Neural Computing and Applications manuscript No.
(will be inserted by the editor)

Architectural Richness in Deep Reservoir Computing

Claudio Gallicchio · Alessio Micheli

Received: date / Accepted: date

Abstract Reservoir Computing (RC) is a popular class of Recurrent Neural Networks (RNNs) with untrained dynamics. Recently, advancements on deep RC architectures have shown a great impact in time-series applications, showing a convenient trade-off between predictive performance and required training complexity. In this paper, we go more in depth into the analysis of untrained RNNs by studying the quality of recurrent dynamics developed by the layers of deep RC neural networks. We do so by assessing the richness of the neural representations in the different levels of the architecture, using measures originating from the fields of dynamical systems, numerical analysis and information theory. Our experiments, on both synthetic and real-world datasets, show that depth - as an architectural factor of RNNs design - has a natural effect on the quality of RNN dynamics (even without learning of the internal connections). The interplay between depth and the values of RC scaling hyper-parameters, especially the scaling of inter-layer connections, is crucial to design rich untrained recurrent neural systems.

Keywords Reservoir Computing · Deep Reservoir Computing · Deep Recurrent Neural Networks · Richness of Neural Networks Dynamics

1 Introduction

Recurrent Neural Networks (RNNs) represent a consolidated computational abstraction for learning with variable-length times-series data. Their extensions towards deep architectural settings, i.e. deep RNNs, are of special interest as they enable the development of richer and multi-scale internal representations of the dynamical input information in an effective fashion, with

Claudio Gallicchio (✉) · Alessio Micheli
Department of Computer Science, University of Pisa
Largo B. Pontecorvo 3, Pisa, Italy.
E-mail: {gallicch,micheli}@di.unipi.it

breakthrough applications in relevant domains such as language, document and speech processing [21,35,23]. However, a major downside of traditional approaches is that training (deep) RNNs, typically by means of gradient back-propagation through time [43] algorithms, is difficult [3] and involves high computational requirements. Hence, whenever accuracy is not the only constraint in place, finding efficient alternatives to standard RNN training becomes of paramount importance (e.g., in distributed intelligence applications embedded on low-powerful devices).

In this context, randomization-based deep neural networks algorithms provide a useful trade-off between complexity and accuracy by exploiting as much as possible the architectural biases of deep neural networks models [20,39]. The fundamental idea is that of leaving the hidden layers of the deep architecture untrained after random - but constrained - initialization, thereby limiting the learning algorithms to operate only on an output readout component. The resulting striking advantage is that the training algorithms are much simpler and faster, and can even be implemented at the edge of a distributed computing setup [2]. Reservoir Computing (RC) [32,41] is the reference paradigm for randomization-based RNN models. In this sense, Echo State Networks [25,28] represent the most notable instance of the RC paradigm. The crucial insight is that it is possible to separate the treatment of the dynamical component of the network, called the *reservoir*, from that of the *readout* part. The dynamical reservoir can be left untrained after random initialization provided that it implements an asymptotically stable input-driven dynamical system [25,46].

In particular, the recently introduced study of Deep RC systems [17,18,13], comprising a stacked composition of multiple untrained reservoirs, is a research line of special interest as it allows to combine the expressiveness of deep architectures for time-series with the efficiency of RC-based training algorithms. The approach has proven over the past few years to be extremely effective in addressing real-world problems in several application domains, including industrial plants monitoring [7,5], speech and polyphonic music processing [19], meteorological forecasting [31], energy consumption and wind power generation prediction [24]. Beyond the practical merits, the analysis of the properties of Deep RC models is also important because it enables to characterize the intrinsic behavior of deep recurrent systems prior to, or in the absence of, learning of the recurrent connections. Relevantly, studies on Deep RC showed that a stacked composition of recurrent hidden layers inherently produces a qualitative diversification of the dynamical response of the neurons in the successive layers of the architecture [18,19]. However, despite the great recent interest in this class of neural network models, the effects of architectural choices on the quality of the generated dynamics still remain unexplored in the literature.

To fill this gap, in this paper we go more in-depth into the analysis of the dynamics developed by the hidden layers of Deep RC neural networks. Specifically, we ask whether (and under what circumstances) a deep setting of an RC architecture is able to unleash a beneficial process of enriching state dynamics. We address this research question by performing an empirical evaluation that uses several metrics for quantifying the concepts of 'richness' and 'goodness' of

deep reservoirs. The considered metrics cover heterogeneous aspects, including dynamical stability, diversity and intrinsic dimensionality of the reservoir manifold, as well as the conditioning of the resulting learning problem for the readout. The experiments reported in this paper are conducted on several datasets with different properties, including popular RC benchmarks as well as real-world time-series. As our investigation aims at analyzing the intrinsic quality of state dynamics in progressively deeper layers of an RC-system, we follow an unsupervised approach and take aside all the aspects related to training the readout based on specific target output signals. Overall, this work intends to shed light on the intrinsic effects of layering in RNNs, and on the interplay with the values of fundamental RC-based scaling hyper-parameters.

