ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

# Deep Reservoir Computing: A Critical Analysis

Claudio Gallicchio and Alessio Micheli

Department of Computer Science, University of Pisa
Largo Bruno Pontecorvo 3 - 56127 Pisa, Italy

**Abstract**. In this paper we propose an empirical analysis of deep recurrent neural networks (RNNs) with stacked layers. The analysis aims at the study and proposal of approaches to develop and enhance multiple time-scale and hierarchical dynamics in deep recurrent architectures, within the efficient Reservoir Computing (RC) approach for RNN modeling. Results point out the actual relevance of layering and RC parameters aspects on the diversification of temporal representations in deep recurrent models.

## 1 Introduction

Deep learning models have progressively attracted the interest of the Machine Learning community for their ability to learn data representations at different levels of abstraction. Focusing on the neuro-computing area, the recent extension of deep neural architectures to the case of Recurrent Neural Networks (RNNs) has gained a growing interest [1, 2, 3, 4], in particular in relation to the possibility of developing a hierarchical processing of temporal data. In this concern, some works aimed at achieving multiple time-scales dynamics in a stacked deep RNN architecture by progressively sub-sampling the input to the higher layers [3], forcing the different layers to operate at different frequencies, or by learning the weights of all the layers in the stack, which is an extremely time consuming process even using GPUs and can require ad-hoc incremental training strategies in order to be effective [4]. However, some observations and intuitions present in literature deserve further research and critical assessments. In particular, the observation that stacking RNNs layers inherently creates different time scales at different layers [4, 2], and therefore a hierarchical representation of temporal information per se, is worth to be investigated and analyzed.
In this paper we propose different approaches to achieve a hierarchy in time scales representation by efficient deep recurrent models. In this concern, Reservoir Computing (RC) [5] represents a state-of-the-art approach for extremely efficient RNN modeling, yielding to the possibility of investigating the influence of architectural aspects on the time-scale dynamics differentiation separately from learning. Previous works on hierarchical organizations of RC networks mainly focus on *ad-hoc* architectures of trained modules for multiple temporal feature discovery [5], but still lack in a general view over the effective potentiality and emerging properties of deep architectures of layered reservoirs. Our analysis on the one hand aims at providing an analytic tool for investigating the effects of layering on temporal dynamics representations, and on the other hand paves the way to the development of new models by exploiting the advantages of the stacked deep recurrent architectures in representing different time-scales along with the extreme efficiency of RC training algorithms.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

## 2 Deep Echo State Networks

Within the RC framework, we focus on Echo State Networks (ESNs) [6]. ESNs implement discrete-time dynamical systems, by means of an untrained recurrent reservoir that provides a Markovian representation of the input history [7], and by a trained linear readout. We consider the Leaky Integration ESN (LI-ESN) [8], a variant of the ESN in which the state is updated as $\mathbf{x}(t) = (1-a)\mathbf{x}(t-1) + a\tanh(\mathbf{W}_{in}\mathbf{u}(t) + \hat{\mathbf{W}}\mathbf{x}(t-1))$, where $\mathbf{u}(t)$ and $\mathbf{x}(t)$ are respectively the input and state at time $t$, $\mathbf{W}_{in}$ is the input-to-reservoir weight matrix, $\hat{\mathbf{W}}$ is the recurrent weight matrix and $a \in [0,1]$ is the leaky parameter. The reservoir is left untrained after initialization according to the echo state property (ESP) [6]. We focus our analysis on two key reservoir hyper-parameters: the spectral radius (i.e. the largest eigenvalue in absolute value) $\rho$ of the recurrent weight matrix, and the leaky parameter $a$. The value of $\rho$ is related to the variable memory length and the degree of contractivity of reservoir dynamics [7], with larger values of $\rho$ resulting in longer memory length. It is also related to the ESP for valid ESN initialization, according to which the condition $\rho < 1$ is typically adopted. The value of $a$ is related to the speed of reservoir dynamics in response to the input, with larger values of $a$ resulting in a faster reaction to the input [8, 5]. The readout computes the output as a linear combination of the reservoir state, and is typically trained by direct methods. See [5, 6, 7] for details on RC.

