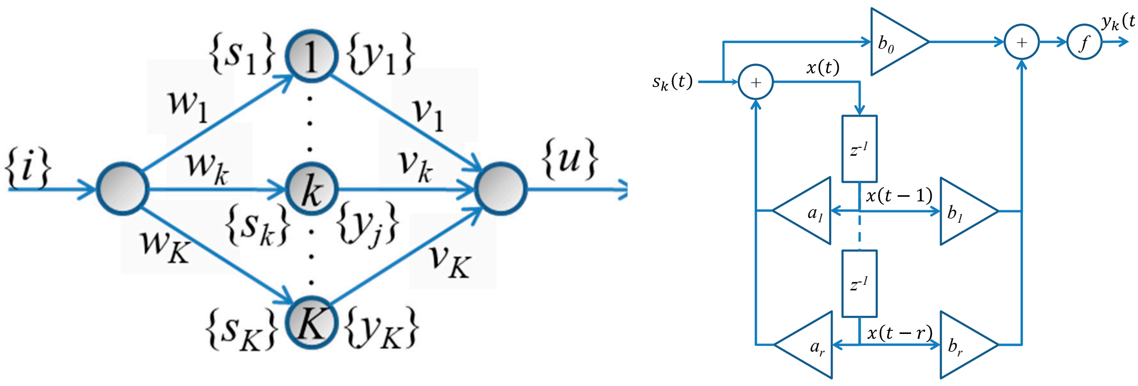
The standard algorithm for the training of feedback NNs is the Backpropagation Trough Time (BPTT [18]), which can be adapted to any feedback topology of NN. This algorithm has been adapted to LRNNs in [17–20], substantially developing the feedback loop a number of times sufficient to consider extinct the impulsive response. Such measure allows one to adapt the structure in spite the feedback structure is non-casual, and it makes it possible to use for training the same procedures defined for feedforward structures. Nonetheless, some issues remain, as the number of times the loop should be developed is unknown a priori, so the assumed value could be too small, giving rise to interference among different impulsive responses, or too large, in this case oversizing the computational cost. Furthermore, the stability of the impulsive response remains a main issue to solve [21].

In the present work, a new training algorithm is presented, which at the same time overcomes the problems of training and stability, in this way allowing one to extend the applicability of the LRNNs. The organization of the paper is as follows. In Section 2 the neural model is presented. In Section 3, the method is applied to the forecasting of a chaotic series. In Section 4 the results are commented and some conclusions are given.

2. Neural Model

The global structure of a LRNN is like that of MLP where neurons are organized in layers, but dynamic properties are achieved using neurons with internal feedback. In Figure 1 the assumed structure of NN is shown. For sake of simplicity and without prejudice to the generality, the NN has a single-input single-output structure, only one hidden layer, where the dynamic element of the network is concentrated, and a linear activation function is assigned to the output neuron. In the rest of the paper, we will refer to this neural structure since such a treatment has the advantage of simplicity and matches the exigencies of the paper. The dynamic part of the network is an ARMA filter, where the parameters a_i create the IIR part and b_i the FIR part. The FIR part is a feedforward structure, for which the literature provides a layout of efficient methodologies, therefore this work will focus only on the IIR part.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2025 doi:10.20944/preprints202501.0702.v1



(**a**) (**b**)

Figure 1. Structure of the LRNN. In (**a**) the global structure is shown, which is identical to that one of MLP. Without loss of generality, a one-input-one-output structure is assumed. The symbols between curl brackets are time sequences, the values without brackets indicate the weights. In (**b**) the internal structure of a hidden neuron is represented. The blocks with z^{-1} constitute the delay line. The battery of b_i gains are the weights of the FIR filter, while the a_i gains are the IIR weights. The gain b_0 represents the a-dynamic connection with the output, while f is the nonlinear activation function of the neuron.