The rest of this paper is organized as follows. We start by introducing the Deep Reservoir Computing paradigm in Section 2. After that, in Section 3, we address the richness of reservoir dynamics, presenting the metrics used to quantify it. The experimental analysis is reported in Section 4 (with additional results given in Appendix A). Finally, Section 5 concludes the paper.

2 Deep Reservoir Computing Neural Networks

A Deep Reservoir Computing (Deep RC) [17,18,20] neural network is made by a stacked composition of multiple hidden layers of sparsely connected recurrent neurons that are left untrained after initialization. Although this concept can be in principle instantiated to both continuous and discrete time dynamics, here we refer to the latter, and use the formalism introduced by the Deep Echo State Network (DeepESN) model in [18]. In a deep reservoir system, the time varying input signal is applied only to the first recurrent layer. Each successive reservoir layer is driven by the dynamics generated by the previous reservoir in the stack. The overall Deep RC architecture is illustrated in Fig. 1.

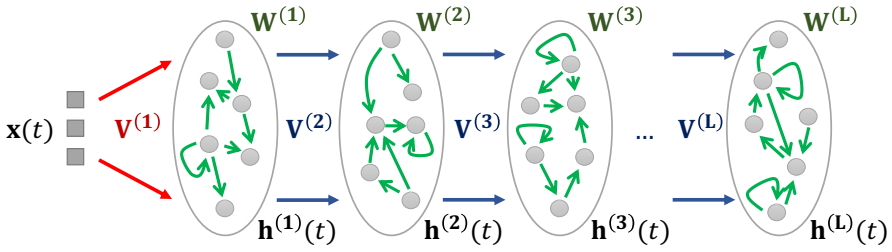


Fig. 1: Architecture of a Deep RC neural network.

In mathematical terms, at each time-step t an X-dimensional input feature denoted by $\mathbf{x}(t)$ is fed to the first reservoir layer, which filters it according to

the following state transition function:

$$\mathbf{h}^{(1)}(t) = \tanh\big(\mathbf{V}^{(1)}\mathbf{x}(t) + \mathbf{W}^{(1)}\mathbf{h}^{(1)}(t-1)\big), \tag{1}$$

where $\mathbf{h}^{(1)}(t)$ is the $H^{(1)}$-dimensional reservoir state of the first layer at time-step t, $\mathbf{V}^{(1)}$ and $\mathbf{W}^{(1)}$ are respectively the input and the recurrent weight matrix for the first layer, and tanh denotes the element-wise applied hyperbolic tangent non-linearity. Following the pipeline of computation, each successive i-th layer (with $i > 1$) receives as input at time-step t the state of the previous reservoir at the same time-step, and implements a state transition function described as follows:

$$\mathbf{h}^{(i)}(t) = \tanh\big(\mathbf{V}^{(i)}\mathbf{h}^{(i-1)}(t) + \mathbf{W}^{(i)}\mathbf{h}^{(i)}(t-1)\big), \tag{2}$$

where $\mathbf{h}^{(i)}(t)$ is the $H^{(i)}$-dimensional state of the i-th reservoir layer, $\mathbf{V}^{(i)}$ is the inter-layer weight matrix that modulates the information transmission between layer $i-1$ and layer i, and tanh is the non-linearity as in the previous case. Notice that in both equations 1 and 2 we omit the bias terms for the ease of notation. Furthermore, a more general definition pertaining to leaky integration reservoir neurons [29] can be found, e.g., in [18]. The formulation in equations 1 and 2 is complemented by a set of initial reservoir conditions, i.e., $\mathbf{h}^{(1)}(0), \ldots, \mathbf{h}^{(L)}(0)$, all of which are typically set to a null vector $\mathbf{0}$. Also note that, while in general the number of neurons in the different layers of the deep reservoir architecture can be different, for the sake of simplicity in this paper we focus on a setup in which all reservoir layers contain the same number of neurons, indicated by H.

Crucially, all the weight matrices $\mathbf{V}^{(1)}, \ldots, \mathbf{V}^{(L)}$, $\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L)}$, involved in the deep reservoir computation are left untrained after initialization. This initialization needs to consider constraints related to asymptotic stability of the resulting dynamical system, a condition that in this context is known under the name of Echo State Property (ESP) [25,46]. The interested reader can find a detailed analysis of the stability conditions for Deep RC neural networks in [14]. The essential outcome of such analysis is that it is possible to use randomized and untrained weights in the reservoirs, provided that the magnitude of the weights is kept under control and properly scaled. As such, to initialize the reservoirs in a Deep RC system, we randomly draw all the weight values from a uniform distribution in $[-1, 1]$, and then:

- re-scale the input weight matrix $\mathbf{V}^{(1)}$ to have a maximum singular value equal to an *input scaling* ω_{in};
- for each layer $i > 1$, re-scale the inter-layer weight matrix $\mathbf{V}^{(i)}$ to have a maximum singular value equal to an *inter scaling* $\omega_{il}^{(i)}$;
- for each layer $i \geq 1$, re-scale the recurrent weight matrix $\mathbf{W}^{(i)}$ to have a *spectral radius*[1] equal to $\rho^{(i)}$.

