Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Revising recurrent neural networks from a granular perspective

Francesco Colace [a], Vincenzo Loia [b,*], Stefania Tomasiello [b]

[a] Dipartimento di Ingegneria Industriale (DIIN), Università degli Studi di Salerno, via Giovanni Paolo II, 132, Fisciano 84084, Italy
[b] Dipartimento di Scienze Aziendali - Management & Innovation Systems (DISA-MIS), Università degli Studi di Salerno, via Giovanni Paolo II, 132, Fisciano 84084, Italy

HIGHLIGHTS

- A new computational recurrent model by data granulation is deduced.
- It is a kind of Echo State Network with an additional granular layer.
- The new scheme is transparent, stable and accurate against state-of-the-art methods.

ARTICLE INFO

Article history:
Received 8 January 2019
Received in revised form 12 March 2019
Accepted 17 April 2019
Available online 3 June 2019

Keywords:
Echo state network
Data granulation
Fuzzy sets
Stability
Deep learning

ABSTRACT

In this paper, we formally deduce a new computational model, with a recurrent structure, by means of data granulation. The proposed scheme can be regarded as an Echo State Network (ESN), with an additional granular layer. ESNs have been recently revisited in the context of deep learning. In view of such a state-of-the-art, and coherently with the concept of data granulation, the aim herein is to propose a more efficient and transparent structure. The stability of the proposed scheme is formally discussed. The performance is shown by means of several benchmarks against the state-of-the-art methods. The proposed architecture exhibits a lower computational cost and a higher accuracy.

1. Introduction

Recurrent neural networks (RNNs) are able to approximate dynamical systems with arbitrary accuracy, offering a mapping between any input and output sequence [1]. They found several applications in different fields; just to mention: for predictive analytics in healthcare [2], for target recognition [3], to predict turbine engine vibration [4], for language modelling [5], for modelling and analysis of squeeze casting process [6], for the energy efficiency problem [7], for the software reliability prediction [8].

In RNNs each hidden state is a function of all previous hidden states and then they can be regarded as inherently deep in time. Actually, they are one of the four typical deep learning (DL) models [9]. Differently from the other DL models (that is auto-encoders, deep belief networks and convolutional neural networks), they allow to take the time series into account.

Being sequential learning models, RNNs may exhibit a complex dynamics and investigating their dynamical properties is crucial. In particular, among them a primary role is played by the convergence for reliable practical applications [10]. Without a proper understanding of the convergence properties of RNNs, many applications would not be possible [11]. The training of a RNNs is usually complex and might be divergent [12]. Several gradient algorithms have been proposed to train RNNs, such as backpropagation through time, real-time recurrent learning, and extended Kalman filtering approaches (e.g. see [13]). This class of algorithms may be affected by the vanishing and exploding gradient problems. In order to address all these issues, an echo-state network (ESN) was proposed as an alternative to RNNs [14–16].

An ESN is characterized by a kernel part, that is a single reservoir consisting of a large number of interconnected neurons (even one thousand order). Unlike traditional RNNs, the connection weights between neurons in the reservoir layer do not require any supervised training. During the ESN training, the reservoir is left unchanged, while only the output weights are computed through least square methods [14–16], with a reduced computational complexity with respect to traditional RNNs. ESNs found several applications, for instance in pattern recognition [17], robot control [18], time series prediction [19,20].

Some recent works have been devoted to the improvement of ESNs. For instance, in [19] a simplified structure for ESN was proposed, by using a single fixed absolute weight value for all reservoir connections and a single weight value for input connections. The authors performed several numerical experiments,

* Corresponding author.
E-mail addresses: fcolace@unisa.it (F. Colace), loia@unisa.it (V. Loia), stomasiello@unisa.it (S. Tomasiello).

supported by some theoretical achievements. In [20], a growing ESN with multiple subreservoirs, built in an incremental way, was proposed to design size and topology of the reservoir layer automatically. The authors showed the performance of the ESN and proved the convergence. In [21], an ESN-like architecture was introduced in the context of DL. Such architecture was substantially an ESN equipped with a pre-training input network. The performance of the network was shown by means of several numerical experiments, but not formally.

Here we are concerned with a discrete model, capturing the granularity of information processed by the network. By starting with the model of a delayed neural network with convolution (see [22]), we deduce a new architecture which may have or not feedbacks from the output layer and which can be regarded as an ESN with an additional layer between the reservoir and the output layer. This additional layer is a granular layer, built by means of fuzzy granules. Data granulation in Neural Networks was introduced to create new computing architectures, that is Granular Neural Networks (GNNs) [23], aimed at achieving a higher degree of transparency in front of a high accuracy.

We discuss formally the properties of the proposed scheme and we present several experiments on widely used benchmarks against the state-of-the-art approaches. From the numerical experiments, we see that the granular layer helps to improve the accuracy, without increasing the reservoir. Large reservoirs have a high computational cost. In comparison, adding an even large granular layer, by keeping small the reservoir, has a lower computational cost. The paper is structured as follows. The next section recalls some basic notions useful to further reading. In Section 3, the new model is deduced. In Section 4, the stability is discussed. Section 5 is devoted to numerical experiments, and finally Section 6 gives some conclusions.

2. Preliminaries

In this section, some basic notions are introduced. Herein we refer to data granulation with fuzzy sets. The principle of justifiable granularity guides the construction of an information granule. It can be roughly summarized, by saying that the information granule should be representative of as many data as possible, though the granule should be quite specific. These two conflicting requirements can be expressed through the following performance index [23]:

$$maximize \sum_k A(z_k)/supp(A), \tag{1}$$

where A is the information granule, which may belong to a certain family of fuzzy sets, and $supp(A)$ its support. The maximization in (1) is realized with respect to the parameters of the information granules, say the bounds of the interval information granule.

Now, we recall all notions related to fuzzy sets and useful to further reading.

Let $I = [\zeta_1, \zeta_m]$ be a closed interval and $\zeta_1, \zeta_2, \dots, \zeta_m$, with $m \geq 3$, be points of I, called nodes, such that $\zeta_1 < \zeta_2 < \cdots < \zeta_m$. A fuzzy partition of I is defined as a sequence $\{A_1, A_2, \dots, A_m\}$ of fuzzy sets $A_i : I \to [0, 1]$, with $i = 1, \dots, m$ such that

- $A_i(\zeta) \neq 0$ if $\zeta \in (\zeta_{i-1}, \zeta_{i+1})$ and $A_i(\zeta_i) = 1$;
- A_i is continuous and has its unique maximum at ζ_i;
- $\sum_{i=1}^{m} A_i(\zeta) = 1, \quad \forall \zeta \in I$.

