Local Lyapunov Exponents of Deep Echo State Networks

Claudio Gallicchio[1,*], Alessio Micheli[1], Luca Silvestri

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

Abstract

The analysis of deep Recurrent Neural Network (RNN) models represents a research area of increasing interest. In this context, the recent introduction of deep Echo State Networks (deepESNs) within the Reservoir Computing paradigm, enabled to study the intrinsic properties of hierarchically organized RNN architectures. In this paper we investigate the deepESN model under a dynamical system perspective, aiming at characterizing the important aspect of stability of layered recurrent dynamics excited by external input signals. To this purpose, we develop a framework based on the study of the local Lyapunov exponents of stacked recurrent models, enabling the analysis and control of the resulting dynamical regimes. The introduced framework is demonstrated on artificial as well as real-world datasets. The results of our analysis on deepESNs provide interesting insights on the real effect of layering in RNNs. In particular, they show that when recurrent units are organized in layers, then the resulting network intrinsically develops a richer dynamical behavior that is naturally driven closer to the edge of criticality. As confirmed by experiments on the short-term Memory Capacity task, this characterization makes the layered design effective, with respect to the shallow counterpart with the same number of units, especially in tasks that require much in terms of memory.

Keywords: Reservoir Computing, Echo State Network, Deep Learning, Deep Recurrent Neural Networks, Deep Echo State Network, Stability

*Corresponding Author
Email addresses: gallicch@di.unipi.it (Claudio Gallicchio), micheli@di.unipi.it (Alessio Micheli), silvestriluca@hotmail.it (Luca Silvestri)

1. Introduction

The extension of deep learning methodologies to the class of Recurrent Neural Networks (RNNs) is currently stimulating an increasing interest in the machine learning community [1, 2]. In this area, the study of hierarchically structured RNN architectures (see e.g. [3, 4, 5, 6, 7, 8]) paved the way to the design of models able to develop feature representations of temporal information at increasing levels of abstraction, enabling a natural approach to tasks on time-series featured by multiple time-scales (especially in the cognitive area). Besides, the elaboration of temporal information in a layered and recurrent fashion is also motivated by strong evidences of biological plausibility emerged from the area of neuroscience [9, 10].

However, the analysis of deep RNNs is relatively young, and one of the major topics still deserving research attention is related to understanding and characterizing their dynamical behavior, especially in relation to the inherent role of the hierarchical composition of the recurrent units in layers. A useful methodology in this regard is provided by the Reservoir Computing (RC) [11, 12] paradigm and the Echo State Network (ESN) [13, 14] approach to RNNs modeling. In particular, allowing to taking apart all the effects due to learning, the recent introduction of the deepESN model [15, 16] enabled the study of the intrinsic role played by the layering factor in deep RNN architectures. Moreover, by inheriting the training characterization typical of standard RC models, deepESNs also provide an efficient methodology for designing and training deep learning models in the temporal domain.

A first mean to investigate the characteristics of recurrent network dynamics is given in the RC area by the Echo State Property (ESP) [17], which has recently been extended to the case of deep networks in [18]. The analysis provided by the study of the ESP conditions in has started to reveal the natural characterizations of deep RNNs under a dynamical system perspective [18], but it might result of reduced utility in practical cases as it basically neglects the influence of the external input on networks dynamics. By their very nature, recurrent neural models implement dynamical systems whose trajectories in the state space are influenced by initial conditions and by the external input signals, which practically realize a link between the system dynamics and the computational task at hand. In this context, the anal-

ysis of stability of deep RNNs dynamics when driven by an external input represents a topic of great importance and still demanded in literature.

In this paper, by pursuing the study of the dynamical behavior of recurrent models typical in the RC area, we provide a theoretical and practical tool that allows us to investigate and control the stability of deep recurrent networks driven by the input. Specifically, we extend the applicability of the study of local Lyapunov exponents [19, 20] from the case of shallow ESNs (see e.g. [21, 22, 12]) to the case of deepESNs. In particular, the maximum among the local Lyapunov exponents is a useful mean to express the network's sensibility to small perturbations of its state trajectories, and as such it can well quantify the degree of stability (or order) in the dynamical behavior of the system. Given the actual input for the system, the proposed methodology can be used to identify the different dynamical regimes that follow from different cases of networks design conditions, such as the RC scaling factors, the number of recurrent units and the depth of the network. The proposed tool is practically demonstrated on artificial data as well as on signals from real-world datasets.

While the developed tool could be certainly applied to the case of deep RNNs at any stage of training, its application in the RC context enables us to investigate the actual role of layering in RNNs and shed light on its natural effect on the richness and stability of the developed network's dynamics. In this regard, a particularly interesting condition of dynamical behavior is represented by the stable-unstable transition where the maximum local Lyapunov exponent is null, a region of the state space known as the edge of criticality. Previous works in the RC literature already showed that the performance of recurrent models for tasks requiring a long memory span peaks near the criticality of their dynamical behavior [23, 24, 25, 26]. Examples are represented by the benchmark tasks in the RC area (e.g. [27, 28, 12, 29, 30]), tasks in the domain of neural circuit models (e.g. [31, 25, 24]), as well as real-world tasks, e.g. in the area of speech processing [12] and mobile traffic load estimation [32]. Although the methodology proposed in this paper is not put forward as a performance predictor for trained recurrent models, as an additional element of analysis here we use it to study the relation between the memory and the regimes of deepESN behaviors through the short-term Memory Capacity task [33].

The rest of this paper is organized as follows. In Section 2 we introduce the basic elements of RC and describe the deepESN model. In Section 3 we provide the mathematical characterization of the stability analysis of deep-

ESNs in terms of the maximum local Lyapunov exponent. The outcomes of our experimental analysis are reported and discussed in Section 4. Finally, conclusions are presented in Section 5.

2. Deep Echo State Networks

Within the framework of randomized neural networks [34], the RC paradigm [11, 12] has attested as a state-of-the-art methodology for efficient RNN modeling. The most widely known model in this context is represented by the ESN [13, 14, 35]. From the architectural perspective, an ESN comprises a recurrent hidden layer of non-linear units, called reservoir, and a feed-forward output layer of typically linear units, called readout. The essence of the ESN operation is that the reservoir part implements a set of randomized filters that serve to dynamically and non-linearly encode the input history into a high dimensional state space, where the task at hand can be approach satisfactorily even by a linear output tool.

From a dynamical system point of view, the reservoir of an ESN computes a discrete-time input-driven non-linear dynamical system, such that at each time step the state evolution is ruled by the reservoir state transition function. By referring to the case of leaky integrator reservoir units [36], at each time step t the reservoir state update equation is given by:

$$\mathbf{x}(t) = (1-a)\mathbf{x}(t-1) + a\tanh(\mathbf{W}_{in}\mathbf{u}(t) + \boldsymbol{\theta} + \hat{\mathbf{W}}\mathbf{x}(t-1)), \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^{N_R}$ and $\mathbf{u}(t) \in \mathbb{R}^{N_U}$ are respectively the reservoir state and the input at time step t, $a \in [0,1]$ is the leaking rate parameter, $\mathbf{W}_{in} \in \mathbb{R}^{N_R \times N_U}$ is the input weight matrix, $\boldsymbol{\theta} \in \mathbb{R}^{N_R}$ is the weight vector corresponding to the unitary input bias, $\hat{\mathbf{W}} \in \mathbb{R}^{N_R \times N_R}$ is the recurrent reservoir weight matrix and tanh denotes the element-wise application of the hyperbolic tangent non-linearity. Typically, a null state is used as initial condition, i.e. $\mathbf{x}(0) = \mathbf{0}$.

The output at time step t is computed by the readout as a linear combination of the activation of the reservoir units, according to the following equation:

$$\mathbf{y}(t) = \mathbf{W}_{out}\mathbf{x}(t) + \boldsymbol{\theta}_{out}, \tag{2}$$

where $\mathbf{y}(t) \in \mathbb{R}^{N_Y}$ is the output at time step t, $\mathbf{W}_{out} \in \mathbb{R}^{N_Y \times N_R}$ is the output weight matrix and $\boldsymbol{\theta}_{out} \in \mathbb{R}^{N_Y}$ is the vector of weights corresponding to the unitary input bias for the readout.

A major peculiarity of the ESN approach is that only the readout undergoes a training process, such that the weights in $\mathbf{W}_{out}$ and $\boldsymbol{\theta}_{out}$ are adjusted on a training set in order to solve a least squares problem, typically in an off-line fashion and in closed form, using of pseudo-inversion or Tikhonov regularization. The reservoir's parameters are instead left untrained after initialization constrained to the dictates of the Echo State Property (ESP) [14]. The ESP states that the reservoir's dynamics should asymptotically depend only on the driving input signal, while dependencies on initial conditions should vanish with time such that the state of the network tends to represent an "echo" of the input. Essentially, the ESP links the asymptotic behavior of the reservoir dynamics to the input signal on which the reservoir is running. Although a certain research effort has being devoted in the last years to describe and understand more and more in depth the conditions under which the ESP holds (see e.g. [37, 17, 38]), two basic conditions are widely adopted in literature for this purpose. Specifically, a sufficient condition and a necessary condition are applied to the weight matrix $\hat{\mathbf{W}}$, requiring to respectively control its 2-norm (i.e. its maximum singular value) and its spectral radius (i.e. the maximum among the eigenvalues in modulus) to be below unity. In the following, we will refer to the standard ESN model, as described by eq. 1 and 2 as *shallow ESN*.

In this paper we are concerned with the extension of the shallow ESN model towards a deep architecture, in which the recurrent component is hierarchically organized into a stack of reservoir layers. The corresponding model is termed deepESN, as introduced in [16, 15]. From a general perspective, it is worth to note that, although several possible ways of constructing deep recurrent architectures have been investigated in literature [3], a stacked composition of recurrent hidden layers is likely to represent the most common choice (see e.g. [4, 5, 6, 8]).

Focusing on the recurrent part of the architecture of a deepESN, graphically illustrated in Figure 1, at each time step the state computation follows a pipeline from the external input towards the higher layer. Specifically, at time step t the first layer is fed by the external input, whereas each layer in the hierarchy at depth higher than 1 is fed by the output of the previous layer at the same time step t.

Keeping the basic notation introduced above for shallow ESNs, here we use N_L to denote the number of reservoir layers in the stacked architecture, assuming for the ease of presentation that every layer has the same dimension (i.e. the same number of recurrent units), which we indicate by N_R.

Moreover, for every $i = 1, 2, \ldots, N_L$, we use $\mathbf{x}^{(i)}(t) \in \mathbb{R}^{N_R}$ to indicate the state of the reservoir in the i-th layer at time step t.

