Feed-forward Echo State Networks

Michal Čerňanský
Faculty of Informatics and Information Technologies
Slovak University of Technology
Ilkovičova 3, 812 19 Bratislava
E-mail: cernansky@fiit.stuba.sk

Matej Makula
Faculty of Informatics and Information Technologies
Slovak University of Technology
Ilkovičova 3, 812 19 Bratislava
E-mail: makula@fiit.stuba.sk

***Abstract*— New method for modeling nonlinear systems called the echo state networks (ESNs) has been proposed recently [5]. ESNs make use of the dynamics created by huge randomly created layer of recurrent units. Dynamical behavior of untrained recurrent networks was already explained in the literature and models using this behavior were studied [6], [9]. They are based on the fact that the activities of the recurrent layer of the recurrent network randomly initialized with small weights reflect history of the inputs presented to the network. Knowing how the recurrent layer stores the information and understanding the state dynamics of recurrent neural networks we propose modified ESN architecture. The only "true" recurrent connections are backward connection from output to recurrent units and the reservoir is built only by "forwardly" connected recurrent units. We show that this simplified version of the ESNs can also be successful in modeling nonlinear systems.**

I. Introduction

The key part of recurrent neural networks (RNNs) performance is encoded in activities of recurrent units (network state) and their variations in time (network dynamics). It is well-known fact that recurrent neural networks have universal approximation capability, although development of desired dynamics in training might be sometimes difficult or even unfeasible task. On the other hand recent studies show that sometimes instead of complicated RNN weights adaptation, it might be beneficial to leave network dynamics randomly initialized.

It has been known for some time that when RNN is used to process symbolic sequences, activations of recurrent units show considerable amount of information about input sequence prior to training [1], [6]. It was experimentally shown that RNNs initialized with small weights are inherently biased towards Markov models [9], [10]. This phenomenom is refered as Markovian architectural bias of RNNs. When dealing with problems, where this Markovian representation is useful, initial network dynamics can be left unchanged and only transformation of state to desired output has to be carried out. This can be performed either by simple neural network layer or by other advanced method, e.g. by prediction model [7], [8]. One interesting and promising approach based on these principles is the Echo state network (ESN), where untrained "huge" randomly initialized RNN is fed by real-valued input sequence.

Major advantage of this 'unusual' approach is the elimination of the recursive dependencies between weights in adjustment process, i.e. problem when even small weight change in one step can have huge impact on network activities in other steps. Thus, instead of complicated training of the whole network, only the output layer (or the prediction model) is adjusted to produce desired output from the inherent 'Markovian' dynamics.

The next section shortly reviews the ESNs and the idea of echo states is explained. To clarify the dynamics of the ESN, in the third section we recapitulate the notion of the architectural bias property of recurrent neural networks. In the fourth section we propose simple modification to the ESN called "feed-forward" ESN (FF-ESN), where the input sequence history is used explicitly by choosing feed-forward topology of the dynamical reservoir. The concept of the architectural bias or echo states is emphasized by removing recurrent connections and we hope the resulting model will be easier to study and further modifications to meet task-specific requirements will be possible. Preliminary experimental results are shown and conclusions are formulated in the last sections.

II. Echo State Networks

Echo state networks represent a new powerful approach in recurrent neural network research [2], [3], [4]. Instead of difficult learning process, ESNs utilize Markovian architectural bias of untrained RNN to reflect history of seen inputs - here referred to as echo state property. More precisely, the recurrent layer of a large RNN is interpreted as a rich "reservoir" of complex dynamics. Output units are used to extract interesting features from this dynamics, thus only network-output connections are modified during learning process. A significant advantage of this approach is that simple linear regression algorithms can be used for adjusting output weights.

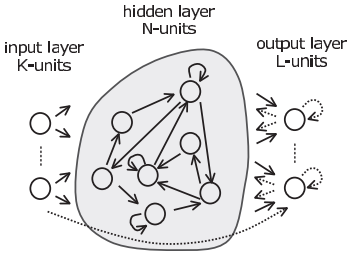

Fig. 1. The simple ESN architecture. Dashed arrows indicate connections that are possible but not required [2].