The fuzzy sets $\{A_1, A_2, \dots, A_m\}$ are called basic functions and they form a uniform fuzzy partition if the nodes are equidistant. The norm of the partition is $h = max_i|\zeta_{i+1} - \zeta_i|$, which simplifies to $h = (\zeta_m - \zeta_1)/(m - 1)$ for uniform partitions, with $\zeta_j = a + (j - 1)h$.

A fuzzy partition can be realized by means of several basic functions; typical basic functions are the triangular ones

$$A_j(\zeta) = \begin{cases} (\zeta_{j+1} - \zeta)/(\zeta_{j+1} - \zeta_j), & \zeta \in [\zeta_j, \zeta_{j+1}] \\ (\zeta - \zeta_{j-1})/(\zeta_j - \zeta_{j-1}), & \zeta \in [\zeta_{j-1}, \zeta_j] \\ 0, & otherwise \end{cases} \tag{2}$$

In order to satisfy (1), in this study we will consider fuzzy partitions with small support.

A fuzzy partition with small support has the additional property that there exists $r \geq 1$ such that $supp(A_i) = \{\zeta \in I : A_i(\zeta) > 0\} \subseteq [\zeta_i, \zeta_{i+r}]$.

Possible basic functions to realize such fuzzy partitions are Bernstein basis polynomials and B-splines (e.g. see [24]). An explicit form for cubic B-splines, for $j = 0, \dots, m$, is given as follows (e.g. see [25])

$$A_j(\zeta) = \frac{1}{4h^3} \begin{cases} (\zeta - \zeta_{j-2})^3, & \zeta \in [\zeta_{j-2}, \zeta_{j-1}) \\ (\zeta - \zeta_{j-2})^3 - 4(\zeta - \zeta_{j-1})^3, & \zeta \in [\zeta_{j-1}, \zeta_j) \\ (\zeta_{j+2} - \zeta)^3 - 4(\zeta_{j+1} - \zeta)^3, & \zeta \in [\zeta_j, \zeta_{j+1}) \\ (\zeta_{j+2} - \zeta)^3, & \zeta \in [\zeta_{j+1}, \zeta_{j+2}) \\ 0, & otherwise \end{cases} \tag{3}$$

It should be pointed out that in order to apply B–splines, some auxiliary points are needed: for cubic B–splines two auxiliary points both on the left and on the right of the considered interval are required.

3. The proposed model

3.1. Description

Here we consider the following N–dimensional model [22]

$$\dot{x}(t) = -Bx + \int_0^t \overline{W}(t - s)y(s)ds + \overline{W}^{IN}u, \tag{4}$$

where $u \in \mathbb{R}^L$ collects L external (input) signals, $y(t) = D\phi(x(t))$, being $\phi(x(t))$ a generic function of $x \in \mathbb{R}^N$ collecting the internal states, $y \in \mathbb{R}^M$ representing M output signals, $D \in \mathbb{R}^{M\times N}$ and $B \in \mathbb{R}^{N\times N}$, $\overline{W}^{IN} \in \mathbb{R}^{N\times L}$, $\overline{W} \in \mathbb{R}^{N\times M}$ at a given time t.

By considering a h-uniform fuzzy partition, then there exists an even function $\bar{A} : [-h, h] \to [0, 1]$ s.t. $A_k(t) = \bar{A}(t - t_k) = \bar{A}(t_k - t)$, with $t \in [t_{k-1}, t_{k+1}]$, for $k = 0, 1, \dots, q$.

So we can write

$$\dot{x}(t) = -Bx + \sum_{k=1}^{q} \overline{W}_k \bar{y}_k + \overline{W}^{IN}u, \tag{5}$$

where

$$\bar{y}_k = \int_0^t A_k(t - s)y(s)ds. \tag{6}$$

By considering the basic functions A_k as hat functions and differentiating (6) using Leibniz' rule, we get

$$\dot{y}_k(t) = c \int_0^t y_0(s)ds + y_0(t), \tag{7}$$

where $c \in \mathbb{R}$, $y_0(t) = y(t)$ for the uniformity of notation, and giving the same kind of contribution so we can omit k.

In discrete form:

$$y(n + 1) = y(n) + \sum_{i=1}^{n} \bar{c}_i y_0(i) + y_0(n), \tag{8}$$

where the coefficients $\bar{c}_i$ come from quadrature rules.

By considering a fuzzy partition over the internal states domain, we can write

$$y_0(i) = DA(i)\overline{\omega}, \tag{9}$$

where $A(i) = A(x(i)) \in R^{N\times m}$ is the matrix whose lr–th entry is $A_{lr}(i) = A_r(x_l(i))$, with $l = 1,\dots,N$, $r = 1,\dots,m$ and ω an m-dimensional vector.

Finally, we get

$$x(n+1) = Wx(n) + W^{BACK}y(n+1) + W^{IN}u(n+1), \tag{10}$$

$$y(n+1) = y(n) + D\sum_{i=1}^{n} A(x(i))\overline{\omega}_i, \tag{11}$$

where $\overline{\omega}_i = \overline{c}_i\omega$ and the matrices $W \in \mathbb{R}^{N\times N}$, $W^{BACK} \in \mathbb{R}^{N\times M}$, $W^{IN} \in \mathbb{R}^{N\times L}$ takes into account the effect of the discretization in (5).

By assuming $y(0) = A(x(0))\overline{\omega}_0$, then we can write

$$y(n+1) = D\sum_{i=0}^{n} A(x(i))\overline{\omega}_i, \tag{12}$$

and the equation above can be arranged as follows

$$y(n+1) = \sum_{i=0}^{n} \Omega_i\overline{V}(i), \tag{13}$$

where

$$\overline{V}(i)^T = (A_1(x_1(i)),\dots,A_m(x_1(i)),\dots,A_1(x_N(i)),\dots,A_m(x_N(i))), \tag{14}$$

and $\Omega_i \in \mathbb{R}^{M\times mN}$, whose kth row is $\{D_{k1}\overline{\omega}_1,\dots,D_{km}\overline{\omega}_1,\dots,D_{k1}\overline{\omega}_N,\dots,D_{km}\overline{\omega}_N\}$.

Assumption 1. We assume $\Omega_n = \overline{\Omega}$ for any n and $\Omega_i = 0$ for every $i \neq n$.

Under Assumption 1, the output (13) is computed as follows

$$y(n+1) = \overline{\Omega V}(n). \tag{15}$$

In general, we can write that the activation of internal units is updated according to

$$x(n+1) = W^{n+1}x(0) + H_{n+1}W^{BACK}y(n+1) + U_{n+1}, \tag{16}$$

where $H_{n+1} = \sum_{i=0}^{n+1} W^i$ and $U_{n+1} = \sum_{i=1}^{n+1} W^{n-i+1}W^{IN}u(i)$.