[1] The maximum among the eigenvalues in modulus.

Controlling the values of the spectral radii of the reservoirs in the deep architecture is particularly important. Commonly, values smaller than, but close to, 1 are considered as those giving the best trade-off between expressivity and stability [32, 25].

In common setups, the Deep RC neural network also comprises a readout output layer that performs the output computation being fed by reservoir states, and that is the only trained component of the architecture [18]. In this work, we focus explicitly on studying the quality of the neural representations developed in the reservoir, and all aspects related to the readout part of the network is left aside of the current study[2].

3 Richness of Reservoir Dynamics

Measuring the quality of reservoir systems' dynamics is a matter involving analysis under several perspectives. While there is not a unique commonly adopted measure of reservoirs richness, there are a number of desired properties that a reservoir should possess. Stability, diversity and high intrinsic dimension of internal state dynamics, in addition to a "good" conditioning of the resulting training problem (in the reservoir space) are among the desirable properties that a randomized recurrent system should have, and set the ground for the experimental analysis in this paper. Addressing the above-mentioned properties, we specifically quantify the richness of reservoir dynamics by using the measure described in the following sub-sections. Notice that, in introducing such measures, we refer to the case in which the deep reservoir is driven by a unique long input time-series and we are interested in assessing the quality of the transient state dynamics at each time-step. This setup can be straightforwardly extended to the case in which we have many input time-series and we are interested in evaluating the richness of the final reservoir representation developed in the last time-step of each time-series.

Specifically, to measure the richness and goodness of reservoir dynamics we resort to the following: ESP_{index} (described in Section 3.1), Average State Entropy of reservoir neurons activations (described in Section 3.2, Linearly Uncoupled Dynamics (described in Section 3.3), and Condition Number (described in Section 3.4).

3.1 ESP index

Studying the dynamical behavior of a reservoir layer under the effect of the external driving input is of paramount practical importance. In particular, studied as a non-autonomous dynamical system, each reservoir layer should be asymptotically *stable*, i.e. robust to perturbations in the initial conditions (equivalently, in the input [25]). This means that, after a transient period, the

[2] The interested reader is referred to [17, 13] for a more complete overview of the general Deep RC approach, including the readout computation and processing.

trajectory of the reservoir state should be uniquely determined by the driving input. In practice, if the system is started with several different initial conditions and receives the same input time-series, after a transient period all the possible trajectories in the reservoir state space synchronize (to a unique attractive trajectory). This idea has been used in [10] to develop a practical measure of reservoir stability, called the *ESP index*, and consisting in computing the average deviation among reservoir state trajectories obtained starting from different initial conditions under the influence of the same input signal.

Specifically, referring to the dynamics in the i-th reservoir layer, and given in input the time-series $\mathbf{x}(1), \ldots, \mathbf{x}(S)$, we first compute - as a reference trajectory - the states obtained by starting the reservoir from the $\mathbf{0}$ state, i.e. $\mathbf{h}_0^{(i)}(1), \ldots, \mathbf{h}_0^{(i)}(S)$. We then pick P randomly chosen initial conditions and, using the same driving input, we compute the correspondingly achieved reservoir states, i.e. $\{\mathbf{h}_j^{(i)}(1) \ , \ldots, \ \mathbf{h}_j^{(i)}(S)\}_{j=1}^{P}$. After discarding an initial transient of duration T, we can compute the average deviation between $\mathbf{h}_0^{(i)}(t)$ and $\mathbf{h}_j^{(i)}(t)$ as follows:

$$ESP_j^{(i)} = \frac{1}{S-T} \sum_{t=T+1}^{S} \|\mathbf{h}_0^{(i)}(t) - \mathbf{h}_j^{(i)}(t)\|_2. \tag{3}$$

We finally compute the ESP_{index} by averaging the ESP_j values in equation 3, as follows:

$$ESP_{index}^{(i)} = \frac{1}{P} \sum_{j=1}^{P} ESP_j^{(i)}. \tag{4}$$

In applications, one can then use the $ESP_{index}^{(i)}$ computed as in the above equation 4 to verify that a reservoir layer under consideration is effectively in an asymptotically stable regime. In practice, whenever $ESP_{index}^{(i)} = 0$ we can claim that the ESP is practically satisfied by the reservoir under the influence of the given input signal. On the other hand, whenever $ESP_{index}^{(i)}$ takes values > 0 the reservoir is sensible to perturbations and the ESP is not met for the given input signal.