Let us consider the calculation of the state $x(t)$ of the $k-th$ hidden neuron

$$x(t) = \underline{a}^T \cdot \underline{x}(t-1) + s_k(t) \tag{1}$$

where $\underline{a}$ is the vector of feedback gains, $(\cdot)^T$ indicates the transposal operator, $\underline{x}(t-1)$ is the vector state of the delay line, $s_k(t)$ is the current input of the neuron. By referring to Figure 1.b, the output of the neuron is calculated as:

$$y_k(t) = f\left[b_0 \cdot s_k(t) + \underline{b}^T \cdot \underline{x}(t-1)\right] \tag{2}$$

where $\underline{b}$ is the vector of forward gains, b_0 is the a-dynamic weight, and $f(\cdot)$ is the activation function of the neuron. Equation (2) allows to write a dynamic loss function to be used for the training (or adapting) of the NN. Let be $\{d\}$ the desired output sequence of the NN as an answer to the input sequence $\{i\}$. A loss function can be defined as the mean squared error of the output with respect to the desired sequence:

$$J = \frac{1}{2}\sum_{t=1}^{T} [u(t) - d(t)]^2 \tag{3}$$

where T is the duration of the sequence for which the NN must be trained. The simplest procedure to minimize the (3) is based on the gradient of J calculated with respect to all the parameters of the NN. Referring to Figure 1, no difficult occurs for the calculation of the derivatives of J with respect to the global parameters w_k and v_k , with $k = 1, \dots, K$, where K is the number of hidden neurons, as well as to the internal parameters b_j , with $j = 0, \dots, r$, where r is the number of delays. Instead, the derivatives with respect to the internal parameters a_j, with $j = 1, \dots, r$ is troublesome, because the derivative of one sample depends on all the previous ones:

$$\frac{\partial J}{\partial a_{km}} = \sum_{t=1}^{T} [u(t) - d(t)] \cdot v_k \cdot \sum_{j=1}^{r} f'(t-j)\, b_j \frac{\partial x(t-j)}{\partial a_{km}} \tag{4}$$

The last derivative cannot be solved explicitly, as each state $x(t-j)$ due to the feedback depends on the whole previous sequence. To make the (4) explicit, a formula is needed which allows one to calculate the generic term of the impulsive response of the IIR. This is obtained by leveraging on the Binet's formula to calculate the Fibonacci's series, as described in the next sub-section.

2.1. The Fibonacci's Series and the Binet's Formula

The Fibonacci's series (1,1,2,3,5,8, …) is a numeric sequence which owes its popularity to the fact that it reflects a ubiquitous scheme of growth in nature. It is described analytically by the following finite difference equation:

 doi:10.20944/preprints202501.0702.v1

$$x_n = x_{n-1} + x_{n-2} \tag{5}$$

and as can be seen, to calculate the generic term all the previous terms are required. As can be easily demonstrated, the (5) is the expression of the impulsive response of a second-order IIR filter with unitary feedback gains, as the one represented in Figure 2.

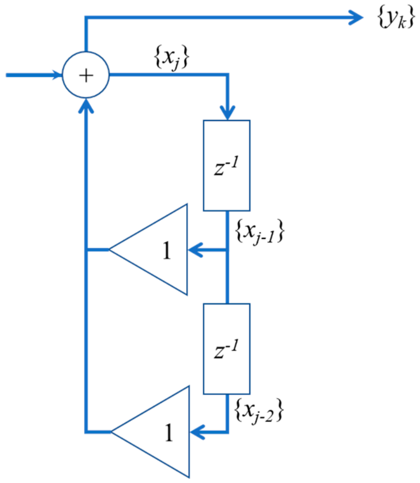


Figure 2. IIR filter which impulsive response corresponds to the Fibonacci's series.

Jacques Philippe Marie Binet (1786-1858) provided a formula which allows to calculate directly any value of the Fibonacci's series without calculating the previous ones. Here the development of that formula is briefly summarized.