When the feedback weight matrix $W^{BACK} = 0$, we find a scheme similar to an Echo System Network (ESN) without signal back from the output to the internal units.

3.2. The granular layer and the learning algorithm

In the proposed model, a new (granular) layer between the internal states layer (reservoir) and the output layer is added (Fig. 1). Like the classical model, the input data are processed by the internal states layer, but unlike the usual model, the internal states data are further processed by means of granules before getting to the output layer. In systems with feedbacks, the signal is sent back from the output to the internal units to be processed again. The granular layer is obtained through some fuzzy sets A_l^k, which form a fuzzy partition of the kth internal state domain $\overline{I}_k$, with $k = 1,\dots,N$ and $l = 1,\dots,m$. Hence, for any $\overline{x} \in \overline{I}_k$, there is an associated m–tuple.

A granule is built through a certain number of fuzzy sets and it is usually intended as a fuzzy relation [26], which may be formulated in several ways (e.g. see [27,28]). Here we use the interpretation of the granule as proposed in [29] and which is recalled below.

Let A_l be normal and convex fuzzy sets, for $l = 1,\dots,m$. We assume that for each granule Γ^r there exists a possibility distribution $\phi_\Gamma \in \mathbb{R}^N$, such that

$$\max_{x_j, j=1,\dots,N, j\neq r} \phi_\Gamma(x_1(i), x_2(i),\dots,x_N(i)) = A_l(x_r(i)) \tag{17}$$

Then, for each granule we are endowing it with the approximation:

$$f^l(A_l(x_1(i), x_2(i),\dots,x_N(i))) = \bigvee_{l=1}^{m} A_l(x_k(i)) * \overline{\omega}_l, \tag{18}$$

where $*$ is a t-norm, $\bigvee$ is the maximum operator, $A_l(x_k(i))$ is the membership degree of the data from the kth internal domain and ω_l are the weights, here assumed different for every granule.

From the proposed scheme one may extract Takagi-Sugeno-Kang like rules such as:

R_j^s: IF $x_1(i)$ is A_s AND $x_2(i)$ is A_s AND ... $x_N(i)$ is A_s THEN $y(i+1) = F(f^s(A_s(x_1(i), x_2(i),\dots,x_N(i))))$,

denoting F a suitable functional.

Let us recall (15) now. By collecting the target values into the $\overline{n}\times M$ matrix T, and the vectors $\overline{V}(i)^T$ into the $\overline{n}\times mN$ matrix V, being $\overline{n}$ the maximum discrete time value considered, then we have

$$T = VW^{OUT}, \tag{19}$$

where $T^T = (y(1)^T,\dots,y(n)^T)^T$, $V^T = (\overline{V}(0)^T,\dots,\overline{V}(n-1)^T)^T$ and W^{OUT} the $mN\times M$ matrix to be determined.

In this case, by means of least-squares minimization, we get

$$W^{OUT} = (V^TV)^{-1}V^TT. \tag{20}$$

We wish to point out that the matrix V may have a reduced number of columns, since all the null columns are deleted.

The steps to implement the proposed approach can be easily summarized as follows:

1. choose the type of basic functions and fix the number of granules m;
2. build the matrix V, whose ith row is given by (14);
3. compute $(V^TV)^{-1}$;
4. compute W^{OUT} by (20).

In the case the results are not satisfactory, this procedure is iterated, by increasing m.

4. Stability analysis

Under Assumption 1, we consider the following map

$$x(n+1) = f(x(n)), \tag{21}$$

with $f(x(n)) = Wx(n) + W^{BACK}\overline{\Omega V}(x(n)) + W^{IN}u$.

An equilibrium (or fixed) point $\overline{\mathbf{x}}$ is such that $\mathbf{g}(\overline{\mathbf{x}}) = \overline{\mathbf{x}}$.

An equilibrium point is said to be asymptotically stable if all the eigenvalues of the Jacobian matrix $Jf(\overline{x})$ have moduli less than one.

If a neural system has only asymptotically stable equilibrium points for the given weights and any input u, then it is said to be absolutely stable (e.g. see [30]).

We will prove first the existence of fixed points of (21) and then that (21) is absolutely stable. We recall first some relevant theoretical achievements.

Lemma 2 ([31]). *Let $H^s = [a,b]^s$ be a closed set of $\mathbb{R}^s$ and $f : H^s \to H^s$ be a continuous vector-valued function. Then f has at least one fixed point in H^s.*

In order to prove the existence of fixed points, we shift first the equilibrium point $\overline{\mathbf{x}}$ of (21) to the origin, using the transformation $z = x - \overline{x}$, by getting

$$z(n+1) = Wz(n) + MG(z(n)), \tag{22}$$

with $M = W^{BACK}\overline{\Omega}$ and $G(z(n)) = \overline{V}(z+\overline{x}) - \overline{V}(\overline{x})$. It is clear that (22) has the trivial fixed point.