The network includes input, hidden and output "classical" sigmoid units (Fig. 1). The reservoir of the ESN dynamics is represented by hidden layer with partially connected hidden units. Main and essential condition for successful using of the ESNs is the "echo state" property of their state space. In general, the network state is required to be an "echo" of the input history. If this condition is met, only network output weights adaptation is sufficient to obtain RNN with high performance. However, for large and rich reservoir of dynamics, hundreds of hidden units are needed. When $\mathbf{u}(t+1)$ is an input vector at time step $t+1$, activations of internal units are updated according to

$$\mathbf{x}(t+1) = f\left(\mathbf{W^{in}} \cdot \mathbf{u}(t+1) + \mathbf{W} \cdot \mathbf{x}(t) + \mathbf{W^{back}} \cdot \mathbf{y}(t)\right) \tag{1}$$

where f is the internal unit's activation function, $\mathbf{W}$, $\mathbf{W^{in}}$ and $\mathbf{W^{back}}$ are hidden-hidden, input-hidden, and output-hidden connections' matrices, respectively. Activations of utput units are calculated as

$$\mathbf{y}(t+1) = f\left(\mathbf{W^{out}} \cdot [\mathbf{u}(t+1), \mathbf{x}(t+1), \mathbf{y}(t)]\right) \tag{2}$$

where $\mathbf{W^{out}}$ is hidden-output and output-output connections' matrix.

Echo state property means that for each internal unit x_i there exists an echo function e_i such that the current state can be written as $x_i(t) = e_i(u(t), u(t-1), \ldots)$ [2]. The recent input presented to the network has more influence to the network state than an older input, the input influence gradually fades out. So the same input signal history $u(t), u(t-1),$ will drive the network to the same state $x_i(t)$ in time t regardless the network initial state. To make the idea of the echo states even more clear we recapitulate the concept of the architectural bias in the next section.

III. Architectural Bias

Recurrent units of recurrent networks show considerable amount of structural differentiation [9], [10] before learning. It means, that even in an untrained - randomly initialized recurrent neural network activities of recurrent neurons can be grouped in clusters [6]. This phenomenon can be explained by means of the Iterated Function System theory. IFS theory was originally developed by Barnsley (Fractals Everywhere 1988) as a method of describing the limit behavior of systems of transformations.

An iterated function system is a finite set of contraction transformations

$$\Omega = \{\omega_i | \omega_i : X \longrightarrow X, i \leq n\} \tag{3}$$

Limit behavior of a single transformation is a single point in the space. Limit set over the union of transformations can be extremely complex with recursive structures. This limit behavior of composite mapping is called the IFS attractor. An example of an IFS is set of these three transformations over state space $X = [0,1]^2$:

$$\begin{aligned} \omega_a(x,y) &= (0.5x + 0.5, 0.5y) \\ \omega_b(x,y) &= (0.5x, 0.5y + 0.5) \\ \omega_c(x,y) &= (0.5x, 0.5y) \end{aligned} \tag{4}$$

Limit behavior of the composition of these three transformations is a complex set representation known as the Sierpinski triangle (Fig. 2).

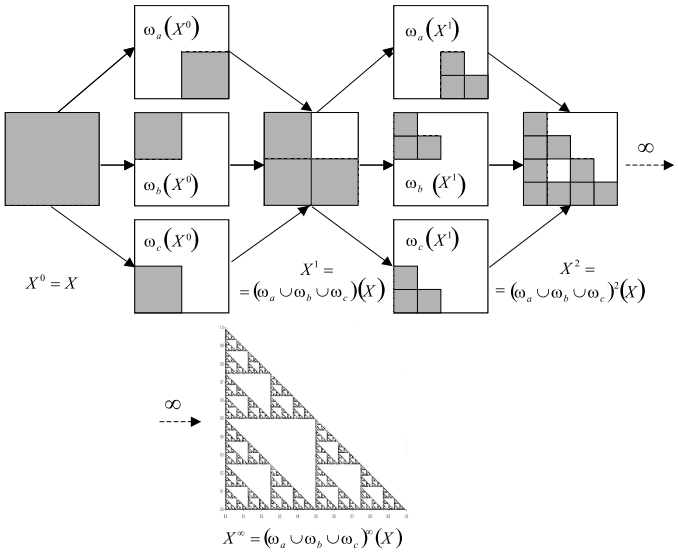

Fig. 2. Sierpinski triange created by infinite number of the union of transformations described by eq. 4