Demonstration of the Binet's formula. Let's assume that an explicit function which provides the terms of the Fibonacci's series exists and it has the general expression:

$$h(j) = C \cdot z^j \tag{5}$$

with C and z constant values to be determined. By substituting (5) in (4) the following expression is obtained

$$C \cdot z^j = C \cdot z^{j-1} + C \cdot z^{j-2} \tag{6}$$

and then:

$$C \cdot z^{j-2}(z^2 - z^1 - 1) = 0 \tag{7}$$

Equation (7) has two trivial solutions ($C = 0\,; z = 0$) which must be excluded because they couldn't generate the series, and two non-trivial solutions, namely the roots of the polynomial within brackets. These two solutions are:

$$z_1 = \frac{1+\sqrt{5}}{2} \quad ; \quad z_2 = \frac{1-\sqrt{5}}{2} \tag{8}$$

It is worth noting that the first root z_1 in (8) is the golden ratio value. Let us now assume that the sought function is obtained as a linear combination of the two solutions corresponding to the two roots z_1 and z_2:

$$h(j) = C_1 z_1^j + C_2 z_2^j \tag{9}$$

with C_1 and C_2 to be determined. To this end, we can impose the correspondence with two arbitrary values of the series, for example the first two: 1, 1.

$$\begin{cases} h(1) = C_1 \cdot z_1 + C_2 \cdot z_2 = 1 \\ h(2) = C_1 \cdot z_1^2 + C_2 \cdot z_2^2 = 1 \end{cases} \tag{10}$$

The solution of the system (10) is $C_1 = \frac{1}{\sqrt{5}}$; $C_2 = -\frac{1}{\sqrt{5}}$ from which the following expression of the Binet's formula comes:

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2025 doi:10.20944/preprints202501.0702.v1

$$h(j) = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^j - \left(\frac{1-\sqrt{5}}{2}\right)^j\right] \quad (11)$$

2.2. Exploitation of the Binet's Formula to Calculate the IIR Impulsive Response

The method to determine the Binet's formula can be applied without formal changes to calculate the impulsive response of an IIR filter.

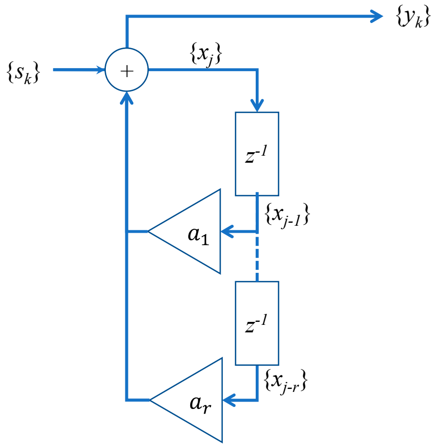


Figure 3. IIR filter with a generic memory depth.

Let be an IIR filter such as the one described in Figure 3, with r delays, which impulsive response writes:

$$x_n = a_1 x_{n-1} + \cdots + a_r x_{n-r} \quad (12)$$

Let's assume that a function exists which provides the arbitrary term of its impulsive response.

$$h(j) = C \cdot z^j \quad (13)$$

By substituting (13) in (12) it results:

$$C \cdot z^j = a_1 C z^{j-1} + \cdots + a_r C z^{j-1} \quad (14)$$

and finally:

$$C \cdot z^{j-r}(z^r - a_1 z - \cdots - a_r) = 0 \quad (15)$$

The non-trivial solutions of (15) are the r roots of the polynomial between brackets. From them, the following general solution can be obtained:

$$h(j) = C_1 z_1^j + \cdots + C_r z_r^j \quad (16)$$

with $C_1, \ldots, C_r$ to be determined. To this end, the first r samples of the impulsive response are calculated and imposed in the following linear equations system:

$$\begin{cases} h(1) = C_1 \cdot z_1 + \cdots + C_r \cdot z_r \\ \vdots \\ h(r) = C_1 \cdot z_1^r + \cdots + C_r \cdot z_r^r \end{cases} \quad (17)$$

The solution of (17) represents the set combination coefficients of the sought function:

$$h(j) = C_1 z_1^j + \cdots + C_r z_r^j \quad (18)$$

2.3. Derivative of the Loss Function with Respect to the Feedback Parameters

The equation (4) describes the derivative of the loss function with respect to the generic feedback parameter. As remarked in the previous sections, such derivatives require the sequence of all the previous samples, therefore an explicit calculation is impossible unless a limit is imposed to the duration of the impulsive response. As said before, the vanishing time of the impulsive response