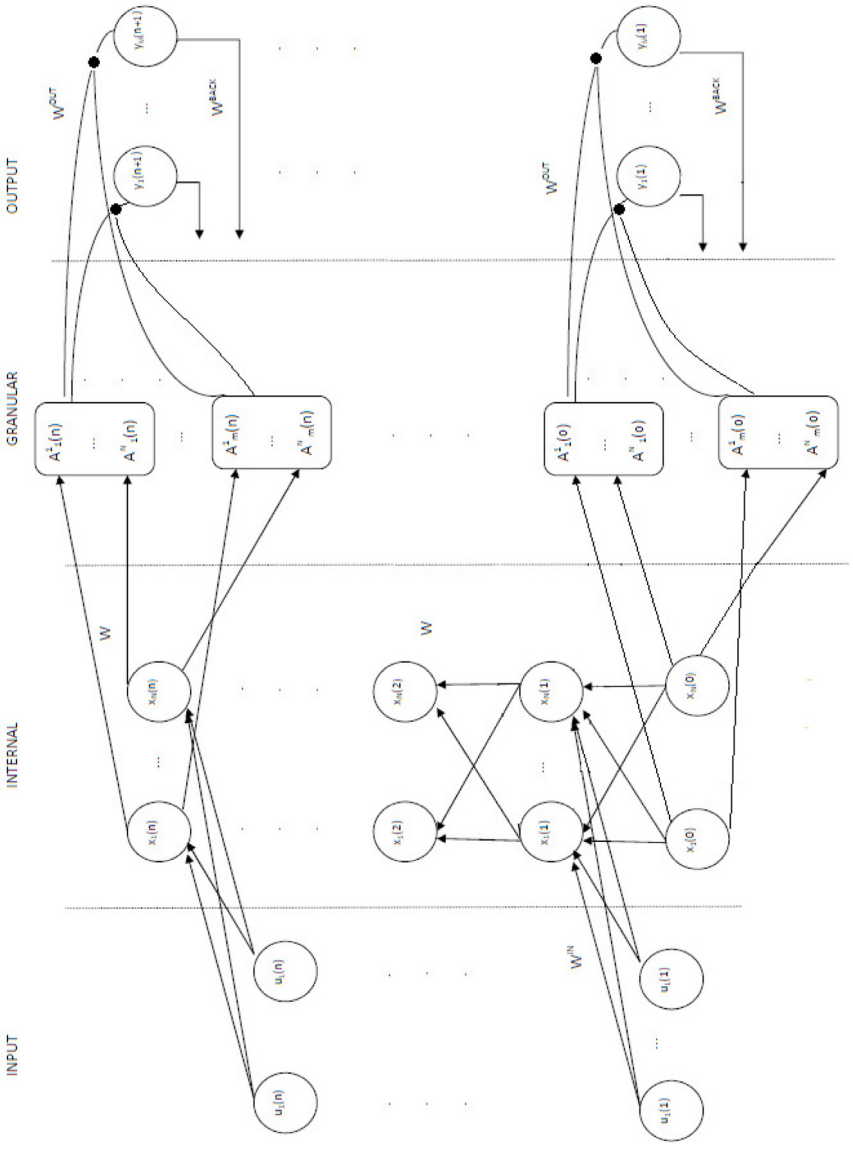

Fig. 1. The proposed model.

Definition 3. A real matrix **M** of size $N \times N$ is *quasi-diagonally (column) dominant* if there exists a positive diagonal matrix $\overline{\mathbf{P}}$ with nonzero entries $\overline{p}_i$, such that

$$\overline{p}_i|M_{ii}| \geq \sum_{\substack{j=1 \\ j\neq i}}^{N} \overline{p}_j|M_{ji}|, \quad \forall i \in \{1, \ldots, N\}. \tag{23}$$

Assumption 4. Let W be a positive diagonal matrix with nonzero entries w_i.

In the following, we refer to the diagonal matrix $P = I_N - W$, being I_N the identity matrix of size $N \times N$.

Theorem 5. *Suppose $w_i < 1$, for $i = 1, \ldots, N$, and M be a quasi-diagonally column dominant matrix. Let $\beta = \max_i |\frac{M_{ii}}{p_i}|$. Then there exists at least one equilibrium point $\overline{z} \in [-\beta, \beta]^s$ of (22).*

Proof. Since we have

$$P\overline{z} = MG(\overline{z}),$$

that is

$$\overline{z}_i = \frac{1}{p_i}\sum_{j=1}^{N} G_j M_{ji} \leq \frac{1}{p_i}\sum_{j=1}^{N} |G_j||M_{ji}| \leq \frac{1}{p_i}|M_{ii}|,$$

since $0 \leq |G_j| \leq 1$, and the conclusion follows. □

From Theorem 5 follows that there exists at least one fixed point $\overline{x}$ of (21) for any input u.

In order to discuss the eigenvalues of the Jacobian $Jf(x)$, we recall the following [32].

Lemma 6. *Let Q be an $N \times N$ complex matrix, $\gamma \in [0, 1]$, and $R_{Q,i}$ and $C_{Q,i}$ denote the deleted row and column sums of Q respectively as follows*

$$R_{Q,i} = \sum_{\substack{j=1 \\ j\neq i}}^{N} |Q_{ij}|, \qquad C_{Q,i} = \sum_{\substack{j=1 \\ j\neq i}}^{N} |Q_{ji}|. \tag{24}$$

All the eigenvalues of Q are then located in the union of N closed discs in the complex plane with centres Q_{ii} and radii $r_i = R_{Q,i}^{\gamma} C_{Q,i}^{1-\gamma}$, $i = 1, \ldots, N$.

Corollary 7. *Let Q be an $N \times N$ complex matrix, $\gamma \in [0, 1]$, and $R_{Q,i}$ and $C_{Q,i}$ be defined by (24). If*

$$|Q_{ii}| + R_{Q,i}^{\gamma} C_{Q,i}^{1-\gamma} < 1, \qquad i = 1, \ldots, N \tag{25}$$

then all the eigenvalues of Q are inside the unit circle in the complex plane.

Theorem 8. *Let δ denote the maximum slope of $\overline{V}$. Suppose $\frac{1-w_i}{\delta} < 1$. If the matrix M satisfies the inequality*

$$|M_{ii}| + R_{M,i}^{\gamma} C_{M,i}^{1-\gamma} < \frac{1 - w_i}{\delta}, \qquad i = 1, \ldots, N, \tag{26}$$

then (21) is absolutely stable.

Proof. Let $\overline{M} = M^T$ and $\overline{V}'$ denote the derivatives vector. The Jacobian is

$$J = W + \overline{V}'^T \overline{M},$$

that is

$$J_{ii} = W_{ii} + M_{ii}\overline{V}_i', \qquad J_{ij} = \overline{M}_{ij}\overline{V}_i'.$$

Hence, we can write

$$\begin{aligned} &|J_{ii}| + \left(\sum_{\substack{j=1 \\ j\neq i}}^{N} |J_{ij}|\right)^{\gamma} \left(\sum_{\substack{j=1 \\ j\neq i}}^{N} |J_{ji}|\right)^{1-\gamma} \\ &\leq w_i + |M_{ii}\overline{V}_i'| + |\overline{V}_i'| \left(\sum_{\substack{j=1 \\ j\neq i}}^{N} |\overline{M}_{ij}|\right)^{\gamma} \left(\sum_{\substack{j=1 \\ j\neq i}}^{N} |\overline{M}_{ji}|\right)^{1-\gamma} \\ &\leq w_i + \delta|M_{ii}| + \delta R_{M,i}^{\gamma} C_{M,i}^{1-\gamma} < 1. \end{aligned}$$

Using Corollary 7, then all the eigenvalues of the Jacobian are inside the unit circle, implying that (21) is absolutely stable. □

5. Numerical experiments

In this section, in order to show the effectiveness of the proposed approach, we considered some benchmark examples: the NARMA system, the Mackey–Glass system, the multiple superimposed oscillator problem, and the video traffic prediction. The first three ones are typical benchmark examples used in a number of papers (see [20,21] and references therein) for time series modelling and identification. For these three examples, data were generated by using suitable equations (see next subsections). The fourth one is an application example in the field of video transmission. In this example, publicly available data were used. More details about all of them will be provided in the related subsections.