Behavior of recurrent networks in symbolic processing can be explained by IFS theory. For example the dynamics of simple RNN can be expressed by equation

$$\mathbf{x}(t+1) = f\left(\mathbf{W^{in}} \cdot \mathbf{u}(t+1) + \mathbf{W} \cdot \mathbf{x}(t)\right), \tag{5}$$

where f stands for activation function. $\mathbf{W}$ and $\mathbf{W^{in}}$ are matrices with recurrent and input weights respectively. Having finite input alphabet this dynamics can be rewritten to

$$\mathbf{x}(t+1) = f\left(\mathbf{V_i} \cdot \mathbf{x}(t)\right). \tag{6}$$

For each input vector i from the input alphabet $A = \{a_1, \ldots, a_n\}$ corresponding weight matrix $\mathbf{V_i}$ can be found. Both approaches for calculation of the next activations of recurrent units (state) $\mathbf{x}(t+1)$ are identical. When an input symbol appears, corresponding weight matrix $\mathbf{V_i}$ is applied to the current state $\mathbf{x}(t)$. In other words, IFS transformations are represented by weight matrices $\mathbf{V_i}$ and specific input vector selects, which transformation is applied to the current state.

The next IFS is a modification of the previous one. The fourth transformation was added, its attractor is the top right-hand corner of state space:

$$\begin{aligned} \omega_a(x,y) &= (0.5x + 0.5, 0.5y) \\ \omega_b(x,y) &= (0.5x, 0.5y + 0.5) \\ \omega_c(x,y) &= (0.5x, 0.5y) \\ \omega_d(x,y) &= (0.5x + 0.5, 0.5y + 0.5) \end{aligned} \tag{7}$$

Single transformation shrinks the entire image into one-forth sized copy of the original. A position of a point is

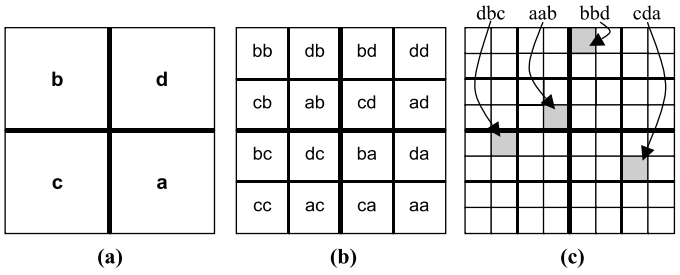

Fig. 3. Regions of points with common IFS address prefixes.

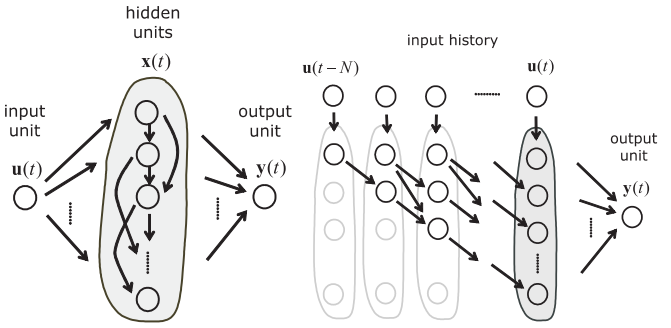

Fig. 4. (a) Modified "feed-forward" ESN architecture. (b) Feed-forwad ESN unfolded in time into regular feed-forward network.

mostly determined by the last performed transformation. This last performed transformation corresponds to the last symbol presented to the network. Next input symbol will release corresponding transformation and again, whole state space is mapped into a specific subspace. But its position within subspace is determined by the second last transformation. With an infinite precision, current point in the state space reflects all performed transformations, i.e. its position is determined by all input symbols. This notion is illustrated in Fig. 3.

The key difference between the ESN and the simple IFS model is the huge and randomly interconnected recurrent layer serving as the ESN reservoir and the real-valued input sequence in the case of the ESN. Nevertheless the principle remains the same and is formulated through the definition of the echo states - activities of internal units reflect the history of the inputs presented to the network with the most recent input having the biggest impact.

IV. Simplified "feed-forward" ESN model

As already mentioned, the proper preparation of the reservoir plays the key role in the ESN training. Usually multiple trials are needed to find appropriate parameters for ideal ESN setup. To help us to understand, how do the ESNs achieve good performance in dynamical modeling by using activities found in the reservoir and possibly improve and facilitate reservoir creation we propose simplified ESN architecture called "feed-forward" ESN (FF-ESN).

Taking into account the architectural bias principles behind the ESN echo states, the network output can be seen as simple nonlinear function of the input history with finite length since the influence of inputs fades out exponentially in time and inputs presented in earlier time steps can be ignored. By removing recurrent connections we propose modified model with the reservoir of units connected in a feed-forward manner. Units in a reservoir can be indexed and their activities depend only on activities of units with smaller indices. No cycles are present in the graph with nodes representing units and edges representing connections. FF-ESN is shown in the Fig. 4a.