In this paper we propose the study of deep RC architectures in which multiple reservoir layers are stacked one on top of each other. The main model that we consider is a straight stack of reservoirs, called *deepESN* and shown in Fig 1(a). In a deepESN, the first layer is fed by the external input and operates like the reservoir of a shallow ESN, whereas each successive layer is fed by the output of the previous one. The state transition function of deepESNs can be expressed as: $\mathbf{x}^{(l)}(t) = (1-a^{(l)})\mathbf{x}^{(l)}(t-1) + a^{(l)}\tanh(\mathbf{W}_{in}^{(l)}\mathbf{i}^{(l)} + \hat{\mathbf{W}}^{(l)}\mathbf{x}^{(l)}(t-1))$, where the superscript $(l)$ is used when referring to the reservoir state, parameters and input at layer $l$, with $\mathbf{i}^{(0)}(t) = \mathbf{u}(t)$ and $\mathbf{i}^{(l)}(t) = \mathbf{x}^{(l-1)}(t),\ for\ l > 0$. The possible relevance of layering in deepESNs is investigated by considering a RC network containing sub-reservoirs that are all fed only by the input and are not organized in a stack. The resulting architecture is called here groupedESN, shown in Fig.1(b). Finally, the importance of layering with respect to the construction of a progressively more abstract encoding of the input history is studied by considering a deepESN in which the input is provided to every layer. The resulting model is called deepESN Input to All (deepESN-IA), shown in Fig.1(c).

In the following we investigate possible strategies aimed at driving the emergence of different time-scales dynamics through the different layers of a deep recurrent architecture. Our first proposal consists in imposing by design a state dynamics differentiation among the layers, by setting different values of $\rho$ and $a$ at different layers. Varying the values of $\rho$ implies a variability of contractivity and memory length among the state dynamics of different layers. Using different values of $a$ implies a differentiation of state dynamics speed in the different layers. Observe that the advantage of having RNN units with different leaky

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

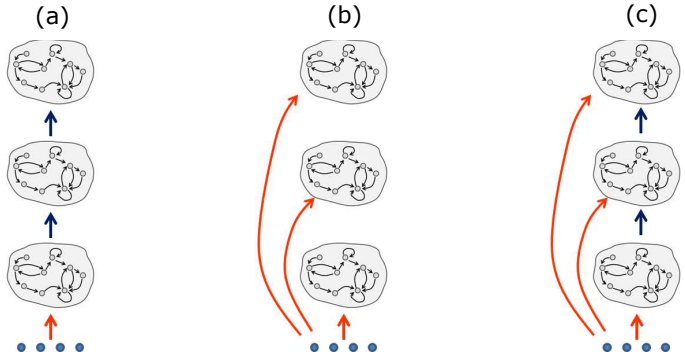


Fig. 1: Deep RC architectures: (a) deepESN, (b) groupedESN, (c) deepESN-IA.

parameters in order to achieve multiple time-scales dynamics has already been discussed in pioneering works of the 1980s [9, 10]. Our second proposal consists in using an efficient unsupervised layer-wise training of reservoir units by means of Intrinsic Plasticity (IP) [11]. The IP adaptation rule aims at maximizing the entropy of each layer's output by a gradient descent learning that adapts the gain and bias parameters of the reservoir units activation function.

## 3 Experiments

To investigate the extent of time-scales differentiation among the different layers similarly to [4], but avoiding biases towards specific applications, we constructed a *time-scales dataset* containing 2 random input sequences. The first *unperturbed* sequence $S_1$ contains 5000 elements uniformly drawn from an alphabet of 10 elements, represented by a 1-of-10 binary encoding. The second *perturbed* sequence $S_2$ differs from $S_1$ only at step 100, where a typo is inserted. We ran the same network on $S_1$ and $S_2$, and collected the obtained states. Then we evaluated for each layer the Euclidean distance between the states corresponding to $S_1$ and $S_2$ as a function of time, to see how long the perturbation affects each layer. We considered deep RC architectures with 12 layers of 10 units each.

In Fig. 2 we present a selection of results representative of the cases of interest, also considering that the limited variability typical of the contractive RC systems does not significantly affects the results under a qualitative point of view. Fig. 2(a), 2(b) and 2(c) show the results achieved with deepESN, groupedESN and deepESN-IA, with $\rho = 0.9$ and $a = 0.55$ fixed for every layer. Continuous blue lines refer to the different layers of the deep architecture, with darker colors corresponding to higher layers. For comparison, the red dotted line refers to a shallow ESN with the same values for $\rho$ and $a$, and reservoir size of 120 (the total number of units in the deep networks). The intrinsic differentiation among the time-scales dynamics at the different layers of a deepESN is shown in Fig. 2(a), from which it is possible to observe that the effects of the input perturbation last longer for higher layers in the stack. Such differentiation is indeed related