We find it useful to remark here that the case in which $ESP_{index} > 0$ is potentially harmful for practical applications, as it implies some form of instability in the reservoir dynamics, which might ultimately result in poor generalization performance [22]. In this paper, then, we use the ESP_{index} to measure the "goodness" of a specific reservoir system under a given input signal. Accordingly, the reservoir is considered "good" if $ESP_{index} = 0$, and "bad" otherwise.

3.2 Average State Entropy (ASE)

A popular downside of RC-based approaches is that the random and untrained settings of the reservoir connections (under stability constraints) concretely

results into a limitation of the variety and expressiveness of the information that is conveyed by the reservoir state. Intuitively, the components of the system's state should be as diverse as possible to provide a richer pool of dynamics from which the trainable part can appropriately combine. From an information-theoretic point of view, we can measure this form of richness by means of the entropy of instantaneous reservoir states. In particular, to this purpose we follow the work in [34] and adapt the use of the efficient estimator of Renyi's quadratic entropy in [37,36] to the case of Deep RC neural networks. Specifically, for each time-step t and for each layer i, we compute the value of instantaneous state entropy $\mathcal{H}^{(i)}(t)$ as follows:

$$\mathcal{H}^{(i)}(t) = -\log\big(\frac{1}{H^2}\sum_{j=1}^{H}(\sum_{k=1}^{H}\mathcal{K}(h_j^{(i)}(t)) - h_k^{(i)}(t))\big), \tag{5}$$

where $h_j^{(i)}(t))$ is the j-th component of $\mathbf{h}^{(i)}(t)$, and $\mathcal{K}$ indicates a Gaussian kernel[3]. Given an input time-series of length S, for each layer we average the values $\mathcal{H}^{(i)}(t)$ from the above equation 5, obtaining the Average State Entropy (ASE):

$$ASE^{(i)} = \frac{1}{S}\sum_{t=1}^{S}\mathcal{H}^{(i)}(t), \tag{6}$$

which we use as one of the metrics to quantify the "goodness" of deep reservoir layers, where higher values are preferable and denote richer dynamics.

3.3 Linearly Uncoupled Dynamics (LUD)

Another manifestation of the possibly limited variety of reservoir dynamics is typically observed in the high correlation among the reservoir state variables. In other words, reservoir's units exhibit behaviours that are strongly coupled among each other [32,45], i.e. the system state contains redundant components. This means that the reservoir state's trajectories are constrained into a state manifold whose dimension can be much smaller than the size of the reservoir. This phenomenon has been experimentally linked to the inherent Markovian organization of reservoir state spaces [40,12] resulting from contractive initialization [11]. The stronger is this coupling effect, the lower is the actual dimensionality of the reservoir manifold, and hence the poorer is the quality of the resulting reservoir dynamics. In this spirit, we propose to assess the richness of the reservoir by measuring the actual dimensionality of uncoupled state dynamics. Intuitively, the dimension of a reservoir state manifold can be thought of as the number of orthogonal directions in the reservoir state space. Hence, a simple way to compute the number of (linearly) uncoupled reservoir dynamics consists in performing PCA of the state activations, and

[3] The size of $\mathcal{K}$ is set to 0.3 times the std of instantaneous reservoir activations, as in [34].

then computing the number of principal components that explain the most of the variance in the sampled state space. Specifically, suppose to drive the dynamics of the i-th layer in a deep reservoir by an input time-series, and let us denote by $\mathbf{H}^{(i)}$ the column-wise concatenation of the resulting reservoir states, and by $\sigma_1^{(i)}, \ldots, \sigma_H^{(i)}$ the singular values of $\mathbf{H}^{(i)}$ in decreasing order. The normalized relevance of the j-th principal component of the sampled i-th reservoir space can be then computed as follows:

$$R_j^{(i)} = \frac{\sigma_j^{(i)}}{\sum_{k=1}^{H} \sigma_k^{(i)}}. \tag{7}$$

Intuitively, the larger is the value of $R_j^{(i)}$, the larger is the amount of variability of reservoir activations explained by the j-th principal component. Ideally, i.e. in a 'good' reservoir, the whole variability would be spread across many principal components rather than being concentrated on a few components. On this ground, we propose to evaluate the number of linearly uncoupled dynamics (LUD) as the number of principal components that are necessary to explain a desired amount of (normalized) reservoir space variability $\theta \in (0, 1]$. In particular we compute the following:

$$LUD^{(i)} = arg\ min_d \left\{ \sum_{k=1}^{d} R_k^{(i)} \geq \theta \right\}, \tag{8}$$

where we use the value of $\theta = 0.9$, meaning that we measure the number of linearly uncoupled directions in the reservoir state space that can explain at least the 90% of the whole observed variability. The higher is the value of $LUD^{(i)}$ the richer are considered to be the dynamics of the i-th reservoir layer.