In our experiments we used a 5-fold cross validation. The results presented in this section are referred to averaged values, as usual. The highest value of the standard deviation in our experiments is 0.003. As in [20,21], the normalized root-mean-square error (NRMSE) is used as a measure of the prediction accuracy:

$$NRMSE = \sqrt{\frac{\sum_{j=1}^{l_{test}} (y(j) - d(j))^2}{l_{test} * \sigma^2}}, \tag{27}$$

where l_{test} is the length of test samples, $y(j)$ and $d(j)$ are the test output and desired output at time step j, respectively, and σ^2 is the variance of desired output $d(j)$.

In what follows, n denotes a certain time step.

The numerical computations were performed by using a CPU clocking in at 2.40 GHz.

5.1. Example 1: NARMA system

This benchmark case was considered in [20,21], as well as in a number of papers to test the performance of neural networks.

The 10th order NARMA system is written as:

$$\begin{aligned} y(n) = 0.3y(n-1) + 0.05y(n-1) \sum_{i=1}^{10} y(n-1) \\ + 1.5x(n-10)x(n-1) + 0.1, \end{aligned} \tag{28}$$

where $y(n)$ is the system output at time n, and $x(t)$ is the system input at time n, randomly generated by a uniform distribution over the interval [0, 1] and initialized as 0 for $n = 1, 2, \ldots, 10$. By means of (28), 15 000 points were generated. In this case, it is $L = M = 1$. We performed several experiments by varying N, m and the type of basic functions, for both the cases $W^{BACK} = 0$ and $W^{BACK} \neq 0$.

In [21], an architecture with $N = 150$ and a 40-units preprocessing layer was considered, with $W^{BACK} \neq 0$, by finding $NRMSE = 0.0832$ with $\rho = 0.8$ (no information about the training time were provided). In [20], with a final reservoir size $\overline{N} = 618$ and $W^{BACK} = 0$, the authors found $NRMSE = 0.0695$ with a training time of 342.95 s (though they did not give details on the processor used).

For $W^{BACK} = 0$, with $N = 16$, $\rho = 0.76$, $m = 500$, by using hat basic functions, we found $NRMSE = 0.05352$, with a running time on the internal layer of 143 s and on the granular layer 168 s. This result does not change significantly by means of cubic B-splines ($NRMSE = 0.05171$) or sinusoidal shaped basic functions ($NRMSE = 0.05263$). For $m = 400$ the accuracy is slightly worser, that is $NRMSE = 0.0625$, by using again hat functions (as for the case $m = 500$, this result does not change by means of the above mentioned basic functions). A worser accuracy can be also noticed for lesser values of ρ, e.g. for $\rho = 0.4$ and $m = 400$, we found $NRMSE = 0.09464$.

For $W^{BACK} \neq 0$, by keeping $N = 16$, the running times increase, but the accuracy does not improve. In fact, we found for $m = 500$, by means of hat functions, $NRMSE = 0.05675$ in 390.6 s over the internal layer and 189.6 s over the granular layer.

The results in Fig. 2 are referred to $W^{BACK} = 0$ and hat functions. As one can see, for a fixed N, the accuracy improves by increasing m. Instead, by increasing N, for a certain m, one gets no better accuracy, but a higher computational cost. In fact, the running time for $N = 20$ is twice about that for $N = 16$, independently of m. The best results are achieved for $m = 500$, for any ρ, with $NRMSE = 0.07736$ for $\rho = 0.1$ and $NRMSE = 0.05352$ for $\rho = 0.76$.

5.2. Example 2: Mackey–Glass system

This example was taken from [21] and [20]. It is a typical benchmark case for problems as time series modelling or identification. The Mackey–Glass (MG) system is deduced by means of a time-delay differential system as follows

$$\frac{dy(t)}{dt} = \frac{\alpha y(t-\tau)}{1 + y^{\eta}(t-\tau)} + \beta y, \tag{29}$$

where $\eta = 10$, $\alpha = 0.2$, $\beta = -0.1$. For $\tau > 16.8$ the MG system exhibits a chaotic behaviour. Thus $\tau = 17$ is chosen.

In [21], an architecture with $N = 15$ and two 5-units preprocessing layers was considered, with $W^{BACK} \neq 0$, by finding $NRMSE = 6.4 \times 10^{-4}$. In [20], with a final reservoir size equal to 1000, the authors found $NRMSE = 1.32 \times 10^{-4}$, with a training time 51.4 s.

As in [21], we considered 10 000 points. By fixing $N = 15$, $m = 1060$ and $W^{BACK} = 0$, with $\rho = 0.23$, by means of

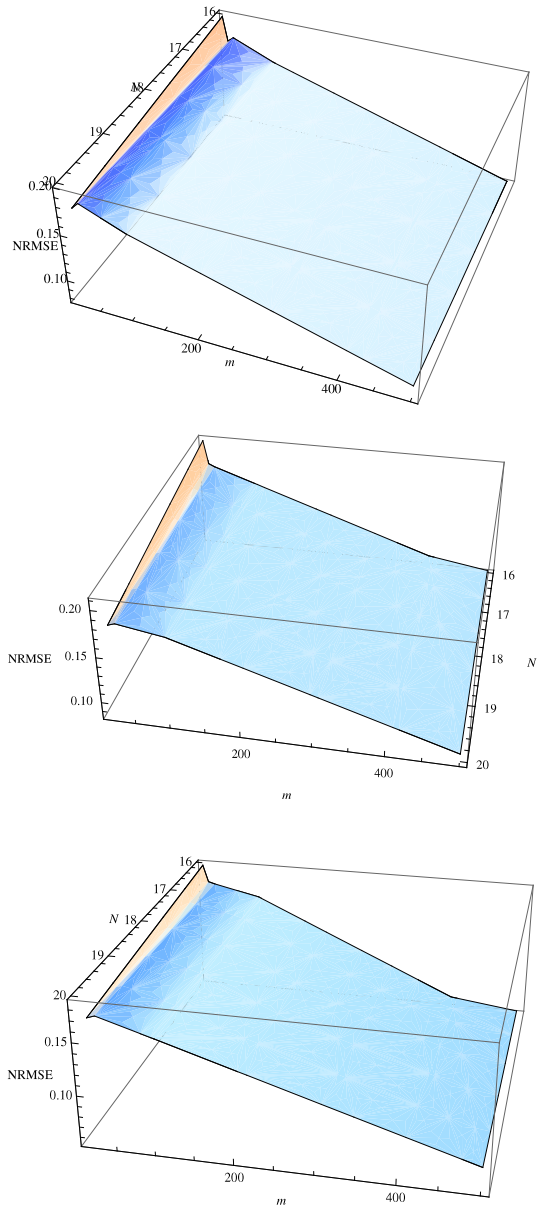

Fig. 2. Example 1: *NRMSE* vs. N, m for different values of ρ (from the top: $\rho = 0.1$, $\rho = 0.4$, $\rho = 0.76$).

hat functions, we found $NRMSE = 0.98 \times 10^{-5}$. By means of cubic B-spline, the accuracy was 30% worse. Similar results by using sinusoidal shaped basic functions. The running time on the granular layer was three times the one on the internal layer. The running times were almost doubled for $W^{BACK} \neq 0$, but without any improvement of the accuracy.