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2025 doi:10.20944/preprints202501.0702.v1

depends on the feedback parameters we are calculating, so that the assumption is subject to uncertainty. The procedure described in section 2.2 allows one to fix this problem. The IIR filter is a linear system, therefore its answer to a generic input sequence can be expressed as the convolution product between the input signal and the impulsive response:

$$x(t) = (s * h)(t) \tag{19}$$

The equation (19) allows us to calculate the derivatives with respect to the roots z_j rather than the feedback coefficients a_j. This makes it possible to take under control the stability of the NN. In fact, as the state of the neurons depends on the roots z_j, if they are constrained to have the module less than 1, no matter the other parameters, the stability of the network is guaranteed. Therefore, it is convenient to train the network by adapting the zeros of the polynomials, and then calculating the corresponding feedback parameters, which are the coefficients of the polynomials having the z_j as roots. The equation (4) to calculate the derivatives of the loss function with respect to the feedback parameters is substituted by the following one:

$$\frac{\partial J}{\partial z_{km}} = \sum_{t=1}^{T} [u(t) - d(t)] \cdot v_k \cdot f'_k(t) \sum_{j=1}^{r} b_j \left[s_k * \frac{\partial h_k}{\partial z_{km}} \right] (t - j) \tag{20}$$

where $f'(t-j)$ is the derivative of the activation function of the hidden layer. From (18) comes that $h_k(j)$ is a polynomial in z_k and then only one term of its derivative is not null. The derivative finally writes:

$$\frac{\partial J}{\partial z_{km}} = \sum_{t=1}^{T} [u(t) - d(t)] \cdot v_k \cdot w_k \cdot f'_k(t) \cdot C_{km} \sum_{j=1}^{r} b_j \cdot i(t-j) * z_{km}^{t-j-1} \tag{21}$$

The convolution product in (21) implies that a limit must be assumed for the duration of the impulsive response, but thanks to the use of the roots, this term can be established a priori.

2.4. The Training Algorithm

A simple gradient descent method [22] is used to train the LRNN.

$$\Gamma_{p+1} = \Gamma_p - \eta \cdot \nabla J \tag{22}$$

where Γ is the set of all the parameters of the NN, either forward or feedback parameters, p is the iteration index, η is the learning rate. The gradient is calculated with respect to all the independent parameters, considering that both the feedback coefficients a_j and the coefficients of the impulsive responses C_j are univocally determined once the values of the zeros z_j has been established. It is worth noting that the values of the coefficients C_j cannot affect the stability of the NN. The only parameters which affect stability are the zeros z_j, so that the iterative procedure applying the (22) must be meant in constrained terms, in the sense that in updating the set of parameters Γ, the zeros with module greater than 1 must be avoided, by truncating the increment, or by projecting properly the move. The use of more advanced procedures, even possible, is beyond the scope of this study.

3. Results

As a case study, the Willamoski-Rössler reaction [23,24] has been chosen. This benchmark is widely known because it has been the first case which showed that deterministic chaos can be generated by a chemical reaction. The process consists of a multi-step catalytic reaction in the open system, which involves five species, between initiators and products and three intermediates. The following equations describe the five steps of the process:

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2025 doi:10.20944/preprints202501.0702.v1

$$
\begin{aligned}
&A_1 + X \underset{K_{-1}}{\overset{K_1}{\rightleftharpoons}} 2X \\
&X + Y \underset{K_{-2}}{\overset{K_2}{\rightleftharpoons}} 2Y \\
&A_5 + y \underset{K_{-3}}{\overset{K_3}{\rightleftharpoons}} A_2 \\
&X + Z \underset{K_{-4}}{\overset{K_4}{\rightleftharpoons}} A_3 \\
&A_4 + Z \underset{K_{-5}}{\overset{K_5}{\rightleftharpoons}} 2Z
\end{aligned}
\qquad (23)
$$

where A_1, A_4 and A_5 are the initiators, A_2 and A_3 are the products and X, Y and Z are the intermediates. By assuming the 5 step equations the following overall reaction is obtained:

$$A_1 + A_4 + A_5 \Leftrightarrow A_2 + A_3 \qquad (24)$$

In an open system the concentration of both initiators and products is constant in nominal conditions, while the concentrations of the three intermediate species assume a chaotic behavior. By assuming X, Y and Z as state variables, the phase space can be represented graphically. The evolution of the state variables can be studied by means of the following set of differential equations:

$$
\begin{cases}
\dot{X} = K_1 X - K_{-1} X^2 - K_2 XY + K_{-2} Y^2 - K_4 XY + K_{-4} \\
\dot{Y} = K_2 XY - K_{-2} Y^2 - K_3 Y + K_{-3} \\
\dot{Z} = -K_4 XZ + K_5 Z - K_5 Z^2 + K_{-4}
\end{cases}
\qquad (25)
$$

In Figure 3 the evolution in the phase space is shown corresponding to the following reaction rates: $K_1 = 30,\ K_{-1} = 0.25,\ K_2 = 1,\ K_{-2} = 10^{-4},\ K_3 = 10,\ K_{-3} = 10^{-3},\ K_4 = 1,\ K_{-4} = 10^{-3},\ K_5 = 16.5,\ K_{-5} = 0.5$, and initial conditions are: $X_0 = 0.21, Y_0 = 0.01$ and $Z_0 = 0.12$ [24].

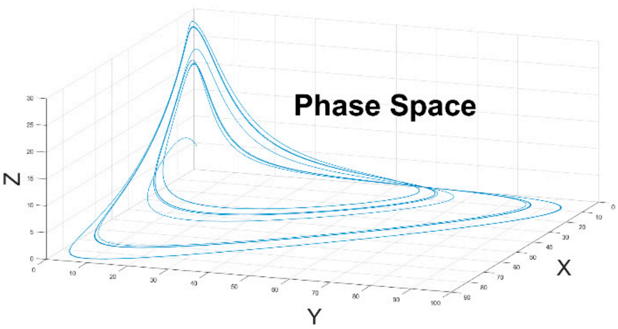


Figure 4. Phase space trajectory of the Willamowski-Rössler model.

In the process chaotic behavior is necessary to obtain an efficient mixing of the reacting species, avoiding spending a great quantity of energy. Therefore, it is important to forecast any variation of the parameters before the chaotic behavior of the system is lost. In Figure 5 the evolution of the state is reported due to a variation of the state variables.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2025 doi:10.20944/preprints202501.0702.v1

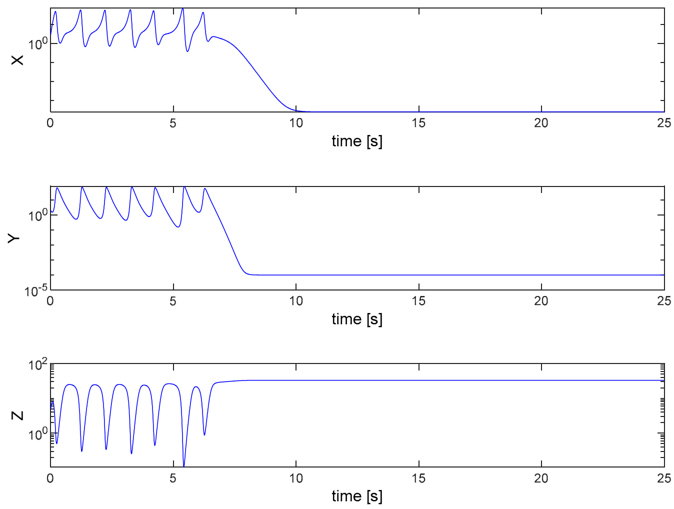


Figure 5. State evolution in presence of a 3% deviation of the reaction rate K_1.