Viewing the deepESN as a whole system, the global state space of the network can be considered as the product of the N_L state spaces of the layers in the architecture. Accordingly, the global state of the deepESN at time step t is represented by $\mathbf{x}_g(t) = (\mathbf{x}^{(1)}(t), \mathbf{x}^{(2)}(t), \ldots, \mathbf{x}^{(N_L)}(t)) \in \mathbb{R}^{N_L N_R}$. From a dynamical system point of view, the global dynamics of a deepESN is ruled by its global state transition function F:

$$F : \mathbb{R}^{N_U} \times \underbrace{\mathbb{R}^{N_R} \times \ldots \times \mathbb{R}^{N_R}}_{N_L \text{ times}} \to \underbrace{\mathbb{R}^{N_R} \times \ldots \times \mathbb{R}^{N_R}}_{N_L \text{ times}} \tag{3}$$

which, given the external input, describes the evolution of the global system's dynamics between two any consecutive time steps, i.e. for every t it results $\mathbf{x}_g(t) = F(\mathbf{u}(t), \mathbf{x}_g(t-1))$. The global state transition function F can be conveniently considered in its layer-wise form, i.e. $F = (F^{(1)}, F^{(2)}, \ldots, F^{(N_L)})$, where for each $i = 1, 2, \ldots, N_L$, $F^{(i)}$ represents the state transition function ruling the dynamics of the i-th layer. In particular, the state transition function of the first layer, i.e. $F^{(1)}$, can be defined as follows:

$$\begin{aligned} &F^{(1)} : \mathbb{R}^{N_U} \times \mathbb{R}^{N_R} \to \mathbb{R}^{N_R} \\ &\mathbf{x}^{(1)}(t) = \quad F^{(1)}(\mathbf{u}(t), \mathbf{x}^{(1)}(t-1)) = \\ &\qquad (1 - a^{(1)})\mathbf{x}^{(1)}(t-1) + a^{(1)} \tanh(\mathbf{W}_{in}\mathbf{u}(t) + \boldsymbol{\theta}^{(1)} + \\ &\qquad\qquad \hat{\mathbf{W}}^{(1)}\mathbf{x}^{(1)}(t-1)), \end{aligned} \tag{4}$$

where $a^{(1)} \in [0, 1]$ is the leaking rate parameter of the first layer, $\mathbf{W}_{in} \in \mathbb{R}^{N_R \times N_U}$ is the input weight matrix (as in eq. 1 for the shallow ESN case),

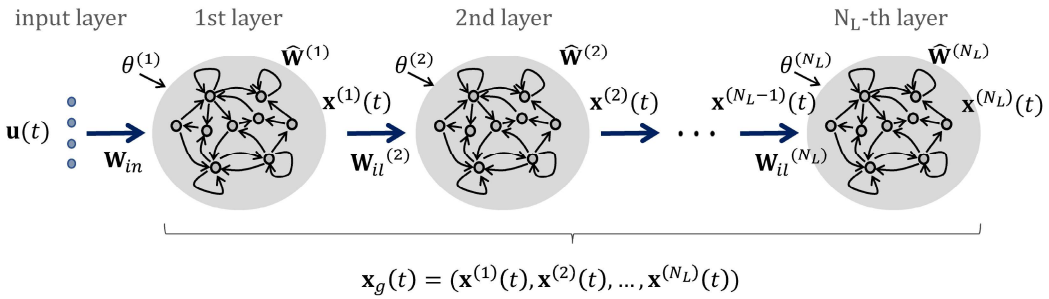

Figure 1: Layered reservoir architecture of a deepESN.

$\boldsymbol{\theta}^{(1)} \in \mathbb{R}^{N_R}$ is the weight vector associated to the unitary input bias for the first layer and $\hat{\mathbf{W}}^{(1)} \in \mathbb{R}^{N_R \times N_R}$ is the recurrent reservoir weight matrix for the first layer.

For layer $i > 1$, the state at time step t, i.e. $\mathbf{x}^{(i)}(t)$, directly depends on the state of the previous layer $i-1$ at the same time step, i.e. $\mathbf{x}^{(i-1)}(t)$, and on the state of the layer i at the previous time step, i.e. $\mathbf{x}^{(i)}(t-1)$, as can be seen in the third line of eq. 5. Apart from the dependence on the state of the same layer at the previous time step, we can further observe that $\mathbf{x}^{(i)}(t)$ recursively depends on the activation at the previous time step $t-1$ of all the previous layers $i-1, i-2, \ldots, 2, 1$, back to the input at the present time step $\mathbf{u}(t)$. Thereby, the state transition function $F^{(i)}$, for $i > 1$, can be defined as follows:

$$
\begin{aligned}
&F^{(i)} : \mathbb{R}^{N_U} \times \underbrace{\mathbb{R}^{N_R} \times \ldots \times \mathbb{R}^{N_R}}_{i \text{ times}} \rightarrow \mathbb{R}^{N_R} \\
&\\
&\mathbf{x}^{(i)}(t) = \quad F^{(i)}(\mathbf{u}(t), \mathbf{x}^{(1)}(t-1), \ldots, \mathbf{x}^{(i)}(t-1)) = \\
&\\
&\qquad (1-a^{(i)})\mathbf{x}^{(i)}(t-1) + a^{(i)} \tanh(\mathbf{W}_{il}^{(i)} \mathbf{x}^{(i-1)}(t) + \boldsymbol{\theta}^{(i)} + \\
&\qquad\qquad\qquad\qquad\qquad\qquad \hat{\mathbf{W}}^{(i)} \mathbf{x}^{(i)}(t-1)) = \\
&\\
&\qquad (1-a^{(i)})\mathbf{x}^{(i)}(t-1) + a^{(i)} \tanh(\mathbf{W}_{il}^{(i)} F^{(i-1)}(\mathbf{u}(t), \mathbf{x}^{(1)}(t-1), \\
&\qquad\qquad\qquad\qquad\qquad\qquad \mathbf{x}^{(2)}(t-1), \ldots, \mathbf{x}^{(i-1)}(t-1)) + \\
&\qquad\qquad\qquad\qquad\qquad\qquad \boldsymbol{\theta}^{(i)} + \hat{\mathbf{W}}^{(i)} \mathbf{x}^{(i)}(t-1)),
\end{aligned}
\tag{5}
$$

where $a^{(i)} \in [0, 1]$ is the leaking rate parameter of the i-th layer, $\mathbf{W}_{il}^{(i)} \in \mathbb{R}^{N_R \times N_R}$ is the inter-layer reservoir weight matrix for layer i, corresponding to the connections from the state of layer $i-1$, i.e $\mathbf{x}^{(i-1)}(t)$, to the state of layer i, i.e. $\mathbf{x}^{(i)}(t)$, $\boldsymbol{\theta}^{(i)} \in \mathbb{R}^{N_R}$ is the weight vector associated to the unitary input bias for layer i and $\hat{\mathbf{W}}^{(i)} \in \mathbb{R}^{N_R \times N_R}$ is the recurrent reservoir weight matrix at layer i.

Remark 1. *Note that a shallow ESN can be obtained as a special case of a deepESN whose reservoir architecture has just one layer. In fact, whenever the hierarchy of reservoirs in the deepESN contains only one layer, i.e. if $N_L = 1$, then the state transition function F in eq. 3 reduces to $F^{(1)}$ in eq. 4, which in turn corresponds to the case of a shallow ESN in eq. 1.*

Remark 2. *It is interesting to observe that a deepESN can be interpreted*

as obtained by applying some constraints to the architecture of a single-layer (shallow) ESN with the same total number of recurrent units. Specifically, the reservoir architecture of a deepESN does not present connections from the input layer to layers at a level higher than 1, as well as connections from higher layers to lower layers and connections from each layer to higher layers different from the one that immediately follows in the pipeline. Under this viewpoint it is possible to see that the deepESN has a simpler architecture than a single-layer ESN, with a reduction in the number of reservoir weights that can be quantified. For instance, assuming full-connectivity in the involved matrices, this absolute reduction amounts to $(\sum_{i=1}^{N_L} N_R N_U) + (\sum_{i=1}^{N_L} N_R^2\ (i-1)) + (\sum_{i=1}^{N_L-2} N_R^2\ i) = N_R N_L N_U + N_R^2(N_L^2 - N_L)/2 + N_R^2(N_L^2 - 3N_L + 2)/2$, *which is quadratic in both the number of units per layer and in the number of layers. This peculiar architectural organization influences the way in which the temporal information is processed by different sub-parts of the hierarchical reservoir, composed by recurrent units that are progressively more distant from the input. Apart from this architectural simplification, notice that layering is implemented by using non-delayed connections between successive reservoir levels. The absence of delays in the transmission of the state information between successive layers enables to process the temporal information at each time step in a deep fashion, through a hierarchical composition of multiple levels of recurrent units. As an additional point, notice that the use of (tanh) non-linearities applied individually to each layer during the state computation does not allow to describe the deepESN dynamics by means of an equivalent shallow system.*
Overall, the combination of the above described constraints (i.e. architectural simplification and non-delayed connections between layers) realizes a specific type of model that is different from the standard ESN and that has distinctive characteristics, started to be analyzed in [15, 18, 16] and further investigated in this paper.

Training a deepESN is a process that is carried out similarly to the case of a shallow ESN, i.e. by adjusting on a training set the parameters of a linear readout layer, whose operation is characterized as in eq. 2. The difference with respect to the shallow ESN case is that at each time step t the input for the readout is given by the global state of the deepESN, i.e. by $\mathbf{x}_g(t)$. Apart from these considerations, in the rest of this paper we shall not further address the aspects related to the network training, keeping the focus of our investigations on the deepESN dynamics.

We shall assume in our analysis that the input and the reservoir state spaces are compact sets. Moreover, from the notation viewpoint, we will use the symbol $\rho(\cdot)$ to denote the spectral radius of its matrix argument, and $\mathbf{I}$ to denote the identity matrix (whose dimension can be obtained from the context).

Recently, the ESP for valid reservoir's dynamics initialization has been extended to the case of deepESNs in [18], where a sufficient condition and a necessary condition for the ESP to hold in case of hierarchical reservoir architectures have been provided. These conditions are briefly recalled in the following, while the reader is referred to [18] for their proofs. In particular, the sufficient condition for the ESP of a deepESN is related to the study of contractivity of the state transition function in eq. 3 that rules the network's dynamics. Basically, such condition states that the maximum rate of contraction among the dynamics of all the reservoir layers should be below unity, as reported in the following Proposition 1.

Proposition 1. *Consider a deepESN whose dynamics is given by the state transition functions in eq. 3, 4 and 5. A sufficient condition for the ESP to hold is then given by*

$$\max_{i=1,\dots,N_L} C^{(i)} < 1, \tag{6}$$

where the $C^{(i)}$ values are Lipschitz constants for the state transition functions of the reservoir layers in the deepESN architectures. Specifically,

(a) for the first layer ($i = 1$)

$$C^{(1)} = (1 - a^{(1)}) + a^{(1)} \|\hat{\mathbf{W}}^{(1)}\|_2$$

(b) for higher layers ($i > 1$)

$$C^{(i)} = (1 - a^{(i)}) + a^{(i)} (C^{(i-1)} \|\mathbf{W}_{il}^{(i)}\|_2 + \|\hat{\mathbf{W}}^{(i)}\|_2).$$

The necessary condition for the ESP of deepESNs is rooted in the analysis of asymptotic stability of network's dynamics. Essentially it says that the maximum among the effective spectral radii of the reservoir layers should be below unity, as stated by the following Proposition 2.