3.4 Condition Number (C)

Ideally, the role of a reservoir layer should be to non-linearly project the driving input signal into a high dimensional feature space, where - based on the theoretical insights of the Cover's theorem [6] - the original learning problem is more likely to be solved linearly. It is then interesting to investigate the 'goodness' of the reservoir operation in terms of how much it actually facilitates the readout training phase. A way to quantify this is given by the condition number of the reservoir state matrix [32]. Specifically, using again $\mathbf{H}^{(i)}$ to denote the state matrix obtained the column-wise concatenation of the activations in the i-th reservoir layer, and $\sigma_1^{(i)}, \ldots, \sigma_H^{(i)}$ for its ordered singular values, we can calculate the condition number as follows:

$$C^{(i)} = \frac{\sigma_1^{(i)}}{\sigma_H^{(i)}}. \tag{9}$$

Higher values of $C^{(i)}$ indicate bad conditioning of the learning problem in the i-th reservoir layer state space. We therefore consider lower values of $C^{(i)}$ to be preferable, and characteristic of 'good' reservoirs [27].

4 Experimental Analysis

In this section we describe our experimental analysis on the richness of reservoir state dynamics in deeper layers of Deep RC neural networks. To this aim, we make use of the metrics of dynamical richness introduced in Section 3, empirically evaluated in different architectural configurations of the Deep RC system. Note that our analysis targets the intrinsic goodness of untrained (and input-driven) state dynamics in deep RNNs. Hence, our experiments follow an unsupervised setup, abstracting from the particular target output signal associated to the input time-series.

We introduce the used datasets in Section 4.1, while in Section 4.2 we report the details of the experimental conditions for our experiments. The results are provided and analyzed in Section 4.3.

4.1 Datasets

We considered a diverse pool of datasets to cover the cases of i) both uni-dimensional and multi-dimensional input features at each time-step, ii) both single-sequence (i.e., a unique long input time-series) and multiple-sequence (i.e., several input time-series) datasets, and iii) both synthetic and real-world time-series. The datasets are briefly illustrated in the following.

We first consider a RANDOM dataset, where the input at each time-step $\mathbf{x}(t)$ is a uni-dimensional variable that is sampled from a uniform distribution in $[0, 0.5]$. The generated time-series is 5000 time-steps long. Notice that this case encompasses a number of classical benchmarks in RC literature, from estimation of memory-capacity [26] to learning NARMA systems [1], where a specific target signal is coupled to the random input. Here, instead, abstracting from the possible supervised tasks, we are specifically interested in the assessment of the richness of reservoir dynamics under the influence of a randomized (and temporally unstructured) input signal. The dataset contains a single sequence and it is shown in Figure 2.

The second dataset contains a single time-series of sampled intensities from a far-infrared LASER, taken from the Santa Fe competition [42]. This dataset is representative for real-world time-series in chaotic regime and is a popular benchmark in the RC literature. We used the first 5000 time-steps of the dataset, to which we applied a minimal pre-processing consisting in a simple re-scaling by a factor of 0.01. The resulting LASER dataset is shown in Figure 3.

The third dataset is the electrocardiogram (ECG) dataset [33] from the UCR archive [4]. In this case there are multiple time-series, representing recordings of the cardiac electrical activity in normal and myocardic infarction conditions. We used the first 100 time-series in this dataset, where each of them

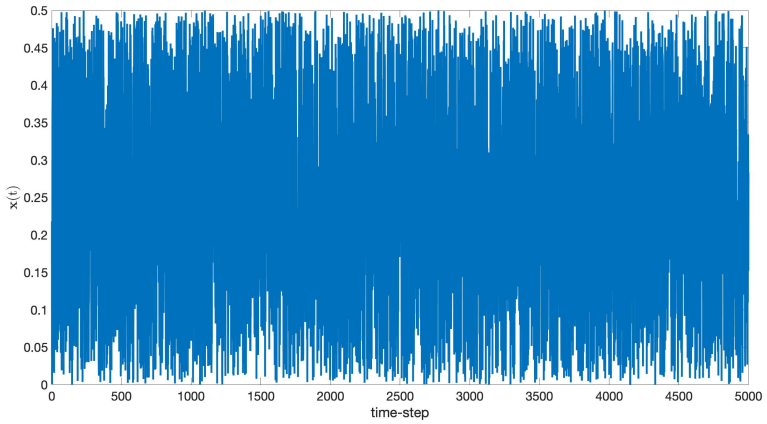


Fig. 2: Plot of the RANDOM dataset.