Three LRNNs have been trained, one for each state variable, to predict the value of the same variable one time step ahead. The optimal structure of the network has been determined by means of a trial-and-error procedure, to optimize the forecasting capability. The neural network used to perform the test has a 1-25-1 structure, with only one delay for each hidden neuron and no delay in the output neuron. As an activation function, the hyperbolic tangent function has been assigned to the hidden neurons, while the output neuron is linear. A sequence of 20s of each state variable of the system in nominal conditions has been acquired with a sample time of 5ms, obtaining sequences of 1000 samples, and these sequences have been used to train the neural networks. In Fig. 5 the evolution of Mean Squared Error (MSE) during the training phase is shown for the variable X. As can be seen, the adapted NN is able to perform a good approximation of the chaotic behavior of the system

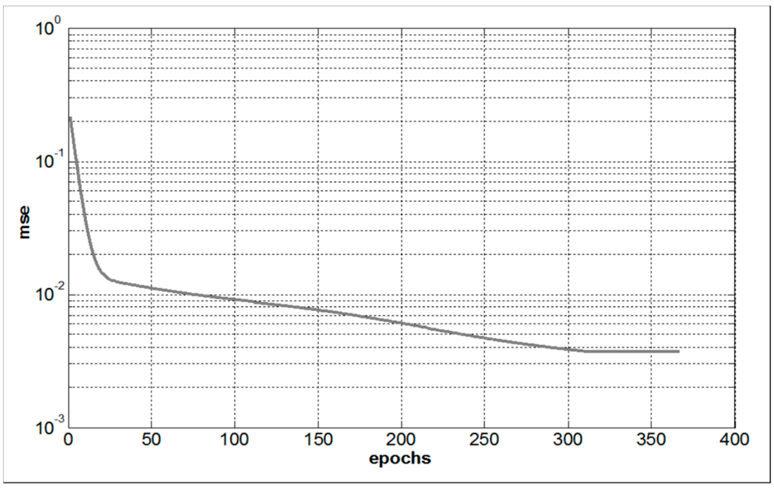


Figure 6. Prediction error of the state variable *X* during the adaptation of the NN.

4. Discussion and Conclusion

The present work aims to introduce a new paradigm for the training of locally recurrent neural networks. The relevance of such neural networks comes from the fact that they conjugate the advantages of feedback with a relatively moderate computational complexity. This is because dynamics is obtained by means of local linear structures, such as FIR and IIR filters. There is wide literature which confirms that this simplification does not affect the potential of the networks with respect to the global feedback, but better performance is achieved because they can be trained with a moderate computational cost. Nonetheless, open issues remain, because of the implicit dependence on past samples, and the stability which cannot be established a priori. The method introduced in this paper provides a solution to both such issues. By extending the formula of Binet for the calculation of the Fibonacci's series terms, a method is presented which allows to calculate directly the terms of the impulsive response of the IIR structure, and then the effect of a change in the parameters of the filter can be estimated a priori. Furthermore, rather than in terms of feedback coefficients, the IIR is expressed in terms of polynomials, which makes it possible to evaluate a priori the effect of a parameter change in terms of stability. The efficiency of the method has been evaluated by developing a predictor of a chaotic behavior of a chemical reactor. The aim of the present paper was to present the theoretical basis of the method and to demonstrate its efficiency in solving non-trivial problems. An extensive comparison with other methods from literature was beyond the scope of the paper, and it will be the subject of future work.

Author Contributions: Conceptualization, A.M; methodology, S.C. and A.M.; software, S.C. and A.M.; validation, S.C.; formal analysis, A.M.; investigation, S.C.; resources, S.C.; data curation, S.C.; writing—original draft preparation, A.M.; writing—review and editing, S.C.; visualization, S.C.; supervision, A.M.; All authors have read and agreed to the published version of the manuscript.