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

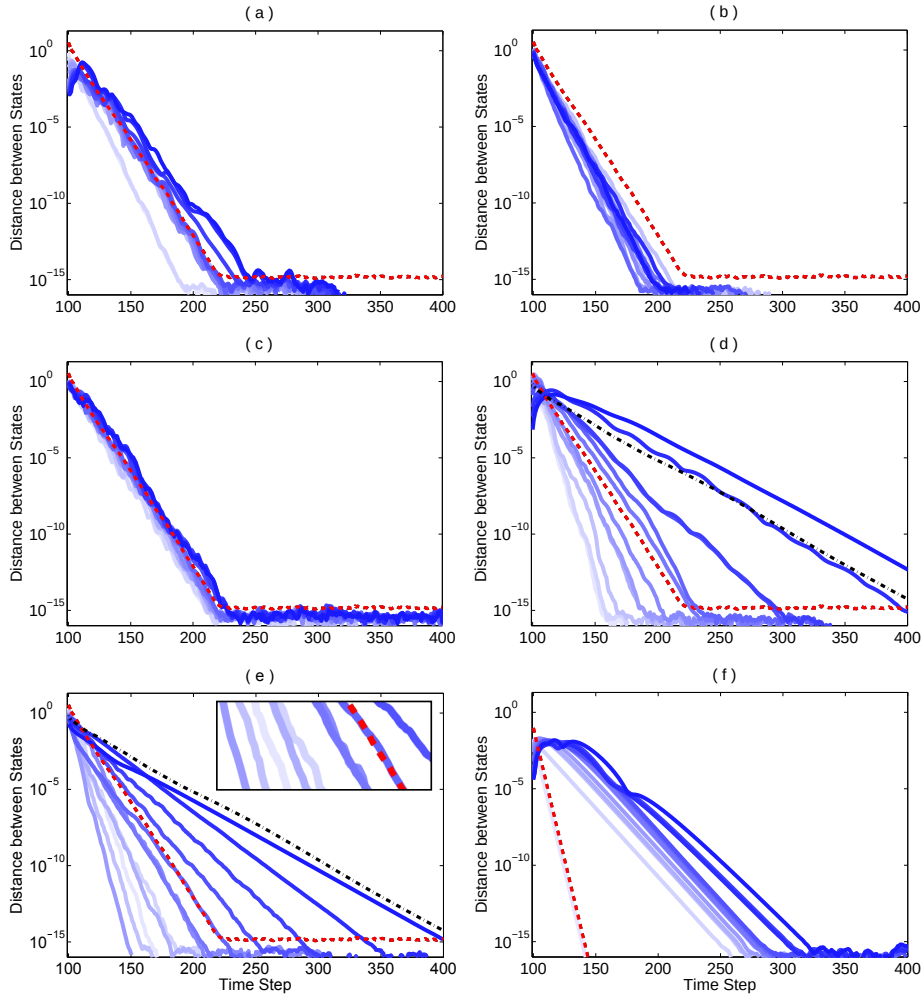


Fig. 2: Distance between perturbed and unperturbed states in **(a)** deepESN, **(b)** groupedESN, **(c)** deepESN-IA, **(d)** deepESN varying $a$, **(e)** groupedESN varying $a$, **(f)** deepESN with IP. Continuous lines correspond to layers in deep RC networks, dotted lines correspond to shallow ESN counterparts (see text).

to the layered deep architecture, as it is strongly attenuated when layering is removed from the architectural design (groupedESN, see Fig. 2(b)) or when the input is provided to each layer (deepESN-IA, see Fig. 2(c)). Fig. 2(b) shows the intrinsic variability that can be already present in reservoirs with the same hyper-parametrization when they are not organized in a stack. In this case all the time-scales dynamics are dominated by the one of a shallow ESN with the same total number of units and hyper-parametrization (red dotted line). A comparison between the behaviors of deepESN (Fig. 2(a)) and deepESN-IA (Fig. 2(c)),