Proposition 2. *Consider a deepESN whose dynamics is given by the state transition functions in eq. 3, 4 and 5, and assume that the set of admissible*

inputs for the system includes the null sequence. Then, a necessary condition for the ESP to hold is given by the following equation:

$$\max_{i=1,\dots,N_L} \rho\big((1-a^{(i)})\mathbf{I} + a^{(i)}\hat{\mathbf{W}}^{(i)}\big) < 1. \tag{7}$$

Notice that in Propositions 1 and 2 we respectively make use of the 2-norm and of the spectral radius. Although these operators are clearly related [39], we recall that their values are generally different unless symmetric matrices are considered, which is typically not the case in common reservoir initialization procedures.

Remark 3. *Note that if the deepESN contains only one reservoir layer, i.e. if $N_L = 1$, then the sufficient and the necessary conditions reported respectively in eq.6 and 7, reduce to the corresponding conditions for the case of shallow ESN, as usually reported in the RC literature [14, 11].*

Here it is worth to observe that, although the necessary condition in eq. 7 is strictly related to the stability analysis of deepESN dynamics, it actually covers only a limited aspect of the topic. Indeed, it deals with asymptotic stability of the null state as fixed point of the state transition function governing the system, under the assumption of a constant null input. In this sense, the analysis of RC dynamics through the spectral radii in the necessary condition expressed by eq. 7 is in fact "blind" to the external input. In more general cases, the stability of the dynamical system implemented by the (hierarchical) reservoir of a deepESN should be analyzed by taking into consideration the actual trajectories in the state space in which the system is driven also in dependence of the actual input signal. This aspect in addressed in the following Section 3.

3. Stability of deepESN Dynamics

In this Section we go more in depth in the analysis of the stability of dynamical systems realized by hierarchically organized RNNs. In particular, in an attempt to remove the constraints that are intrinsic in the analysis provided by the necessary condition for the ESP, we focus on the study of the Lyapunov exponents of the dynamical system implemented by the reservoir part of a deepESN. Lyapunov exponents represent a mathematical tool that provides a measure of the sensitivity of a dynamical system to initial conditions, and as such, in the case of RC networks, are intuitively

and intimately related to the true essence of the ESP. Recently, the link between Lyapunov exponents and the ESP has been further investigated in [40], in which results in the area of mean field theory are used to show the relevant role of Lyapunov exponents in delineating the domain of "local" validity of the ESP in relation to the input [40]. In the RC area, our analysis extends the applicability of the study of the Lyapunov exponents from the context of shallow RC networks (see e.g. [30, 12, 21, 22, 41]) to the case of reservoirs with a structured hierarchical organization.

In dynamical system theory, Lyapunov exponents give a measure of the rate of exponential distortion (i.e. stretching or shrinking) of the trajectories of a dynamical system starting in infinitesimally close initial states [42, 43, 44]. In general, an N-dimensional dynamical system is featured by N Lyapunov exponents, each of which characterizes the rate of distortion along one of the directions in the state space in which the system's trajectory is evolving. If a Lyapunov exponent is smaller than 0 it means that two neighboring trajectories will stay close to each other along the direction corresponding to that exponent. Conversely, if a Lyapunov exponent is greater than 0, then two neighboring trajectories will deviate from each other exponentially fast along the corresponding direction, and in this case predictability of the system can be lost in a relatively short time. The maximum Lyapunov exponent plays a major role in this context, as it dominates the rate of divergence or convergence in the state space and thereby represents a useful indicator of the stability of the whole system during its evolution. In particular, if the maximum Lyapunov exponent has a value that is below (resp. above) 0, then the dynamical system is characterized by stable (resp. unstable) dynamics, with the value of 0 denoting a transition condition known as the edge of criticality [32, 29, 45], the edge of stability [46, 21, 47], or the edge of chaos [25, 24, 23, 30]. In common practical situations, it is useful to consider the local Lyapunov exponents [19, 20], i.e. local finite-time approximations evaluated over a trajectory followed by the dynamical system while it is driven by a real input sequence. The spectrum of the local Lyapunov exponents is strictly related to the Jacobian of the state transition function ruling the dynamics of the system, which describes the instantaneous rate of local distortion at each time step. In particular, such approximations can be computed from the logarithm of the eigenvalues (in modulus) of the Jacobian of the state transition function, typically averaging the outcomes over a long trajectory [42]. In the following, we shall use the symbol λ_{max} to denote the maximum among the local Lyapunov expo-

nents, which in light of the considerations outlined above can be adopted as a practical indicator of the local stability/instability dynamical behavior of the system under consideration along a real trajectory in the state space.

For a deepESN whose dynamics is described by the global state transition function F in eq. 3, and it is layer-wise implemented through equations 4 and 5, the Jacobian matrix at time step t, denoted by $\mathbf{J}_{F,\mathbf{x}_g}(t)$, can be written as a block matrix in the following form:

$$\mathbf{J}_{F,\mathbf{x}_g}(t) = \begin{pmatrix} \mathbf{J}_{F^{(1)},\mathbf{x}^{(1)}}(t) & \mathbf{J}_{F^{(1)},\mathbf{x}^{(2)}}(t) & \dots & \mathbf{J}_{F^{(1)},\mathbf{x}^{(N_L)}}(t) \\ \mathbf{J}_{F^{(2)},\mathbf{x}^{(1)}}(t) & \mathbf{J}_{F^{(2)},\mathbf{x}^{(2)}}(t) & \dots & \mathbf{J}_{F^{(2)},\mathbf{x}^{(N_L)}}(t) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{J}_{F^{(N_L)},\mathbf{x}^{(1)}}(t) & \mathbf{J}_{F^{(N_L)},\mathbf{x}^{(2)}}(t) & \dots & \mathbf{J}_{F^{(N_L)},\mathbf{x}^{(N_L)}}(t) \end{pmatrix} \tag{8}$$

where for $i, j = 1, \dots, N_L$, the block in position (i, j), i.e. $\mathbf{J}_{F^{(i)},\mathbf{x}^{(j)}}(t)$, is the partial derivative of the state transition function of the i-th layer with respect to the state of the j-th layer at previous time step, i.e.

$$\mathbf{J}_{F^{(i)},\mathbf{x}^{(j)}}(t) = \frac{\partial F^{(i)}(\mathbf{u}(t), \mathbf{x}^{(1)}(t-1), \mathbf{x}^{(2)}(t-1), \dots, \mathbf{x}^{(i)}(t-1))}{\partial \mathbf{x}^{(j)}(t-1)}. \tag{9}$$

It is interesting to observe that the hierarchical organization of the reservoir layers in a deepESN architecture is reflected into the shape of the Jacobian matrix, which results to be a lower triangular block matrix. This structure can be exploited in the computation of the Jacobian matrix in eq. 8, as stated by the following Lemma 1.

Lemma 1. *Consider a deepESN with N_L layers, and whose dynamics is given by eq. 3, 4 and 5. For every $i, j = 1, 2, \dots, N_L$ the block element (i, j) of the Jacobian matrix of the global state transition function of the deepESN can be computed as follows:*

$$\mathbf{J}_{F^{(i)},\mathbf{x}^{(j)}}(t) = \begin{cases} \mathbf{0} & \text{for } i < j \\ (1 - a^{(i)})\mathbf{I} + a^{(i)}\mathbf{D}^{(i)}(t)\hat{\mathbf{W}}^{(i)} & \text{for } i = j \\ \left(\prod_{k=0}^{i-(j+1)} a^{(i-k)}\mathbf{D}^{(i-k)}(t)\mathbf{W}_{il}^{(i-k)}\right)\left[(1 - a^{(j)})\mathbf{I} + a^{(j)}\mathbf{D}^{(j)}(t)\hat{\mathbf{W}}^{(j)}\right] & \text{for } i > j \end{cases} \tag{10}$$

where, for every $i = 1, 2, \ldots, N_L$, $\mathbf{D}^{(i)}(t)$ *is a diagonal matrix whose non-zero entries are given by* $(1-(\tilde{x}_1^{(i)}(t))^2)$, $(1-(\tilde{x}_2^{(i)}(t))^2)$, $\ldots$, $(1-(\tilde{x}_{N_R}^{(i)}(t))^2)$, *where the elements of the vector* $\tilde{\mathbf{x}}^{(i)}(t)$ *are defined as:*

$$\tilde{\mathbf{x}}^{(i)}(t) = \begin{cases} \tanh(\mathbf{W}_{in}\mathbf{u}(t) + \boldsymbol{\theta}^{(1)} + \hat{\mathbf{W}}^{(1)}\mathbf{x}^{(1)}(t-1)) & \text{if } i = 1 \\ \tanh(\mathbf{W}_{il}^{(i)} F^{(i-1)}(\mathbf{u}(t), \mathbf{x}^{(1)}(t-1), \ldots, \mathbf{x}^{(i-1)}(t-1)) + \\ \qquad \boldsymbol{\theta}^{(i)} + \hat{\mathbf{W}}^{(i)}\mathbf{x}^{(i)}(t-1)) & \text{if } i > 1. \end{cases} \tag{11}$$

Proof. The proof is given in Appendix A. □

The knowledge of the Jacobian of the global state transition function of a deepESN is useful, e.g. for the computation of local first-order approximations of the network's state evolution over time. However, whenever we are interested in the computation of the eigenvalues of the Jacobian in eq. 8 we can exploit a result from matrix theory to simplify the computation and avoid explicit calculation of the lower triangular blocks of the Jacobian. This result is reported in the following Lemma 2.

Lemma 2. *Consider a deepESN with* N_L *layers, and whose dynamics is given by eq. 3, 4 and 5. Then the set of the eigenvalues of the Jacobian matrix of its global state transition function (in eq. 8) is given by the set of the eigenvalues of its diagonal block matrices, i.e. for every time step* t *it results*

$$eig(\mathbf{J}_{F,\mathbf{x}_g}(t)) = \{eig(\mathbf{J}_{F^{(i)},\mathbf{x}^{(i)}}(t)) | i = 1, 2, \ldots, N_L\}, \tag{12}$$

where $eig(\cdot)$ *is used to denote the set of eigenvalues of its matrix argument.*

Proof. The proof is given in Appendix B. □

Given the results of the previous Lemmas 1 and 2, we can now provide an analytical expression for the λ_{max} of a deepESN, as stated in the following Theorem 1.

Theorem 1. *Consider a deepESN with* N_L *layers, whose dynamics is given by eq. 3, 4 and 5, and that is driven by an input sequence of length* N_s*. Then the maximum local Lyapunov exponent of the dynamical system implemented*

by the (hierarchical) reservoir component of the deepESN can be estimated as follows:

$$\lambda_{max} = \max_{\substack{i=1,\dots,N_L \\ k=1,\dots,N_R}} \frac{1}{N_s} \sum_{t=1}^{N_s} \ln\left(|eig_k\big((1-a^{(i)})\mathbf{I} + a^{(i)}\mathbf{D}^{(i)}(t)\hat{\mathbf{W}}^{(i)}\big)|\right), \tag{13}$$

where $|eig_k(\cdot)|$ denotes the modulus of the k-th eigenvalue of its matrix argument and $\mathbf{D}^{(i)}(t)$ is defined as in Lemma 1.