References

1. Han, Y. *et al.* Dynamic Neural Networks: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **44**, 7436–7456 (2022).
2. Waibel, A. Modular Construction of Time-Delay Neural Networks for Speech Recognition. *Neural Comput.* **1**, 39–46 (1989).
3. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E. & Shah, R. Signature Verification using a 'Siamese' Time Delay Neural Network. in *Advances in Neural Information Processing Systems* vol. 6 (Morgan-Kaufmann, 1993).
4. Cao, J. & Wang, J. Global asymptotic and robust stability of recurrent neural networks with time delays. *IEEE Trans. Circuits Syst. Regul. Pap.* **52**, 417–426 (2005).
5. Huang, W., Yan, C., Wang, J. & Wang, W. A time-delay neural network for solving time-dependent shortest path problem. *Neural Netw.* **90**, 21–28 (2017).
6. Nerrand, O., Roussel-Ragot, P., Personnaz, L., Dreyfus, G. & Marcos, S. Neural Networks and Nonlinear Adaptive Filtering: Unifying Concepts and New Algorithms. *Neural Comput.* **5**, 165–199 (1993).
7. Htike, K. K. & Khalifa, O. O. Rainfall forecasting models using focused time-delay neural networks. in *International Conference on Computer and Communication Engineering (ICCCE'10)* 1–6 (2010). doi:10.1109/ICCCE.2010.5556806.
8. Grossberg, S. Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural Netw.* **1**, 17–61 (1988).
9. Haykin, S. S. *Adaptive Filter Theory*. (Pearson Education, 2002).
10. Widrow, B. *et al.* Adaptive noise cancelling: Principles and applications. *Proc. IEEE* **63**, 1692–1716 (1975).
11. Pedro, J. C. & Maas, S. A. A comparative overview of microwave and wireless power-amplifier behavioral modeling approaches. *IEEE Trans. Microw. Theory Tech.* **53**, 1150–1163 (2005).
12. Krotov, D. A new frontier for Hopfield networks. *Nat. Rev. Phys.* **5**, 366–367 (2023).
13. Jordan, M. I. Chapter 25 - Serial Order: A Parallel Distributed Processing Approach. in *Advances in Psychology* (eds. Donahoe, J. W. & Packard Dorsel, V.) vol. 121 471–495 (North-Holland, 1997).
14. Elman, J. L. Finding Structure in Time. *Cogn. Sci.* **14**, 179–211 (1990).
15. Back, A. D. & Tsoi, A. C. FIR and IIR Synapses, a New Neural Network Architecture for Time Series Modeling. *Neural Comput.* **3**, 375–385 (1991).
16. Tsoi, A. C. Recurrent neural network architectures: An overview. in *Adaptive Processing of Sequences and Data Structures: International Summer School on Neural Networks "E.R. Caianiello" Vietri sul Mare, Salerno, Italy September 6–13, 1997 Tutorial Lectures* (eds. Giles, C. L. & Gori, M.) 1–26 (Springer, Berlin, Heidelberg, 1998). doi:10.1007/BFb0053993.
17. Campolucci, P. & Piazza, F. Intrinsic stability-control method for recursive filters and neural networks. *IEEE Trans. Circuits Syst. II Analog Digit. Signal Process.* **47**, 797–802 (2000).
18. Werbos, P. J. Backpropagation through time: what it does and how to do it. *Proc. IEEE* **78**, 1550–1560 (1990).

doi:10.20944/preprints202501.0702.v1

19. Campolucci, P., Uncini, A. & Piazza, F. Causal back propagation through time for locally recurrent neural networks. in *1996 IEEE International Symposium on Circuits and Systems. Circuits and Systems Connecting the World. ISCAS 96* vol. 3 531–534 vol.3 (1996).
20. Campolucci, P., Uncini, A. & Piazza, F. Fast adaptive IIR-MLP neural networks for signal processing applications. in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings* vol. 6 3529–3532 vol. 6 (1996).
21. Campolucci, P. & Piazza, F. Intrinsic stability-control method for recursive filters and neural networks. *IEEE Trans. Circuits Syst. II Analog Digit. Signal Process.* **47**, 797–802 (2000).
22. Battiti, R. First- and Second-Order Methods for Learning: Between Steepest Descent and Newton's Method. *Neural Comput.* **4**, 141–166 (1992).
23. Willamowski, K.-D. & Rössler, O. E. Irregular Oscillations in a Realistic Abstract Quadratic Mass Action System. *Z. Für Naturforschung A* **35**, 317–318 (1980).
24. Niu, H., Wang, H. & Zhang, Q. The Chaos Anti-control in the Willamowski-Rössler Reaction. in *2010 International Workshop on Chaos-Fractal Theories and Applications* 87–91 (2010). doi:10.1109/IWCFTA.2010.60.