5.3. Example 3: Multiple superimposed oscillator problem

This test example was also taken from [21] and [20]. The dataset was generated by combining several sine wave functions

$$y(n) = \sum_{i=1}^{p} \sin(\alpha_i n), \tag{30}$$

where p and α_i denote the number and the frequencies respectively of sine waves. In this case $L = p$ and $M = 1$.

Table 1
Example 3: *NRMSE* for several values of p.

Approach	$p = 2$	$p = 5$	$p = 8$
Proposed (N = 10)	9.80E−06	6.83E−05	5.6E−05
Proposed (N = 11)	8.9E−06	4.77E−05	5.5E−05
DSCR [21]	3.5E−05	1.63E−04	8.06−E04

Table 2
Example 4: *NRMSE* for different approaches.

Approach	News	Star Wars	Tokyo Olympics	Sony Demo
Proposed (N = 16)	0.2956	0.125	0.1983	0.2177
DSCR [21]	0.4542	0.3824	0.3444	0.3567

In [21], an architecture with $N = 10$ and a 5-units preprocessing layer was considered, with $W^{BACK} \neq 0$, by finding for $p = 2$, $NRMSE = 3.5 \times 10^{-5}$. No information about the training time were provided for this case.

In [20], for $p = 5$ and by means of a final reservoir size equal to 40, the authors got $NRMSE = 5.5 \times 10^{-3}$, with a training time 1.77 s.

We generated 10 000 points. For $W^{BACK} = 0$ and $p = 2$, we found for $N = 11$, $\rho = 0.98$ and $m = 600$, by using cubic B–splines, $NRMSE = 2.15 \times 10^{-5}$. Results by means of the hat functions are one order worse. The running time on the granular layer was twice about that on the internal layer. For $W^{BACK} \neq 0$, with the same matrices and parameters as before, the running times on the internal layer doubled with a worst accuracy, getting $NRMSE = 4.3 \times 10^{-4}$. The results in Table 1 were obtained for $m = 700$, with $W^{BACK} = 0$, by using cubic B–splines. The running times on the internal and granular layers are comparable to those already mentioned for $N = 11$.

5.4. Example 4: VBR video traffic prediction

This example is from [21]. Variable-bit-rate (VBR) video traffic prediction is very important for an efficient resource management and a reliable video transmission. Video traces are used in simulations of transport of video over communication networks and as a basis for video traffic models.

As in [21], we considered VBR video traces from the video trace library of Arizona State University, that is NBC News, Star Wars IV, Tokyo Olympics, and Sony Demo. NBC News is a news show with frequent scene changes, Star Wars IV is derived from the motion pictures, Tokyo Olympics is a documentary on the Olympic games, and Sony Demo is obtained from demo material for a Sony high definition camera. The considered video datasets are equipped with a G16B3 GOP structure of [IBBBPBBBPBBB PBBB], meaning that this structure has 16 frames with three B frames between I and P frames. In [21] an architecture with $N = 35$ and two 15-units preprocessing layers was considered. In our experiments, we considered 3500 samples. Table 2 shows the performance of the approach: results were obtained for $m = 1300$ and cubic B-splines. Fig. 3 shows a sample of the prediction results.

It has been shown that network traffic may exhibit statistical self-similarity. We recall that the Hurst parameter $H \in (0.5, 1)$ is the measure of the self-similarity degree: the larger H, the stronger self-similarity degree. As in [21], we used the well-known R/S method to evaluate the Hurst parameter. Figs. 4–7 show comparative self-similarity plots for the considered datasets. As one can see, the self-similarity characteristic of the original video traffic has not much changed in the prediction results.

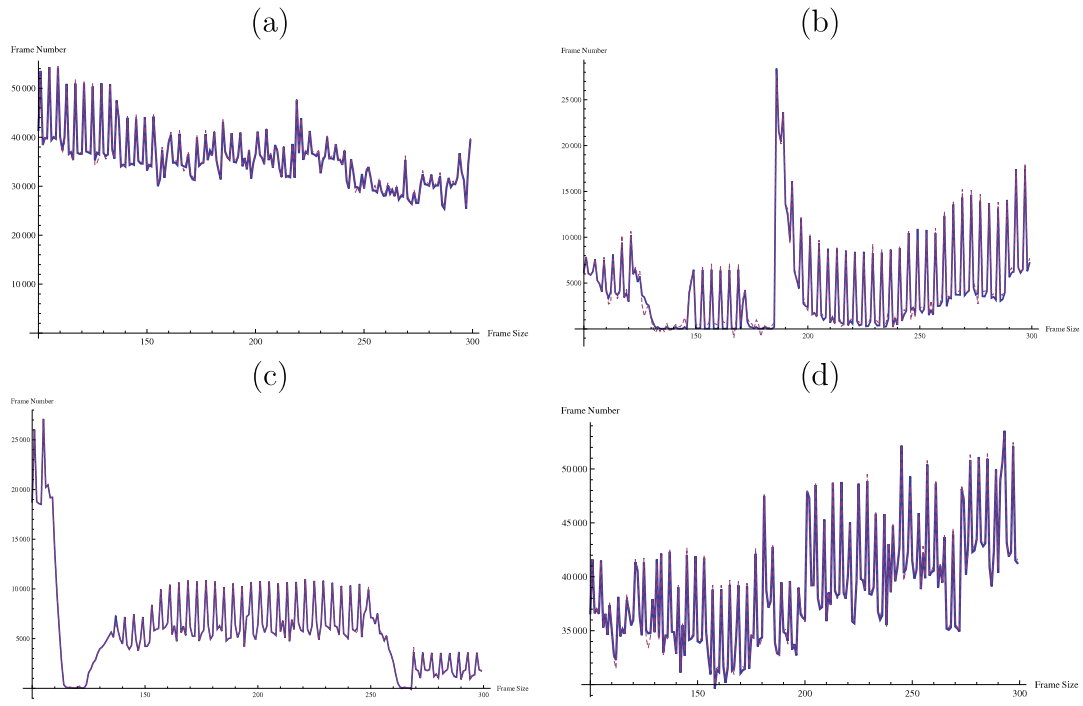

Fig. 3. Example 4: sample output (a) NBC News, (b) Star Wars, (c) Tokyo Olympics, (d)Sony Demo (thick line, original video traffic; dashed line, predicted video traffic).