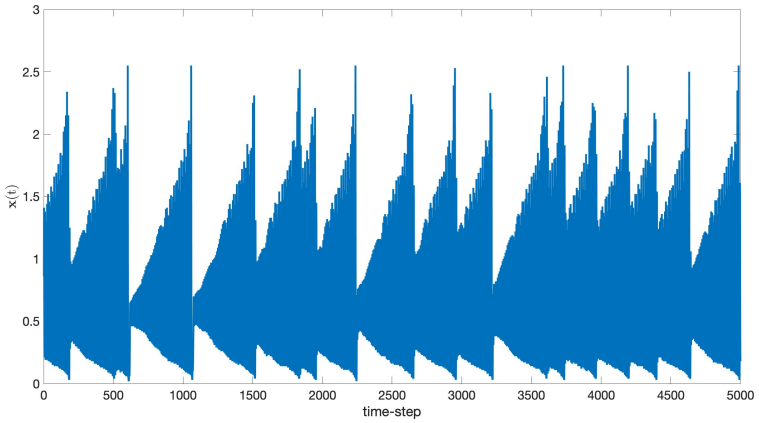


Fig. 3: Plot of the LASER dataset.

regards a single heartbeat, and has a length that varies between 54 and 147 time-steps. The input feature at each time-step $\mathbf{x}(t)$ is two-dimensional, and contains the information regarding leads 0 and 1 of the ECG recordings. Examples of the time-series in this dataset are shown in Figure 4.

The fourth dataset is the LIBRAS[4] dataset [8] from the UCI repository [9]. This dataset contains discrete representations of hand movements that represent specific types of language signs. Each time-series is obtained after pre-processing (with time-frame selection) of a recorded video during which the hand movement is performed, and the two-dimensional input feature at

[4] From *LÍngua BRAsileira de Sinais*, i.e. Brasilian Sign Language.

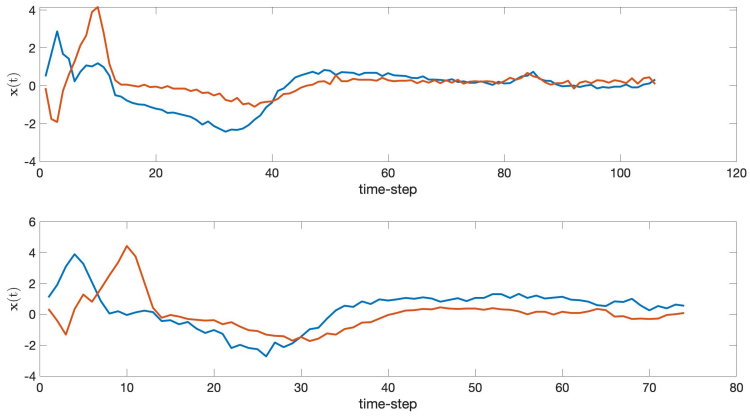


Fig. 4: Examples of time-series from the ECG dataset.

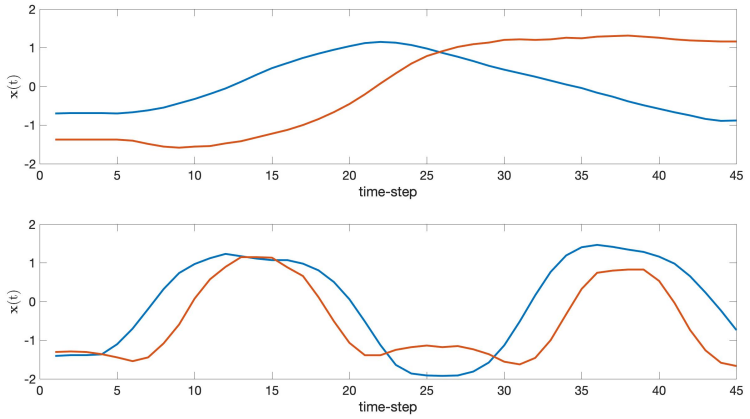


Fig. 5: Examples of time-series from the LIBRAS dataset.

each time-step is the coordinate vector representation of the hand in the video. Time-series are 45 time-steps long, and we used the first 180 time-series in our experiments. Instances from this dataset are given in Figure 5.

The last dataset we used is the Character Trajectories (CHAR) dataset [44] taken from the UCI repository [9]. The dataset contains data gathered from a WACOM tablet while a user is writing alphabet characters. Each time-series pertains to the writing of a single character, and the input information at each time-step represents a three-dimensional velocity trajectory of the pen tip during the writing. We used the first 300 sequences of the dataset, with lengths

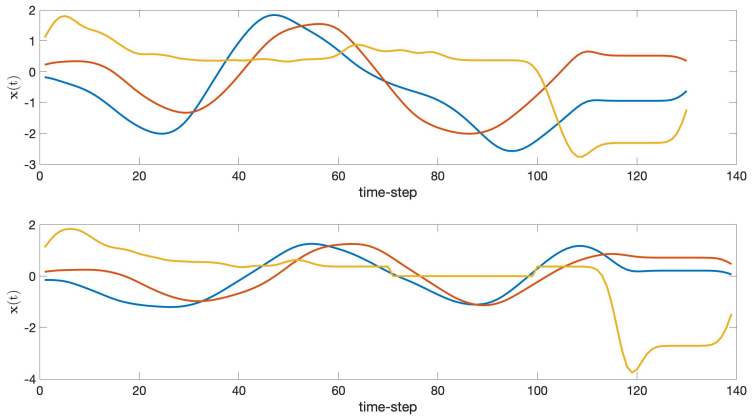


Fig. 6: Examples of time-series from the CHAR dataset.

varying between 63 and 172[5]. We report a couple of time-series examples from this dataset in Figure 6.