Proof. Given the input sequence of length N_s and the resulting network's dynamics, the λ_{max} value can be estimated by the quantity:

$$\lambda_{max} = \max_{k'=1,2,\dots,N_L N_R} \frac{1}{N_s} \sum_{t=1}^{N_s} \ln(|\mathrm{eig}_{k'}(\mathbf{J}_{F,\mathbf{x}_g}(t))|), \tag{14}$$

where the index k' spans over the entire dimension of the whole Jacobian. By applying the results of Lemmas 1 and 2 directly to the expression in eq. 14, we have that at each time step, the maximum of the $|\mathrm{eig}_{k'}(\mathbf{J}_{F,\mathbf{x}_g}(t))|$ values corresponds to the maximum of the $|\mathrm{eig}_k\big((1-a^{(i)})\mathbf{I}+a^{(i)}\mathbf{D}^{(i)}(t)\hat{\mathbf{W}}^{(i)}\big)|$ values. From this, the statement of the theorem easily follows. □

The value of the λ_{max} estimate derived in Theorem 1 provides us with useful information about the quality of the deepESN dynamics in terms of stability. In particular, from eq. 13 we can see that the relation to the driving input signal at each time step is embedded into the $\mathbf{D}^{(i)}(t)$ matrices, through the actual values of the network's states along the trajectory followed in response (also) to the history of the inputs that have excited the reservoir. In this sense, the condition $\lambda_{max} < 0$ can provide a mean for assessing the network stability that is more complete than the one that is given by the necessary condition for the ESP in eq. 7. On the other hand, if we restrict our analysis around the null state and assume null input for the system, then the condition $\lambda_{max} < 0$ reduces back to the necessary condition for the ESP in eq. 7.

Remark 4. *Note that if the deepESN contains only one reservoir layer, i.e. if $N_L = 1$, then the formula for the computation of λ_{max} in eq. 13 reduces to the case of shallow ESNs as reported in literature (see e.g. [21, 22, 46]).*

Remark 5. *Here it is worth noticing that* in general cases *the value of λ_{max} should not be considered as a predictor of the network performance on computational tasks. On the one hand, values of $\lambda_{max} > 0$ denote an unstable (chaotic) network's behavior, which is undesirable as it would imply that two input sequences that only slightly deviate from each other will lead the same network towards completely different states, thus compromising the generalization ability of the model. On the other hand, too small values of λ_{max} could denote a dynamic regime that is restricted into a region of excessive stability (order) of network's states, in which differences in the external input do not have a strong impact on the state. In such cases it is possible that the network forgets the previous inputs too fast, resulting in a system with limited memory that could not be suitable to tackle some kind of tasks (e.g. those requiring much in terms of memory). Essentially, the value of λ_{max} that is more appropriate to solve a specific task results from a delicate trade-off between stability of dynamics and memory span, as already discussed in the RC literature in terms of the relation between pairwise separation property and fading memory of reservoirs [23]. Thus, the result provided by Theorem 1 should not be interpreted as a mean to predict the performance on general tasks, instead it should be seen as a tool to accurately control the regime of system dynamics developed by a deep recurrent architecture.*

Remark 6. *By inspecting eq. 13 we can notice that the individual contributions given to the λ_{max} computation by the different layers in the deep recurrent architecture are collectively aggregated by means of the maximum operator. This means that when we consider a deep recurrent architecture with progressively more layers, the resulting value of λ_{max} can never decrease as the number of layers increases. In this sense, i.e. in the process of incremental network construction, we can see that the value of λ_{max} is a monotonic non-decreasing function of the number of layers.*

Remark 7. *A closer look at the λ_{max} formula in eq. 13 allows us to give an insight on the conditions under which adding reservoir layers to a deepESN architecture actually increases the value of λ_{max} and thereby lowers the degree of stability of the system. Assuming that all the layers in the network are featured by the same hyper-parameterization (leaking rate, inter-layer and recurrent weight matrices), then we can see that for each time step the eigenvalues (in modulus) of $\mathbf{J}_{F^{(i)},\mathbf{x}^{(i)}}(t)$ are upper bounded e.g. by its 2-norm. Following this line of reasoning, we can see that if the norm of the inter-layer weights is smaller than 1, then the bounds on the eigenvalues will tend to increase at*

each layer. Thereby, whenever one desires to have a λ_{max} trend characterized as an increasing function of the number of reservoir layers, all with the same properties, our suggestion is to use "small" weights in the inter-layer connections.

Remark 8. *The mathematical framework of Theorem 1, although proposed for deep RC models, can be applied also to the case of trained deep RNNs made up of stacked recurrent layers, at any stage of the training process, taking care of the evolution of the recurrent weight matrices over time.*

4. Numerical Simulations

In this section we practically demonstrate the tool for stability analysis of deepESN dynamics developed in the previous Section 3. We first investigate the effect of the variation of deepESN hyper-parameters on the resulting values of λ_{max} for increasing number of network's layers. Then, we focus on assessing the effect of layering itself on the λ_{max} value, varying the conditions for network's setting and in comparison with shallow ESN cases. Finally, in Section 4.1 we investigate the relations to the short-term Memory Capacity.

In our experimental investigation, we considered deepESN where the weights in $\mathbf{W}_{in}$ and $\hat{\mathbf{W}}^{(1)}$, as well as, for every layer $i > 1$, the weights in $\mathbf{W}_{il}^{(i)}$ were chosen from a uniform distribution over $[-1, 1]$. For the sake of simplicity, all the bias terms for the reservoir layers were included into the corresponding input or inter-layer weight matrix. The values in $\hat{\mathbf{W}}^{(i)}$ were randomly chosen in $[-1, 1]$, and then re-scaled to meet a desired spectral radius value, denoted by $\rho^{(i)}$, whereas the values in $\mathbf{W}_{in}$ were re-scaled to a desired value of its 2-norm, an input scaling parameter denoted by $scale_{in}$, i.e. $\|\mathbf{W}_{in}\|_2 = scale_{in}$. Analogously, the values in each $\mathbf{W}_{il}^{(i)}$ were re-scaled to a desired 2-norm, an inter-layer scaling parameter denoted by $scale_{il}^{(i)}$, i.e. $\|\mathbf{W}_{il}^{(i)}\|_2 = scale_{il}^{(i)}$. For the sake of simplicity, we used the same hyper-parameterization in all the layers, i.e. for all $i = 1, 2, \ldots, N_L$, we considered $\rho^{(i)} = \rho$, $a^{(i)} = a$ and $scale_{il}^{(i)} = scale_{il}$. We generally varied the considered values of the hyper-parameters, through the different experiments, in the following ranges: $\rho \in [0.5, 1.5]$, $a \in [0.1, 1]$, $scale_{in} \in [0.1, 1]$, and $scale_{il} \in [0.1, 1]$, using in every case a step-size of 0.1. The values of ρ, a, $scale_{in}$ and $scale_{il}$, as well as the number of layers N_L, were set in the different cases with the sole purpose of analysis and without any aim to optimize the achieved results. For every hyper-parameterization we independently generated 100 network realizations (i.e. guesses with different random

seeds), and averaged the results over such realizations. As driving input signal for λ_{max} computation[1] we considered a one-dimensional time-series ($N_U = 1$) of length 5000 ($N_s = 5000$), whose elements were individually drawn from a uniform distribution in $[-0.8, 0.8]$, and whose first 100 steps were used as initial transient for the reservoir states.

In a first set of experiments, we targeted the analysis of λ_{max} variability resulting from changing the values of the deepESN hyper-parameters, while increasing the number of layers and the total number of reservoir units. For these experiments we considered reservoir layers with $N_R = 10$ units, increasing the number of layers from 1 to 10, i.e. $N_L \in \{1, 2, \ldots, 10\}$. The results of the λ_{max} computation under the considered settings are graphically reported in Fig. 2, where progressively lighter colors correspond to higher values of λ_{max}, i.e. to progressively less stable networks dynamics.

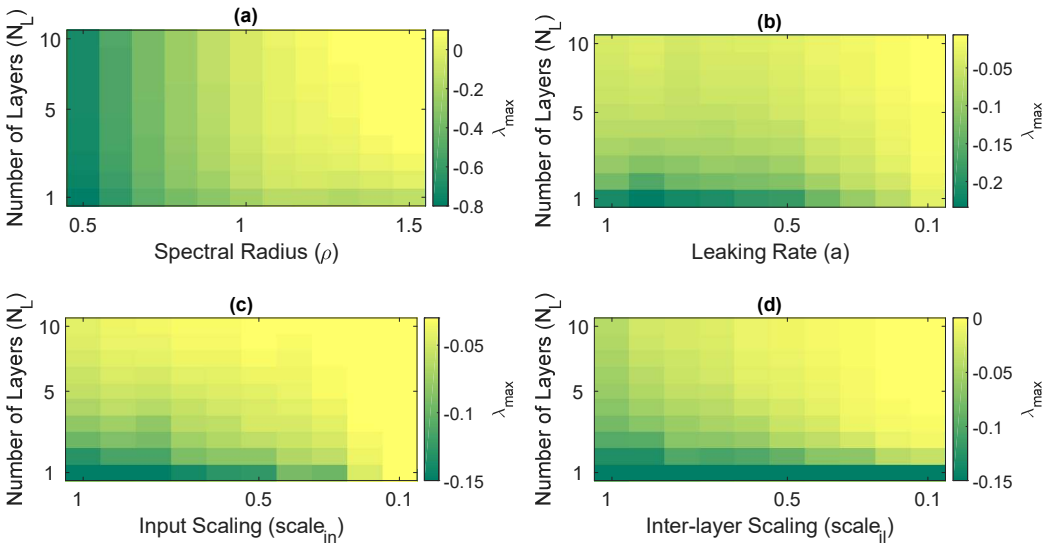

Figure 2: Averaged values of λ_{max} obtained by deepESN for increasing number of reservoir layers. **(a)**: Increasing values of ρ, $a = 1$, $scale_{in} = 1$, $scale_{il} = 1$. **(b)**: Decreasing values of a, $\rho = 1$, $scale_{in} = 1$, $scale_{il} = 1$. **(c)**: Decreasing values of $scale_{in}$, $\rho = 1$, $a = 1$, $scale_{il} = 1$. **(d)**: Decreasing values of $scale_{il}$, $\rho = 1$, $a = 1$, $scale_{in} = 1$.