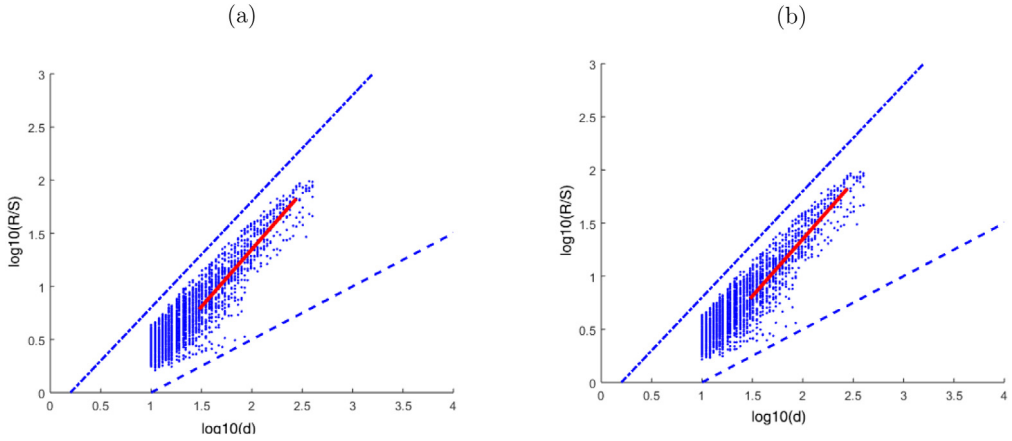

Fig. 4. Example 4: R/S plots on NBC News (a) original, (b) predicted. The slope of the red line is the H value. The upper and lower reference lines correspond to slopes of 1 and 0.5, respectively.

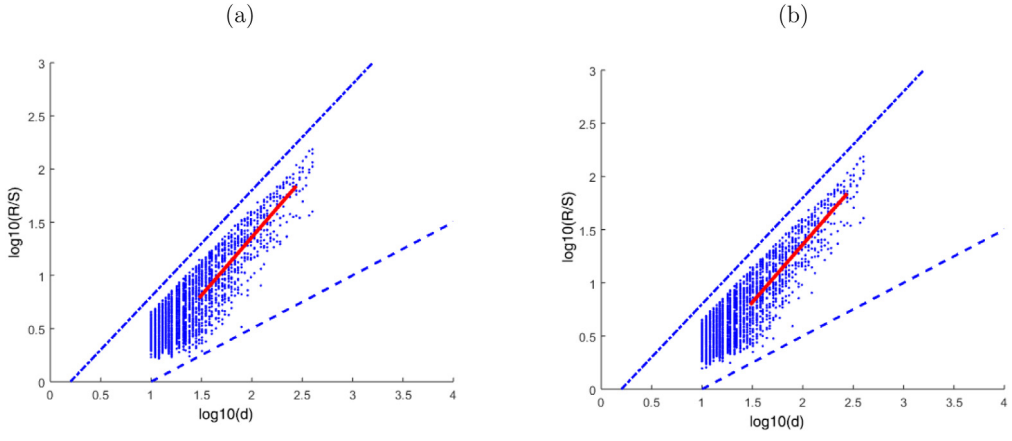

Fig. 5. Example 4: R/S plots on Star Wars (a) original, (b) predicted. The slope of the red line is the H value. The upper and lower reference lines correspond to slopes of 1 and 0.5, respectively.

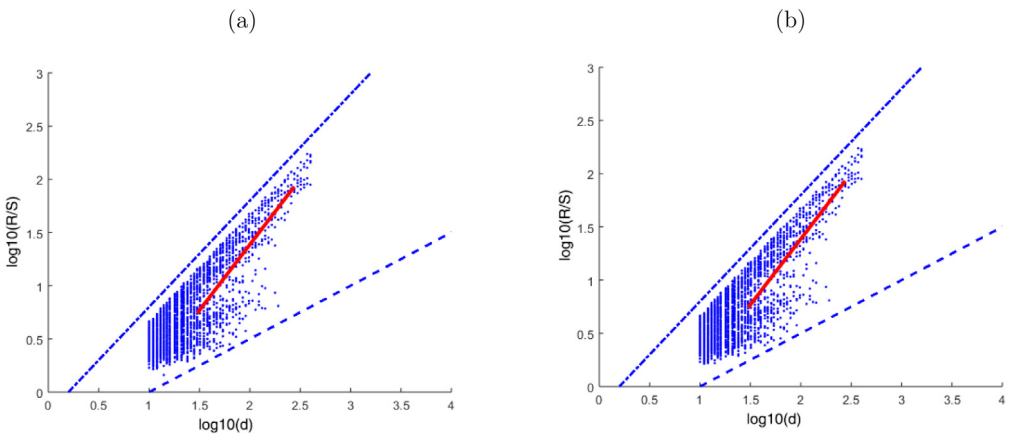

Fig. 6. Example 4: R/S plots on Tokyo Olympics (a) original, (b) predicted. The slope of the red line is the H value. The upper and lower reference lines correspond to slopes of 1 and 0.5, respectively.

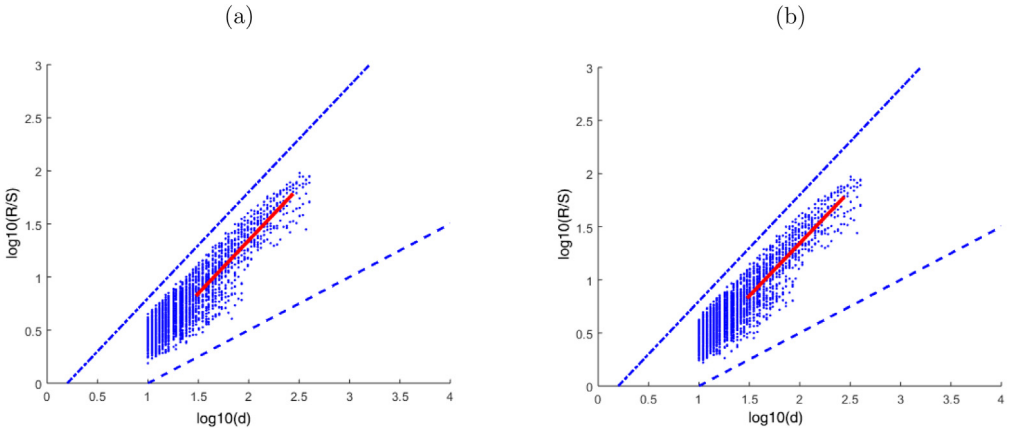

Fig. 7. Example 4: R/S plots on Sony Demo (a) original, (b) predicted. The slope of the red line is the H value. The upper and lower reference lines correspond to slopes of 1 and 0.5, respectively.