Please note, that although we call this network as "feed-forward" ESN, all connections are still recurrent ones because units are fed by activities from previous time steps. But this network can be easily transformed into regular feed-forward network by the process identical to the RNN unfolding in time when using backpropagation through time learning algorithm (see Fig. 4b).

We have used exactly the same training process as is commonly used in training regular ESNs. The only difference is how a recurrent weight matrix is generated. Initial values for biases, recurrent and backward weights falls to the same ranges as described for regular ESNs. Recurrent weight matrix is rescaled to the required spectral radius λ and the matrix is then made lower triangular by keeping only elements below diagonal. To force the ESN to keep longer history of inputs in activities every unit i was connected to the previous one $i-1$ through the weight $w_{i,i-1}$ of chosen constant value, in our experiments we used the value of spectral radius λ.

V. Experimental results

First, we compared FF-ESN with common ESN on simple sinusoid time series $y(n) = 1/2sin^7(n/5)$ with exactly the same parameters and under the same conditions as in [2]. λ for FF-ESN was set to the value of 0.88. For both network architectures 10 simulation runs resulted in the same averaged $MSE_{train} \approx 10^{-10}$ and $MSE_{test} \approx 10^{-10}$.

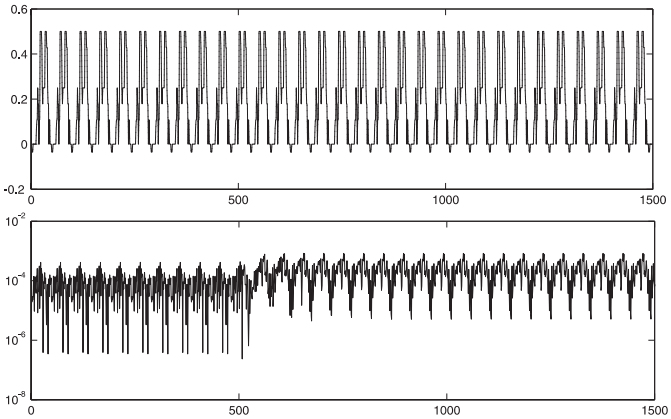

Fig. 5. Generation of periodic discrete sequence ("House of the Rising Sun" [2]) by FF-ESN. The first 500 samples are "teacher forced" and then the network was let to run freely. Original sequence vs. generated one (dotted line) is shown in the top plot, absolute error is shown bellow.

The next simple experiment consisted in training the network to cycle through periodic attractor corresponding to the

discrete sequence ("House of the Rising Sun" [2]). λ parameter for FF-ESN was set to the value of 0.908. The noise of 10^{-3} helped to stabilize the solution in the same way as in regular ESN. Yet again, similar resulting performance of 10 simulation runs was obtained by the FF-ESN and for the ESN, $MSE_{train} \approx 10^{-8}$ and $MSE_{test} \approx 10^{-7}$. Generated and correct output together with the absolute prediction error is shown in the Fig. 5.

Finally, we have tested FF-ESN on the Mackey-Glass (MG) prediction task. Time series was shifted by -1 and $tanh$ function was applied in the same way as described in [5].

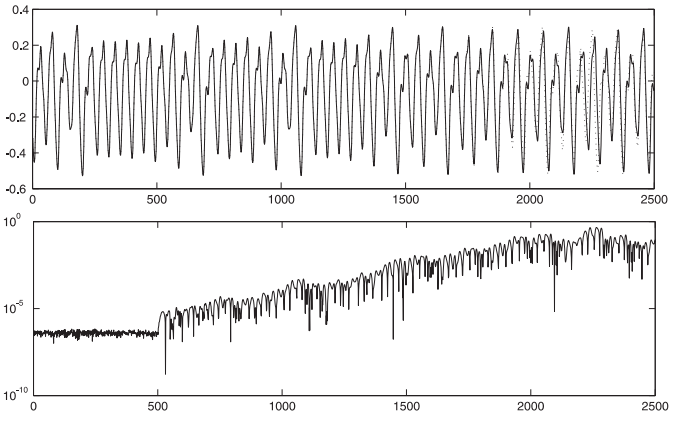

Fig. 6. Modeling Mackey-Glass time series by FF-ESN. The first 500 samples are taken from the initial "teacher forced" sequence.