As we can see, the developed tool allows us to identify regions in the network's hyper-parameters space characterized by different degrees of stability

[1]Every 50 time steps, which did not affect the precision of the results.

of the involved state dynamics. Specifically, we can observe in Fig 2(a) that higher values of ρ result in higher values of λ_{max}, confirming what has been observed also in shallow ESNs e.g. in [22]. Moreover, we can see that the value of λ_{max} tends to increase for increasing number of layers in the deep recurrent architecture, eventually switching from stability to an unstable behavior in correspondence of the higher values of ρ and of N_L. From Fig. 2(b), 2(c) and 2(d) we can appreciate the effect of the variation of a, $scale_{in}$ and $scale_{il}$, respectively, on the value of λ_{max}. As a rule of thumb, we can see that for all these three hyper-parameters smaller values lead to higher values of λ_{max}, with an increasing trend for increasing N_L, although the general impact appears to be less strong than in the case of ρ, as it can be observed by comparing the ranges of λ_{max} variation in the different cases. Results achieved on the synthetic data described so far are qualitatively confirmed by the experimental assessment on cases with real-world input, as reported in Appendix C.

In a second set of experiments, we have analyzed more in depth the impact of the hyper-parameters that led to a higher excursion in the results illustrated in Fig. 2, namely the spectral radius and the number of recurrent units (layers in the deepESN). We started by comparing the values of λ_{max} obtained by deepESNs for increasing values of the ρ hyper-parameter, varying the number of networks layers while keeping constant the number of total reservoir units to 100. Fig. 3 shows the achieved values of λ_{max} when the 100 reservoir units are arranged into 20, 10 and 5 layers, where in each case results are averaged (and standard deviations are computed) over the network realizations. For a further comparison, in the same figure we also plotted the results achieved with a shallow ESN with the same hyper-parameterization and total number of reservoir units. From Fig. 3 the effect of layering emerges clearly: a higher number of layers leads to a higher value of λ_{max}. This effect is amplified and becomes increasingly significant for increasing values of ρ (as can be seen by looking also at the standard deviations in Fig. 3). Moreover, from the same figure we can see that from a certain point on, the same values of λ_{max} achieved by shallow ESNs are obtained by layered reservoirs in correspondence of smaller values of ρ. Although Fig. 3 refers to one among the possible choices of the a, $scale_{in}$ and $scale_{il}$ parameters, we observed that the emerging properties are qualitatively independent of such choice[2].

[2]Exceptions are represented by the degenerate cases achieved in correspondence of a

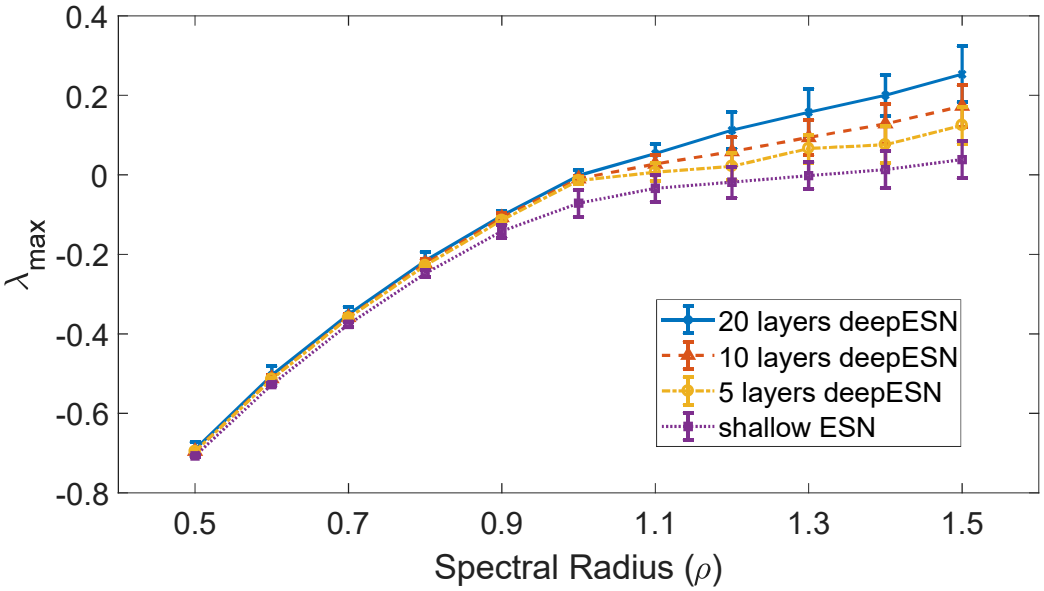

Figure 3: Averaged values of λ_{max} obtained by deepESN with a total number of 100 reservoir units organized in 20, 10 and 5 layers. The considered hyper-parameterization is for increasing values of ρ and $a = 1$, $scale_{in} = 1$ and $scale_{il} = 0.5$. Results achieved by a shallow ESN with the same hyper-parameterization and the same number of reservoir units are reported as well for the sake of comparison.

An analogous experimental assessment, but for increasing total number of recurrent reservoir units, is provided in Fig. 4, which shows the λ_{max} values obtained by deepESNs in which the recurrent units are organized in a progressively higher number of layers, each composed of 10 units. For comparison, in the same figure we plotted the results achieved, under the same experimental conditions and number of reservoir units, by shallow ESNs and ESNs in which the reservoir units are arranged into an increasing number of non connected groups. The latter represents an architectural variant called groupedESN, which in practice implements a similar degree of sparsity than deepESN, but neglecting the layering factor. Specifically, in the architecture of a groupedESN the recurrent units are organized in groups, or sub-reservoirs, each of which is fed by the external input, while connections among the units in different sub-reservoirs are avoided. The groupedESN has been introduced in [15, 16] as an architectural baseline for comparative assess-

values close to 0.

ment of the impact of the layering factor on the characterization of the state evolution in deepESNs. For all the cases considered in Figure 4, the plot shows the values averaged (and the standard deviations computed) over the different network realizations. Results in Fig. 4 clearly show that organizing

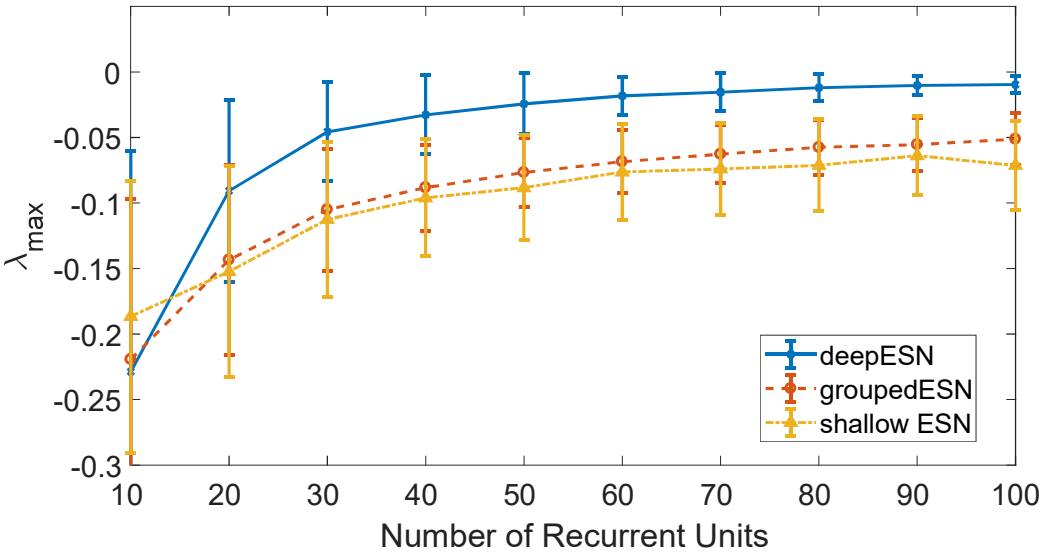

Figure 4: Averaged values of λ_{max} obtained by deepESN for increasing number of reservoir units, organized in layers of 10 units each. The considered hyper-parameterization corresponds to $\rho = 1$, $a = 1$, $scale_{in} = 1$ and $scale_{il} = 0.5$. Results achieved by a shallow ESN and groupedESN with the same hyper-parameterization and the same number of reservoir units are reported as well for the sake of comparison.

the same number of recurrent units into a layered architecture naturally and systematically leads to an overall network's dynamics that is characterized by higher values of λ_{max}, with regimes closer to the edge of criticality. In this respect, the impact of layering can be observed already for networks with a few tens of recurrent units (i.e. with a small number of layers in the deepESN experimental setting), while it becomes increasingly significant for increasing number of units (by looking at the standard deviations in Fig. 4). Moreover, we can observe that the results achieved for increasing number of units by groupedESNs and shallow ESNs are very close to each other, and, in comparison, a smaller number of units in deepESNs leads to similar (and even higher) values of λ_{max}. Looking at the standard deviations in the plot in Figure 4 we can also observe that the variability of the achieved results tends to decrease for increasing network's sizes (a common effect in RC due to the

randomized initialization process on the reservoir's weights, see e.g. [12]), with an even more noticeable effect in the case of deepESN. Furthermore, it is worth noticing that while the case illustrated in Fig. 4 is well representative of the emerging behavior, we observed the same characterization of the dynamics also for other choices of models hyper-parameters in the considered ranges[3].

Overall, the findings described in this section revealed the intrinsic role of layering in recurrent neural models in terms of increased values of the maximum local Lyapunov exponent λ_{max}. This also implies that compared to the shallow case, deep recurrent networks can be brought more easily close to the critical region of stable-unstable transition of system dynamics. In this regard it is worth mentioning that, although in general not directly related to the predictive performance on learning tasks, the edge of criticality identifies an region of dynamical regimes in which interesting and rich behaviors emerge. In fact, several evidences have been reported in literature that RNNs whose dynamics is brought close to the edge of criticality are able to develop richer representations of their input history, with the ability to perform complex processing on temporal data and likely resulting in better performances on computational tasks that require longer short-term memory abilities (see e.g. [23, 24, 25, 26]). The fact that the performance of RNN models has a peak when the their dynamics are nearby the critical transition is a known fact in literature for several classes of problems, e.g. in real-world tasks in the areas of speech processing [12] or telephone load predictions [32]. Other examples are provided by benchmark tasks in the area of spiking neuron models, such as the computation of parity functions of spiking inputs or the classification of noisy spike patterns (see e.g. [31, 25, 24]), as well as in the ESN area, including the assessment of short-term memory and modeling of sinusoidal, Mackey-Glass, and NARMA systems (see e.g. [29, 30, 12, 32]). In particular, a well representative task in this regard is given by the short-term Memory Capacity task [33], whose relation to the critical regime of system dynamics has already been experimentally shown e.g. in [27, 28, 12, 30, 29]. In this context, allowing to incorporate the influence of the driving input signal in the analysis of network stability, the control of the maximum local Lyapunov exponent represents a more accurate way to characterize the

[3]Exceptions are represented by the degenerate cases obtained for large values of $\rho > 1$ and for values of a or $scale_{in}$ close to 0.

richness of reservoir dynamics in the critical regime than just controlling the spectral radius of the recurrent reservoir weight matrix, which in this respect acts just as an a-priori measure [27, 22, 12, 46, 37].