4.2 Experimental Settings

Following the model's description in Section 2, in our experiments we used a simple strategy to designing the Deep RC system, in which all the layers share the same values of the crucial hyper-parameters[6]. Specifically, all the layers shared the same value of the spectral radius ρ (i.e., $\rho^{(1)} = \rho^{(2)} = \ldots = \rho^{(L)}$), and all the layers at depth greater than one shared the same value of the inter layer scaling ω_{il} (i.e., $\omega_{il}^{(2)} = \ldots = \omega_{il}^{(L)}$). In our experiments, we explored Deep RC hyper-parametrizations varying ρ in $\{0.1, 0.3, \ldots, 1.3\}$, ω_{in} and ω_{il} in $\{0.1, 0.2, 0.5, 1, 2, 5, 10\}$. We considered networks with a maximum number of $L = 7$ layers, each of which containing $H = 100$ sparsely connected reservoir neurons, where each reservoir neuron receives one input connection from the previous layer (i.e., from the external input for the 1st layer, and from layer $i - 1$ for every layer $i > 1$) and one recurrent connection from the neurons in the same layer.

We assessed the richness of state dynamics developed in progressively deeper layers of a Deep RC system. To this end, we computed the metrics described in Section 3. In particular, we computed the ESP_{index} (see Section 3.1, where we set $P = 50$), the ASE entropy measure (see Section 3.2),

[5] The original dataset contained a variable number of time-steps with zero input features at the beginning and at the end of each time-series, which we have preliminary removed.

[6] Despite its simplicity, this choice for the architectural design of Deep RC was found useful in several application contexts (see, e.g., [19,16]). Notice, however, that in general the Deep RC approach is not restrictive in this sense, and it enables the flexibility to have different hyper-parameterizations in different levels of the architecture.

the dimensionality of the reservoir manifold LUD (see Section 3.3, where we set $\theta = 0.9$), and the condition number of the reservoir states C (see Section 3.4). For the RANDOM and LASER datasets, where a single time-series is given, we used a transient of $T = 500$ time-steps to washout initial conditions in the state. For the remaining datasets, in which several time-series are given, to compute the richness quality measures we collected the reservoir states in the last time-step of each time-series. For each configuration explored for the setting of the Deep RC networks, we repeated the experiments a number of 50 times (with different random seeds for the initialization), aggregating all the achieved results. In particular, results presented in this paper are aggregated by the median operator, enabling a robust analysis with respect to possible outliers. In considering the performance to the variation of a specific hyper-parameter, we consider fixed the others, using as a reference (default) hyper-parametrization the one given by: $\rho = 0.9$, $\omega_{in} = 1$, $\omega_{il} = 1$.

In the next Section 4.3 we illustrate the achieved results considering the input data without further re-scaling. For the sake of completeness, and for homogeneity of information processing with uniform scalings coefficients in the various cases, the same experiments were also performed on input data after re-scaling to the common range [-1, 1] along each input dimension. The outcomes of this latter analysis are given in Appendix A, and are qualitatively in line with those reported in Section 4.3.

4.3 Results

We report here the results on reservoir's richness at increasing layer depth, analyzing the effect of layering when varying the spectral radius ρ (Figure 7), the input scaling ω_{in} (Figure 8), and the inter-layer scaling ω_{il} (Figure 9). For the sake of neatness of presentation, the results that we show here have been aggregated on all the datasets. The interested reader can find the detailed results, including the deviations across the different datasets and the results aggregated individually on each dataset, in Appendix A. Notice that in the provided figures the values are represented by a color code, where for ASE and LUD higher (i.e., lighter) values indicate a richer reservoir, for ESP_{index} and C (given in $\log_{10}$ scale) lower (i.e., darker) values are preferable. To ease the results description, we drop the superscript (i) from the notation introduced in Section 3, as here the reference to the layer in the architecture is made explicit in the plots.