6. Conclusions

In this paper, we proposed a revised RNN structure, by exploiting data granulation. We deduced a scheme similar to an ESN, which presents an additional granular layer. Considering granularity has the advantage to achieve a lower computational cost and a higher accuracy. Besides, granularity allows to get more transparent architectures. In this paper, granularity was obtained by means of fuzzy sets. We used some basic functions, such as cubic B-splines, to get fuzzy partitions of the internal states domains. We formally discussed the stability of the new scheme. Such scheme is suitable to time series predictions. We presented a comparative numerical study against the existing methods, to show the performance of the approach. As a future work, the effect of higher-order B splines on the needed granulation to achieve a sufficient accuracy will be investigated. This is motivated by the fact that, by means of the basic functions we used in this paper, it seems that a high number of granules is needed for a good accuracy, even though the computational cost is in any case competitive against the state-of-the-art techniques.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.asoc.2019.105535.

References

[1] A.M. Schafer, H.G. Zimmermann, Recurrent neural networks are universal approximators, Int. J. Neural Syst. 17 (4) (2007) 253–263.
[2] A. Manashty, J. Light, Life model: A novel representation of life-long temporal sequences in health predictive analytics, Future Gener. Comput. Syst. 92 (2019) 141–156.
[3] B. Xu, B. Chen, J. Wan, H. Liu, L. Jin, Target-aware recurrent attentional network for radar HRRP target recognition, Signal Process. 155 (2019) 268–280.
[4] A. ElSaid, F. El Jamiy, J. Higgins, B. Wild, T. Desell, Optimizing long short-term memory recurrent neural networks using ant colony optimization to predict turbine engine vibration, Appl. Soft Comput. 73 (2018) 969–991.
[5] H. Deng, L. Zhang, X. Shu, Feature memory-based deep recurrent neural network for language modeling, Appl. Soft Comput. 68 (2018) 432–446.
[6] M. Patel, A.K. Shettigar, P. Krishna, M.B. Parappagoudar, Back propagation genetic and recurrent neural network applications in modelling and analysis of squeeze casting process, Appl. Soft Comput. 59 (2017) 418–437.
[7] L.G.B. Ruiz, M.I. Capel, M.C. Pegalajar, Parallel memetic algorithm for training recurrent neural networks for the energy efficiency problem, Appl. Soft Comput. 76 (2019) 356–368.
[8] P. Roy, G.S. Mahapatra, Pooja Rani, S.K. Pandey, K.N. Dey, Robust feedforward and recurrent neural network based dynamic weighted combination models for software reliability prediction, Appl. Soft Comput. 22 (2014) 629–637.
[9] Q. Zhang, L.T. Yang, Z. Chen, P. Li, A survey on deep learning for big data, Inform. Fusion 42 (2018) 146–157.
[10] L.-K. Li, S. Shao, Convergence analysis of the weighted state space algorithm for recurrent neural networks, Number. Algebra Control Optim. 4 (3) (2014) 193–207.
[11] Y. Zhang, K.K. Tan, Convergence Analysis of Recurrent Neural Networks, Kluwer, Norwell, MA, 2004.

[12] A.F. Atiya, A.G. Parlos, New results on recurrent network training: Unifying the algorithms and accelerating convergence, IEEE Trans. Neural Netw. 11 (2000) 697–709.
[13] K. Doya, Recurrent networks: Learning algorithms, in: The HandBook of Brain Theory and Neural Networks, MIT Press, Cambridge, MA, USA, 2003, pp. 955–960.
[14] H. Jaeger, The echo state approach to analysing and training recurrent neural networks, German National Research Center Information Technology, St. Augustin, Germany, Tech. Rep. 148, 2001.
[15] H. Jaeger, H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, Science 304 (5667) (2004) 78–80.
[16] H. Jaeger, Adaptive nonlinear system identification with echo state networks, in: Advances in Neural Information Processing Systems, 2002, pp. 593–600.
[17] M.C. Ozturk, J.C. Principe, An associative memory readout for ESNS with applications to dynamical pattern recognition, Neural Netw. 20 (3) (2007) 377–390.
[18] M. Salmen, P.G. Ploger, Echo state networks used for motor control, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005, IEEE, 2005, pp. 1953–1958.
[19] A. Rodan, P. Tino, Minimum complexity echo state network, IEEE Trans. Neural Netw. 22 (1) (2011) 131–144.
[20] J. Qiao, F. Li, H. Han, W. Li, Growing echo–state network with multiple subreservoirs, IEEE Trans. Neural Netw. Learn. Syst. 28 (2) (2017) 391–404.
[21] X. Sun, T. Li, Y. Li, Q. Li, Y. Huang, J. Liu, Recurrent neural system with minimum complexity: A deep learning perspective, Neurocomp. 275 (2018) 1333–1349.
[22] B. de Vries, J.C. Principe, A theory for neural networks with time delays, in: Proceedings: Conference on Advances in Neural Information Processing Systems (NIPS-3), 1990, pp. 162–168.
[23] W. Pedrycz, W. Vukovich, Granular neural networks, Neurocomp. 36 (2001) 205–224.
[24] B. Bede, I.J. Rudas, Approximation properties of fuzzy transforms, Fuzzy Sets and Systems 180 (2011) 20–40.
[25] R.C. Mittal, R. Bhatia, A numerical study of two dimensional hyperbolic telegraph equation by modified b-spline differential quadrature method, Appl. Math. Comput. 244 (2014) 976–997.
[26] W. Pedrycz, G. Vukovich, Abstraction and specialization of information granules, IEEE Trans. Syst. Man Cybern. B 31 (1) (2001) 106–111.
[27] W. Lu, W. Pedrycz, X. Liu, J. Yang, P. Li, The modeling of time series based on fuzzy information granules, Expert Syst. Appl. 41 (2014) 3799–3808.
[28] D. Leite, P. Costa, F. Gomide, Evolving granular neural networks from fuzzy data streams, Neural Netw. 38 (2013) 1–16.
[29] V. Loia, D. Parente, W. Pedrycz, S. Tomasiello, A granular functional network with delay: some dynamical properties and application to the sign prediction in social networks, Neurocomp. 321 (2018) 61–71.
[30] L. Jin, N. Nikifork, M.M. Gupta, Absolute stability conditions for discrete-time neural networks, IEEE Trans. Neural Netw. 5 (1994) 954–964.
[31] R.L. Devaney, An Introduction to Chaotic Dynamical Systems, Addison-Wesley, Reading, MA, 1989.
[32] R.A. Horn, C.A. Johnson, Matrix Analysis, Cambridge University Press, Cambridge, UK, 1985.