The experimental results illustrated in this section showed that organizing the same amount of recurrent units into layers has the inherent effect of accelerating the process of approaching the stability limit of network's dynamics. In this sense, layering in recurrent networks can be seen as a sort of reservoir optimization methodology that is both simple and cheap, allowing a network with a smaller total number of recurrent units (than required in shallow cases) to develop a state behavior within a rich dynamical region. As already expressed in Remark 5, in general the performance on a computational task is strongly related to the properties of the task itself (and its target). Thereby, in general cases, dynamics at the edge of stability could neither guarantee nor be necessary to obtain good results (just like setting a spectral radius value close to 1 does not necessarily lead to a better performance in shallow ESNs). Our analysis suggests that whenever one knows that the temporal task at hand has a characterization that requires dynamics close to the edge of stability to be properly addressed, then it is well advisable to organize the recurrent units into layers and control the proximity to the boundary of stability using the mathematical framework proposed in this paper. Besides, note that also when the task at hand is known to require recurrent dynamics quite far from the border of stability, our framework can be used to control the dynamical regime developed by the network and keep it within the region of interest for the task. Moreover, in uncertain cases, when nothing is known on the task to be approached, the reservoir organization by layering could anyway represent a convenient choice for network construction. The results of our analysis, on the one hand, contribute to explain the potentiality of layered recurrent models in outperforming shallow networks with the same number of recurrent units [15, 16] in tasks for which reservoirs brought to the limit of stability have shown a performance maximization, such as the short-term Memory Capacity (see [30] for a discussion on this aspect in the case of shallow ESNs). On the other hand, as the increase in the number of layers could result in an undesired unstable network dynamics, they also emphasize the importance of the analysis and control of the λ_{max} value through the proposed methodology.

The relation between λ_{max} and the short-term Memory Capacity of deep-ESNs is investigated in the following Section 4.1.

4.1. Short-term Memory Capacity

The short-term Memory Capacity (MC) task has been introduced in [33] with the aim of assessing the ability of a reservoir network to precisely recall its previous input history from its state. Specifically, the task consists in training different readout units to reconstruct the input signal with an increasing delay. Using $y_d(t)$ to denote the output of the readout unit trained to recall the input signal with delay d, i.e. $u(t-d)$, the MC is defined as follows:

$$MC = \sum_{d=0}^{\infty} r^2(u(t-d), y_d(t)) \tag{15}$$

where $r^2(u(t-d), y_d(t))$ is the squared correlation coefficient between the signals $u(t-d)$ and $y_d(t)$. As input signal for the task we adopted the same sequence used for the experiments described in Section 4, prolonging it to a total length of 6000 time steps. Specifically, the first 5000 time steps have been used for readout training using pseudo-inversion (with a transient of 100 steps), leaving the remaining 1000 time steps for test. Following a similar approach to e.g. [48, 30], we practically implemented the task by considering a finite number of 200 delays, equal to the twice the maximum number of total reservoir units used in the experiments, i.e. 100 in our case. Also for the experiments described in this section, for each hyper-parameterization considered, the presented results were obtained as the average over 100 network realizations (i.e. guesses with different random seeds for initialization).

We started by considering deepESNs with 10 layers of 10 units each, varying the value of ρ in $[0.5, 1.5]$ and the value of $scale_{il}$ in $[0.1, 1]$, keeping fixed $a = 1$ and $scale_{in} = 1$. We computed the values of λ_{max} for the different cases on the training sequence, plotting the achieved data versus the corresponding test MC values. The result is illustrated in Fig 5. As it can be seen, the MC values tend to increase for increasing values of λ_{max} with a peak close to the edge of stability (i.e. for $\lambda_{max} \approx 0$), after which the MC suddenly drops. We could also observe that the MC values tend to naturally cluster based on the values of the ρ hyper-parameter.

A further set of experiments aimed at comparing the MC values obtained by deepESNs with those obtained by shallow ESNs under the same range of settings, varying the number of recurrent units and the value of ρ. For these experiments we considered an increasing number of total reservoir units varying from 10 to 100, organized in layers of 10 units each in the case of deepESNs, and in a single layer in the case of shallow ESN. Note that this

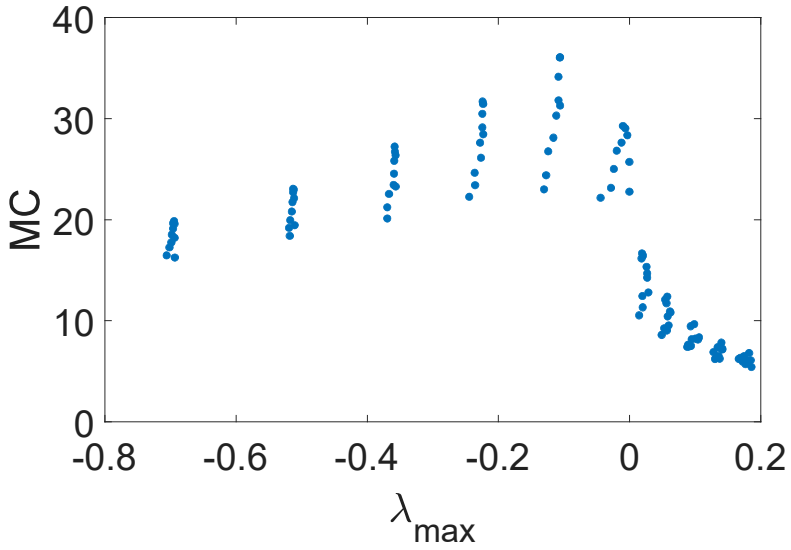

Figure 5: Test MC values of deepESNs plotted versus the λ_{max} estimated on the training set.

choice allowed us to perform a fair comparison between the deep and shallow cases, under the condition of using the same number of trainable weights for the linear readout (i.e. the same number of free parameters for the learner). As regards the other aspects of networks hyper-parameterization, we considered values of ρ in $[0.5, 1.5]$, while keeping fixed the values of $scale_{il} = 0.5$ and $scale_{in} = 1$ (in order to ensure, for both deepESNs and shallow ESNs, to have similar conditions of reservoir non-linearities operating sufficiently far from a linear regime, where the MC results could be biased towards higher values).

The results are illustrated in Fig. 6, where lighter colors corresponds to higher values of the MC. A comparison between the results of deepESN and shallow ESN, respectively in Fig. 6(a) and Fig. 6(b), clearly highlights the advantage of the layered architecture and, in light of the results in Fig. 5, it reflects the evidences reported in Section 4. First of all, we can see that deepESNs achieve higher values of MC than shallow ESNs under the same experimental conditions, which is in line with previous results on hierarchichal reservoir architectures [15, 18]. Secondly, and more importantly, results in Fig. 6 point out that deepESNs are able to reach higher values of MC "before" shallow ESNs, both in terms of smaller ρ values and in terms of a smaller total number of recurrent units (i.e. with a cheaper network's ar-

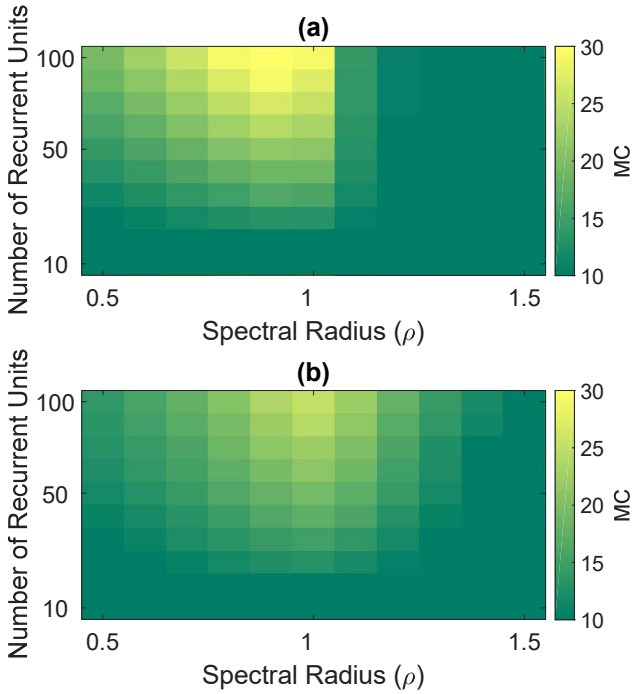

Figure 6: Test MC values for increasing number of total reservoir units and ρ. **(a)**: deepESNs with reservoir layers of 10 units; **(b)**: shallow ESNs under the same hyper-parameterization and number of reservoir units.

chitecture), which confirms the results on the λ_{max} computation provided in Section 4 (see Fig. 3 and 4). Moreover, we can see that the drop in the MC value for higher values of ρ is rather abrupt for deepESNs, while it is smoother for shallow ESNs. In light of our analysis, such observation can be explained in terms of the higher rate of λ_{max} increase emerged in layered reservoir architectures for increasing values of ρ (see Fig. 3).

5. Conclusions

In this paper we have addressed the study of deep RNNs from a dynamical system viewpoint, focusing on the fundamental issue of stability. To this aim, we have developed a mathematical tool that extends the applicability of the analysis through the local Lyapunov exponents to the case of hierarchically organized recurrent neural models. In the analysis of the network's stability, such a tool allows us to take into practical account also the external input signals that are actually influencing the system's dynamics.

The proposed methodology has been introduced in the domain of deep RC networks, where its effectiveness has been shown on three datasets, covering both cases of artificial as well as real-world input signals. Moreover, the proposed tool allowed us to study the inherent role played by the layering factor in recurrent models. In particular, we have shown that increasing the number of layers in a recurrent architecture naturally leads the resulting systems dynamics towards progressively less stable and richer regimes. We have provided experimental evidence suggesting that the same amount of recurrent units has a richer dynamical behavior that is pushed closer to the edge of stability whenever the units are arranged into a hierarchy of layers. Results on the MC task pointed out that this phenomenon has interesting implications in cases in which the temporal task under consideration is better approached by a recurrent model operating close to the border of stability. Specifically, compared to shallow recurrent architectures, hierarchically organized recurrent models required a smaller number of recurrent units to achieve similar memory lengths, at the same time showing a higher peak of memory under analogous settings.

We hope that the approach developed in this paper would help to enhance the understanding on the theoretical properties of the dynamical behaviors developed by deep RNNs. At the same time, we believe that the proposed methodology represents a useful ground for a principled exploitation of the intrinsic potentiality of hierarchical recurrent models.