We used networks with the reservoir of 1000 units with 1% connectivity. Input and recurrent weights were chosen from (-1.0, 1.0), recurrent weight matrix was then rescaled to the spectral radius of 0.8 for ESN and 0.9 for FF-ESN. Each unit in the reservoir has bias weight chosen from (-0.2, 0.2). The noise of 10^{-10} was added to the input sequence to increase the stability of the trained network.

Training and testing was done in the same way as in [5]. The length of the training sequence was 3000, the first 1000 reservoir activities were thrown away and the remaining 2000 activities were used for the $\mathbf{W^{out}}$ calculation. Then the trained network was used to generate following 3000 steps. Generated and correct MG output together with the absolute prediction error is shown in the Fig. 6. The average training error was $MSE_{train} = 1.79 \cdot 10^{-14}$ for FF-ESN and $MSE_{train} = 3.17 \cdot 10^{-14}$ for ESN. We also calculated normalized root mean square error for the 84 step prediction over 10 trials

$$NRMSE_{84} = (\sum_{i=1}^{10}(d(+84) - y(+84))^2/(10\sigma^2))^{1/2}, \quad (8)$$

where d stands for the desired correct MG output and y is the network prediction. Resulting normalized root mean square error was $NRMSE_{84} = 4.73 \cdot 10^{-4}$ for FF-ESN and $NRMSE_{84} = 1.84 \cdot 10^{-4}$ for ESN.

Although very similar results were obtained by using FF-ESN, this architecture is more sensitive to the network initialization parameters. The existence of connections between unit $i-1$ and unit i seems to be crucial for proper function of the FF-ESN, since this connections guarantee sufficient FF-ESN memory capacity.

VI. Conclusions

We showed that by simply done nonlinear combination of past activities we can create prediction machines with very high accuracy comparable to the ESNs, sufficient for the long and precise sequence generation. Since no "real" recurrent connections exist between units (FF-ESN can be transformed into regular feed-forward network), the network output is purely a function of finite sequence history. In regular ESN that meet "echo-state" property activity x_i of dynamical reservoir unit can be expressed as $x_i(t) = e_i(u(t), u(t-1), \ldots)$ where e_i is the "echo function" of indefinitely long history of input signal. On the other hand the activity of FF-ESN reservoir unit x_i can be exactly expressed as a function of finite history of input values $x_i(t) = e_i(u(t), u(t-1), \ldots, u(t-N))$, where N is the number of units in the reservoir, as can be seen in Fig. 4b. This straightforward functional relation can be attractive for deeper study of the echo-state property.

In the proposed model, the dependencies between units are clear and methods adjusting dynamical reservoir by adding or removing units can be studied. For example, since units with higher indices express richer dynamics, reservoir could be iteratively scaled by adding or removing unit with the highest index. Also developing other deterministic approaches to create dynamical reservoir expressing rich and useful behavior can be subject of further research.

Acknowledgment

This work was supported by the grant APVT-20-030204.

References

[1] M.H. Christiansen and N. Chater. Toward a connectionist model of recursion in human linguistic performance. *Cognitive Science*, 23:417–437, 1999.

[2] H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks. Technical Report GMD Report 148, German National Research Center for Information Technology, 2001.

[3] H. Jaeger. Short term memory in echo state networks. Technical Report GMD Report 152, German National Research Center for Information Technology, 2001.

[4] H. Jaeger. Adaptive nonlinear system identification with echo state networks. In *Proceedings of NIPS 02*, 2002.

[5] H. Jaeger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[6] J. F. Kolen. The origin of clusters in recurrent neural network state space. In *Proceedings from the Sixteenth Annual Conference of the Cognitive Science Society*, pages 508–513. Hillsdale, NJ: Lawrence Erlbaum Associates, 1994.

[7] P. Tiňo. Spatial representation of symbolic sequences through iterative function system. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, 29(4):386–392, 1999.

[8] P. Tiňo and G. Dorffner. Recurrent neural networks with iterated function systems dynamics. In *International ICSC/IFAC Symposium on Neural Computation*, 1998.

[9] P. Tiňo, M. Čerňanský, and L. Beňušková. Markovian architectural bias of recurrent neural networks. *IEEE Transactions on Neural Networks*, 15(1):6–15, 2004.

[10] P. Tiňo and M. Čerňanský and L. Beňušková. Markovian architectural bias of recurrent neural networks. In P. Sinčák et al., editor, *Intelligent Technologies – Theory and applications*, pages 17–23. IOS Press, 2002.