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

brings out that having deeper layers at increasing distance from the input is a key architectural factor for obtaining a time-scales separation. However, it is worth to observe that the inherent differentiation among layers dynamics in a deepESN is quite narrow, with the range of emerging time-scales limited in a small tube around the one obtained with a shallow ESN. Such differentiation can be emphasized within the efficient RC approach by resorting to the strategies proposed in Section 2. Fig. 2(d) shows the results obtained by a deepESN with fixed $\rho = 0.9$ and decreasing leaky parameter $a$ for increasing layer depth, from 1 to 0.1. In the same figure, the red dotted line corresponds to a shallow ESN with $\rho = 0.9, a = 0.55$ (mean value of $a$ among the deepESN layers), whereas the black point-dotted line corresponds to a shallow ESN with $\rho = 0.9, a = 0.1$ (value of $a$ at the deepest layer). As can be seen, the variability of the leaky parameter has a great impact on the separation among the emerging time-scales dynamics, showing an ordered differentiation as in the case of deepESN, but with a much wider extent, reaching even longer times-scales than the shallow ESN with the slowest dynamics. This characterization is a result of the interplay between layering and leaky integration variability, and indeed it is lost when non-stacked architectures are considered. This can be observed in Fig. 2(e), corresponding to a groupedESN with different values of $a$ in different sub-reservoirs. Therein, the overlapping among curves (highlighted in the zoom) shows that the previous order through layers is lost. In addition, it is possible to observe that all the emerging time-scales dynamics in groupedESN are generally below the one obtained by the slower shallow ESN. Similar results (not shown here for brevity) can be obtained for constant $a$ and variable $\rho$, though the effect of differentiation in this case is less significant. The result obtained by using IP is shown in Fig. 2(f), corresponding to a deepESN with values of $\rho = 0.9, a = 0.55$ fixed for all the layers, to which unsupervised IP training is applied. The red dotted line in this case refers to a shallow ESN with the same hyper-parametrization and after IP training. Comparing Fig. 2(f) with Fig. 2(a) it is possible to see the great impact of IP learning on the differentiation among time-scales dynamics of a deepESN, with the perturbation effect persisting in the step range $\approx 150$ - $\approx 350$. Also note that after IP training, the lines representing the dynamics of the first deepESN layer and the shallow ESN one overlap in the plot.

Finally, out of the main scopes of this short paper, we have experimentally assessed the effectiveness of the proposed approach on the *memory capacity* (MC) task [12]. The task was implemented similarly to [11], using RC networks with $\rho = 0.9$, $a = 1$, and averaging the results over 5 guesses. For 100 reservoir units the results achieved by shallow ESNs without and with IP learning are respectively $26.8 \pm 1.8$ and $30.7 \pm 1.7$ (in line with [11]). In the deep case, we considered 10 layers of 10 units each, using the same fixed values of $\rho$ and $a$, and considering the concatenation of all the reservoirs in the stack as input for the readout. Results showed that a great improvement on this task is obtained by deepESN with IP learning, leading to a MC value of $50.7 \pm 1.8$, which represents a performance gain of more than 65% with respect to the shallow case, supporting the potentiality of such approaches for future applications on learning tasks.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

## 4 Conclusions

In this paper we have presented an experimental analysis of deep recurrent architectures, investigated by resorting to stacked RC networks. The introduction of deep RC allowed us to investigate the intrinsic property of deep layered recurrent architectures in representing different time-scale dynamics, finding different insights both in terms of architectural structure (input layer position) and in terms of model parameters (leaky integrator effect enhanced by multi-layer architecture). Moreover, the proposals made to enhance the time-scale hierarchical differentiation among layers (using different leaky parameters or unsupervised IP learning focused only the activation function parameters) allowed us to exploit the efficiency of the RC framework, without resorting to a full RNN training (extended to all the units parameters). On the RC side, the proposed approaches allow us to achieve a time-scale differentiation in the model that is higher with respect to a standard ESN without a layered structure, and lead to explicitly address the concept of including time data representation at different level of abstraction inside the RC paradigm. The proposed analysis would finally suggest the design of new learning models boosted by such enriched representation of the input dynamics that could eventually result in a relevant breakthrough in the area of efficiently learning from sequential and temporal data.

## References

[1] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[2] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio. How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026v5*, 2014.

[3] S. El Hihi and Y. Bengio. Hierarchical recurrent neural networks for long-term dependencies. In *NIPS*, pages 493–499, 1995.

[4] M. Hermans and B. Schrauwen. Training and analysing deep recurrent neural networks. In *NIPS*, pages 190–198, 2013.

[5] M. Lukoševičius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

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

[7] C. Gallicchio and A. Micheli. Architectural and markovian factors of echo state networks. *Neural Networks*, 24(5):440–456, 2011.

[8] H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, 20(3):335–352, 2007.

[9] W.S. Stornetta, T. Hogg, and B.A. Huberman. A dynamical approach to temporal pattern processing. In *NIPS*, pages 750–759, 1988.

[10] S. Anderson, J.W.L. Merrill, and R. Port. *Dynamic speech categorization with recurrent networks*. Indiana University, Computer Science Department, 1988.

[11] B. Schrauwen, M. Wardermann, D. Verstraeten, J.J. Steil, and D. Stroobandt. Improving reservoirs using intrinsic plasticity. *Neurocomputing*, 71(7):1159–1171, 2008.

[12] H. Jaeger. Short term memory in echo state networks. Technical report, German National Research Center for Information Technology, 2001.