Appendix A. Proof of Lemma 1

case $i < j$:
This case corresponds to the higher triangular block matrices in $\mathbf{J}_{F,\mathbf{x}_g}(t)$. Based on the definition of the state transition function of individual layers in the deepESN, in eq. 5, it is straightforward to see that such block matrices are all null, as the state of each layer i in the architecture does not depend on the state of higher layers $j > i$.

case $i = j$:
This case corresponds to the diagonal block matrices in $\mathbf{J}_{F,\mathbf{x}_g}(t)$, expressing the dependence between two consecutive states at the same layer. In the following computations we assume that $i > 1$, whereas the case $i = 1$ is

obtained in the same fashion. We have:

$$
\begin{aligned}
&\mathbf{J}_{F^{(i)},\mathbf{x}^{(i)}}(t) = \tfrac{\partial F^{(i)}(\mathbf{u}(t),\mathbf{x}^{(1)}(t-1),\mathbf{x}^{(2)}(t-1),\ldots,\mathbf{x}^{(i)}(t-1))}{\partial \mathbf{x}^{(i)}(t-1)} = \\
&\tfrac{\partial}{\mathbf{x}^{(i)}(t-1)}\big((1-a^{(i)})\mathbf{x}^{(i)}(t-1) + a^{(i)}\tanh(\mathbf{W}_{il}^{(i)}F^{(i-1)}(\mathbf{u}(t),\mathbf{x}^{(1)}(t-1), \\
&\mathbf{x}^{(2)}(t-1),\ldots,\mathbf{x}^{(i-1)}(t-1)) + \boldsymbol{\theta}^{(i)} + \hat{\mathbf{W}}^{(i)}\mathbf{x}^{(i)}(t-1))\big) = \\
&(1-a^{(i)})\mathbf{I} + a^{(i)}\ \tfrac{\partial}{\mathbf{x}^{(i)}(t-1)}\big(\tanh(\mathbf{W}_{il}^{(i)}F^{(i-1)}(\mathbf{u}(t),\mathbf{x}^{(1)}(t-1), \\
&\mathbf{x}^{(2)}(t-1),\ldots,\mathbf{x}^{(i-1)}(t-1)) + \boldsymbol{\theta}^{(i)} + \hat{\mathbf{W}}^{(i)}\mathbf{x}^{(i)}(t-1))\big) = \\
&(1-a^{(i)})\mathbf{I} + a^{(i)}\mathbf{D}^{(i)}(t)\hat{\mathbf{W}}^{(i)},
\end{aligned}
\tag{A.1}
$$

where $\mathbf{D}^{(i)}(t)$ is a diagonal matrix whose non zero elements are the derivative of the activation of the reservoir units in the i-th layer computed by neglecting the leaky dynamics, which in the case of tanh non-linearity is as follows:

$$
\mathbf{D}^{(i)}(t) = \begin{pmatrix}
1-(\tilde{x}_1^{(i)}(t))^2 & 0 & \ldots & 0 \\
0 & 1-(\tilde{x}_2^{(i)}(t))^2 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & 1-(\tilde{x}_{N_R}^{(i)}(t))^2
\end{pmatrix}
$$

and where for $j = 1, 2, \ldots, N_R$, $\tilde{x}_j^{(i)}(t)$ are the elements of the vector $\tilde{\mathbf{x}}^{(i)}(t) = \tanh(\mathbf{W}_{il}^{(i)}F^{(i-1)}(\mathbf{u}(t),\mathbf{x}^{(1)}(t-1),\mathbf{x}^{(2)}(t-1),\ldots\mathbf{x}^{(i-1)}(t-1)) + \boldsymbol{\theta}^{(i)} + \hat{\mathbf{W}}^{(i)}\mathbf{x}^{(i)}(t-1))$.

case $i > j$:
This case corresponds to the block matrices below the diagonal of $\mathbf{J}_{F,\mathbf{x}_g}(t)$, expressing the dependence between the state of the layer i at time step t and the state of the lower layer j in the architecture at the previous time step.

In this case we have:

$$\mathbf{J}_{F^{(i)},\mathbf{x}^{(j)}}(t) = \frac{\partial F^{(i)}(\mathbf{u}(t),\mathbf{x}^{(1)}(t-1),\mathbf{x}^{(2)}(t-1),\ldots,\mathbf{x}^{(i)}(t-1))}{\partial \mathbf{x}^{(j)}(t-1)} =$$

$$\frac{\partial}{\mathbf{x}^{(j)}(t-1)}\big((1-a^{(i)})\mathbf{x}^{(i)}(t-1) + a^{(i)}\tanh(\mathbf{W}_{il}^{(i)}F^{(i-1)}(\mathbf{u}(t),\mathbf{x}^{(1)}(t-1),$$
$$\mathbf{x}^{(2)}(t-1),\ldots,\mathbf{x}^{(i-1)}(t-1)) + \boldsymbol{\theta}^{(i)} + \hat{\mathbf{W}}^{(i)}\mathbf{x}^{(i)}(t-1))\big) =$$

$$a^{(i)}\frac{\partial}{\mathbf{x}^{(j)}(t-1)}\big(\tanh(\mathbf{W}_{il}^{(i)}F^{(i-1)}(\mathbf{u}(t),\mathbf{x}^{(1)}(t-1),\mathbf{x}^{(2)}(t-1),\ldots,$$
$$\mathbf{x}^{(i-1)}(t-1)) + \boldsymbol{\theta}^{(i)} + \hat{\mathbf{W}}^{(i)}\mathbf{x}^{(i)}(t-1))\big) =$$

$$a^{(i)}\mathbf{D}^{(i)}(t)\mathbf{W}_{il}^{(i)}\mathbf{J}_{F^{(i-1)},\mathbf{x}^{(j)}}(t), \tag{A.2}$$

where, solving the recursion in the last line of eq. A.2, we get

$$\mathbf{J}_{F^{(i)},\mathbf{x}^{(j)}}(t) =$$

$$\big(\textstyle\prod_{k=0}^{i-(j+1)} a^{(i-k)}\mathbf{D}^{(i-k)}(t)\mathbf{W}_{il}^{(i-k)}\big)\mathbf{J}_{F^{(j)},\mathbf{x}^{(j)}}(t) = \tag{A.3}$$

$$\big(\textstyle\prod_{k=0}^{i-(j+1)} a^{(i-k)}\mathbf{D}^{(i-k)}(t)\mathbf{W}_{il}^{(i-k)}\big)\big[(1-a^{(j)})\mathbf{I} + a^{(j)}\mathbf{D}^{(j)}(t)\hat{\mathbf{W}}^{(j)}\big].$$

Appendix B. Proof of Lemma 2

The result of Lemma 2 follows from elementary matrix theory. In the following we describe a simple way of seeing it. We first recall that the set of eigenvalues of a matrix is given by the roots of its characteristic polynomial. In our case we have that the eigenvalues of $\mathbf{J}_{F,\mathbf{x}_g}(t)$ are the solutions of

$$det(\mathbf{J}_{F,\mathbf{x}_g}(t) - \lambda\mathbf{I}) = 0, \tag{B.1}$$

where $det(\cdot)$ denotes the determinant of its matrix argument. We can then recall that the determinant of a lower triangular block matrix is given by the product of the determinants of its diagonal blocks. In light of this we can rewrite eq. B.1 as

$$\prod_{i=1}^{N_L} det(\mathbf{J}_{F^{(i)},\mathbf{x}^{(i)}}(t) - \lambda\mathbf{I}) = 0, \tag{B.2}$$

from which we can conclude that the set of eigenvalues of $\mathbf{J}_{F,\mathbf{x}_g}(t)$ is given by the set of eigenvalues of its block diagonal matrices, i.e. $\text{eig}(\mathbf{J}_{F,\mathbf{x}_g}(t)) = \{\text{eig}(\mathbf{J}_{F^{(i)},\mathbf{x}^{(i)}}(t))|i = 1,2,\ldots,N_L\}$.

Appendix C. Numerical Simulations on Real-world Cases

Here we show the application of the proposed tool for the stability analysis of deep recurrent models on real-world cases. Specifically, we report the results of the computation of the maximum local Lyapunov exponent of deep-ESN states, analyzing the impact of varying the network's hyper-parameters and the number of layers in the architecture.

As driving input we have considered temporal signals from two real-world datasets. The first is the Sunspot dataset from `http://www.sidc.be/silso/home`, which provides updated sunspot numbers aggregated in different ways. This dataset has been used, under a predictive setting, in previous works on RC-based networks, e.g. in [49, 50, 51]. In our experiments we considered daily sunspot numbers from October 23 2003 to June 30 2017, for a total number of 5000 time-steps. The second dataset, referred to as the Electricity dataset, contains the hourly single national price of electricity on the Italian market and is online available at `http://www.mercatoelettrico.org/en/download/DatiStorici.aspx`. The work in [52] reported an excellent performance of an ad-hoc RC model in a 24 hours-ahead prediction settings on this data. Here, for our purposes, we considered the first 5000 time-steps of the hourly data pertaining to the year 2013. For both datasets, the original input data were divided by 100 to get signals with similar amplitudes as in the case of artificial data in Section 4, at the same time avoiding trivial saturation effects on the reservoir units in the first layer. All other aspects of the experimental settings adopted were as reported in Section 4 for the case of the artificial input data sampled from a uniform distribution. Results achieved on the Sunspot and on the Electricity datasets are reported in Fig. C.7 and C.8, respectively.

As already observed in Section 4, inspection of the plots in Fig. C.7 and C.8 reveals conditions of network's settings characterized by state dynamics with different regimes. In particular, results show that higher values of λ_{max} are obtained when a progressively deeper architecture is considered, and for increasing values of ρ. As regards the other hyper-parameters we can see that they have a more limited impact on the dynamical regimes than ρ. Specifically, decreasing values of a, $scale_{in}$ and $scale_{il}$ lead to increasing values of λ_{max}, with a trend that is more pronounced for deeper networks.

Overall, results in Fig. C.7 and C.8 generally confirm the qualitative analysis already discussed in Section 4 with regard to Fig. 2, though with few differences on the quantitative side (i.e. on the exact values of λ_{max}).

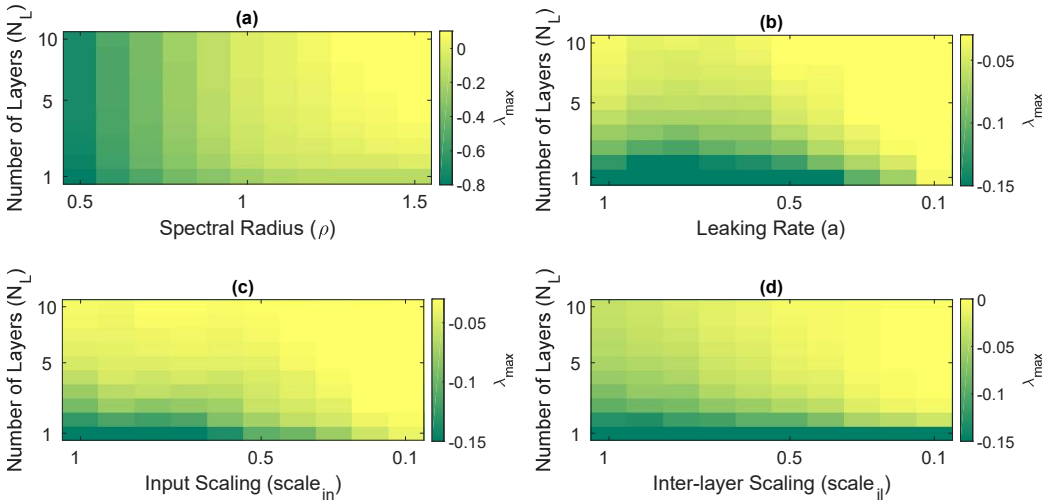

Figure C.7: Averaged values of λ_{max} obtained by deepESN on the Sunspot dataset for increasing number of reservoir layers. **(a)**: Increasing values of ρ, $a = 1$, $scale_{in} = 1$, $scale_{il} = 1$. **(b)**: Decreasing values of a, $\rho = 1$, $scale_{in} = 1$, $scale_{il} = 1$. **(c)**: Decreasing values of $scale_{in}$, $\rho = 1$, $a = 1$, $scale_{il} = 1$. **(d)**: Decreasing values of $scale_{il}$, $\rho = 1$, $a = 1$, $scale_{in} = 1$.

References

[1] P. Angelov, A. Sperduti, Challenges in deep learning, in: Proceedings of the 24th European Symposium on Artificial Neural Networks (ESANN), i6doc.com, 2016, pp. 489–495.

[2] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT press, 2016.

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

[4] M. Hermans, B. Schrauwen, Training and analysing deep recurrent neural networks, in: NIPS, 2013, pp. 190–198.

[5] A. Graves, A.-R. Mohamed, G. Hinton, Speech recognition with deep recurrent neural networks, in: IEEE International Conference on Acoustics, speech and signal processing (ICASSP), IEEE, 2013, pp. 6645–6649.

[6] S. El Hihi, Y. Bengio, Hierarchical recurrent neural networks for long-term dependencies, in: Advances in neural information processing systems (NIPS), 1996, pp. 493–499.

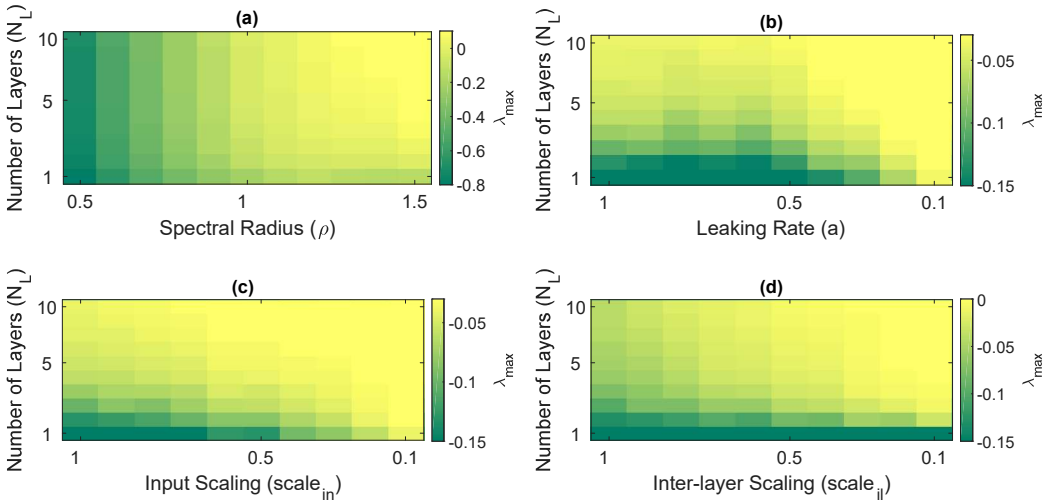

Figure C.8: Averaged values of λ_{max} obtained by deepESN on the Electricity dataset for increasing number of reservoir layers. **(a)**: Increasing values of ρ, $a = 1$, $scale_{in} = 1$, $scale_{il} = 1$. **(b)**: Decreasing values of a, $\rho = 1$, $scale_{in} = 1$, $scale_{il} = 1$. **(c)**: Decreasing values of $scale_{in}$, $\rho = 1$, $a = 1$, $scale_{il} = 1$. **(d)**: Decreasing values of $scale_{il}$, $\rho = 1$, $a = 1$, $scale_{in} = 1$.

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

[8] J. Schmidhuber, Learning complex, extended sequences using the principle of history compression, Neural Computation 4 (2) (1992) 234–242.

[9] W. Gerstner, W. Kistler, Spiking neuron models: Single neurons, populations, plasticity, Cambridge university press, 2002.

[10] P. S. Churchland, T. J. Sejnowski, The computational brain, MIT press, 1992.

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

[12] D. Verstraeten, B. Schrauwen, M. d'Haene, D. Stroobandt, An experimental unification of reservoir computing methods, Neural networks 20 (3) (2007) 391–403.

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

[14] H. Jaeger, The "echo state" approach to analysing and training recurrent neural networks - with an erratum note, Tech. rep., GMD - German National Research Institute for Computer Science (2001).

[15] C. Gallicchio, A. Micheli, L. Pedrelli, Deep reservoir computing: A critical experimental analysis, Neurocomputing 268 (2017) 87–99. doi:https://doi.org/10.1016/j.neucom.2016.12.089.

[16] C. Gallicchio, A. Micheli, Deep reservoir computing: A critical analysis, in: Proceedings of the 24th European Symposium on Artificial Neural Networks (ESANN), i6doc.com, 2016, pp. 497–502.

[17] I. Yildiz, H. Jaeger, S. Kiebel, Re-visiting the echo state property, Neural networks 35 (2012) 1–9.

[18] C. Gallicchio, A. Micheli, Echo state property of deep reservoir computing networks., Cognitive Computation 9 (3) (2017) 337–350.

[19] B. Bailey, Local lyapunov exponents: predictability depends on where you are, Nonlinear Dynamics and Economics, Cambridge University Press (1996) 345–359.

[20] H. D. I. Abarbanel, R. Brown, M. Kennel, Local lyapunov exponents computed from observed data, Journal of Nonlinear Science 2 (3) (1992) 343–365. doi:10.1007/BF01208929.

[21] D. Verstraeten, J. Dambre, X. Dutoit, B. Schrauwen, Memory versus non-linearity in reservoirs, in: Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN), IEEE, 2010, pp. 1–8.

[22] D. Verstraeten, B. Schrauwen, On the quantification of dynamics in reservoir computing, in: International Conference on Artificial Neural Networks, Springer, 2009, pp. 985–994.

[23] R. Legenstein, W. Maass, What makes a dynamical system computationally powerful, New directions in statistical signal processing: From systems to brain (2007) 127–154.

[24] R. Legenstein, W. Maass, Edge of chaos and prediction of computational performance for neural circuit models, Neural Networks 20 (3) (2007) 323 – 334. doi:https://doi.org/10.1016/j.neunet.2007.04.017.

[25] N. Bertschinger, T. Natschläger, Real-time computation at the edge of chaos in recurrent neural networks, Neural computation 16 (7) (2004) 1413–1436.

[26] L. Büsing, B. Schrauwen, R. Legenstein, Connectivity, dynamics, and memory in reservoir computing with binary and analog neurons, Neural computation 22 (5) (2010) 1272–1311.

[27] I. Farkaš, R. Bosák, P. Gergel', Computational analysis of memory capacity in echo state networks, Neural Networks 83 (2016) 109–120.

[28] P. Barančok, I. I. Farkaš, Memory capacity of input-driven echo state networks at the edge of chaos, in: International Conference on Artificial Neural Networks, Springer, 2014, pp. 41–48.

[29] F. Bianchi, L. Livi, R. Jenssen, C. Alippi, Critical echo state network dynamics by means of fisher information maximization, in: International Joint Conference on Neural Networks (IJCNN), IEEE, 2017, pp. 852–858.

[30] J. Boedecker, O. Obst, J. Lizier, N. Mayer, M. Asada, Information processing in echo state networks at the edge of chaos, Theory in Biosciences 131 (3) (2012) 205–213.

[31] B. Schrauwen, L. Büsing, R. A. Legenstein, On computational power and the order-chaos phase transition in reservoir computing, in: Advances in Neural Information Processing Systems, 2009, pp. 1425–1432.

[32] L. Livi, F. M. Bianchi, C. Alippi, Determination of the edge of criticality in echo state networks through fisher information maximization, IEEE Transactions on Neural Networks and Learning Systems PP (99) (2017) 1–12. doi:10.1109/TNNLS.2016.2644268.

[33] H. Jaeger, Short term memory in echo state networks, Tech. rep., German National Research Center for Information Technology (2001).

[34] C. Gallicchio, J. D. Martin-Guerrero, A. Micheli, E. Soria-Olivas, Randomized machine learning approaches: Recent developments and challenges, in: Proceedings of the 25th European Symposium on Artificial Neural Networks (ESANN), i6doc.com, 2017, pp. 77–86.

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

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

[37] G. Manjunath, H. Jaeger, Echo state property linked to an input: Exploring a fundamental characteristic of recurrent neural networks, Neural computation 25 (3) (2013) 671–696.

[38] M. Buehner, P. Young, A tighter bound for the echo state property, IEEE Transactions on Neural Networks 17 (3) (2006) 820–824.

[39] L. N. Trefethen, D. Bau III, Numerical linear algebra, Siam, 1997.

[40] G. Wainrib, M. N. Galtier, A local echo state property through the largest lyapunov exponent, Neural Networks 76 (2016) 39–45.

[41] S. Jarvis, S. Rotter, U. Egert, Extending stability through hierarchical clusters in echo state networks, Frontiers in neuroinformatics 4 (2010) 1–11.

[42] J. Sprott, Chaos and time-series analysis, Oxford University Press Oxford, 2003.

[43] H. Broer, F. Takens, Dynamical systems and chaos, Vol. 172, Springer Science & Business Media, 2010.

[44] J.-P. Eckmann, D. Ruelle, Ergodic theory of chaos and strange attractors, Reviews of modern physics 57 (3) (1985) 617.

[45] F. M. Bianchi, L. Livi, C. Alippi, R. Jenssen, Multiplex visibility graphs to investigate recurrent neural network dynamics, Scientific Reports 7.

[46] F. Bianchi, L. Livi, C. Alippi, Investigating echo-state networks dynamics by means of recurrence analysis, IEEE Transactions on Neural Networks and Learning Systems PP (99) (2016) 1–13.

[47] T. Yamane, S. Takeda, D. Nakano, G. Tanaka, R. Nakane, S. Nakagawa, A. Hirose, Dynamics of reservoir computing at the edge of stability, in: International Conference on Neural Information Processing, Springer, 2016, pp. 205–212.

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

[49] A. Rodan, P. Tino, Minimum complexity echo state network, IEEE transactions on neural networks 22 (1) (2011) 131–144.

[50] M.-H. Yusoff, J. Chrol-Cannon, Y. Jin, Modeling neural plasticity in echo state networks for classification and regression, Information Sciences 364 (2016) 184–196.

[51] X. Sun, T. Li, Q. Li, Y. Huang, Y. Li, Deep belief echo-state network and its application to time series prediction, Knowledge-Based Systems.

[52] E. Crisostomi, C. Gallicchio, A. Micheli, M. Raugi, M. Tucci, Prediction of the italian electricity price for smart grid applications, Neurocomputing 170 (2015) 286–295.