Analyzing the results in Figure 7, we first observe, by looking at the ESP_{index} results, that network's dynamics tend to become progressively more unstable in progressively higher layers. In fact, while the ESP_{index} is zero (hence the ESP condition is met) in the first layer for all the explored values of ρ (up to $\rho = 1.3$), in progressively deeper layers values of $\rho > 0.9$ determine progressively larger values of the $\text{ESP}_{index} > 0$, hence more unstable dynamics. Figure 7 also shows the trend in which for every choice of the layer's depth, the ASE and LUD values tend to increase and C tends to decrease, thereby

indicating richer dynamics, with the increasing value of the spectral radius ρ. On the other hand, at the increase of the considered layer's depth, when the dynamics remain in a stable regime (i.e., when $\text{ESP}_{index} \approx 0$), results show a more or less marked depletion of the reservoir quality. A trend of enrichment at the increase of layer's depth is actually observed only when the reservoir is in an unstable regime (i.e., when $\text{ESP}_{index} > 0$), which makes this potential improvement not usable in practice.

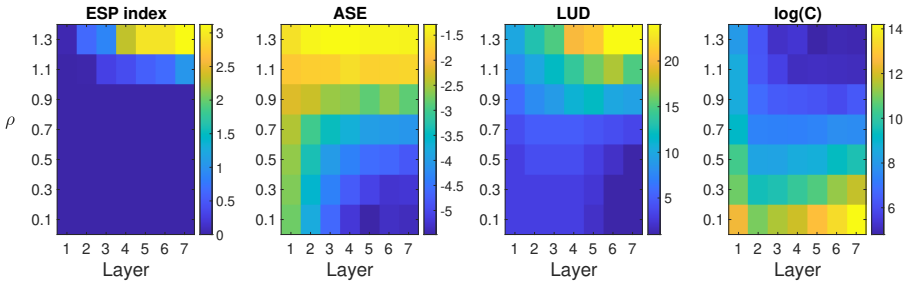


Fig. 7: **Spectral radius variability**. Richness of reservoir dynamics measured in deeper reservoir layers (horizontal axis in each plot), varying the spectral radius ρ hyper-parameterization (vertical axis in each plot). The plots show ESP_{index} (the lower the better), ASE (the higher the better), LUD (the higher the better), and C (in $\log_{10}$ scale, the lower the better). Deviation across datasets and dataset-specific plots are reported in Appendix A.

Looking at Figure 8, we observe that keeping fixed the layer's depth, the goodness of reservoir dynamics tends to improve at the increase of the input scaling hyper-parameter ω_{in}. The reservoir dynamics tend to be more stable (lower values of ESP_{index}) and richer (higher values of ASE and LUD, lower values of C). On the other hand, keeping the value of ω_{in} fixed, at the increase of the layer's depth the results show progressively less stable dynamics (though generally close to ideal values of the ESP_{index}[7] especially for higher values of ω_{in}). The other metrics do not show a uniform trend. While ASE tends to worsen at the increase of layer's depth for $\omega_{in} \geq 1$, and to improve (albeit with low values) for $\omega_{in} < 1$, LUD shows an opposite trend, while C shows a general improvement for increasing depth (though less significant for $\omega_{in} < 1$).

Figure 9 shows interesting and neat results on the synergy between the inter-layer scaling ω_{il} and the depth of the recurrent architecture. Keeping fixed the depth, all the measured metrics display a trend of improvement with the increase of ω_{il}. Moreover, at the increase of the depth of the considered reservoir layer, the richness of system's dynamics generally shows a marked improvement for values of $\omega_{il} > 1$ (with a peak for $\omega_{il} < 10$), a marked

[7] Note the values of the color scale in the ESP_{index} plot in Figure 8, especially in comparison to those in the same plot of Figure 7.

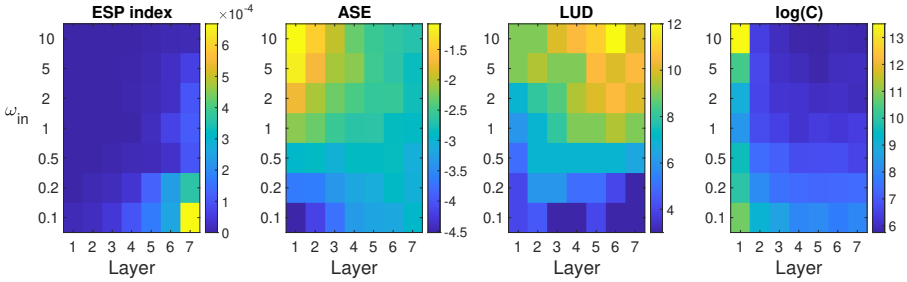


Fig. 8: **Input scaling variability**. Richness of reservoir dynamics measured in deeper reservoir layers (horizontal axis in each plot), varying the input scaling ω_{in} hyper-parameterization (vertical axis in each plot). The plots show ESP_{index} (the lower the better), ASE (the higher the better), LUD (the higher the better), and C (in $\log_{10}$ scale, the lower the better). Deviation across datasets and dataset-specific plots are reported in Appendix A.

degradation for $\omega_{il} < 1$, and a trend of substantial non-variability for $\omega_{il} = 1$.

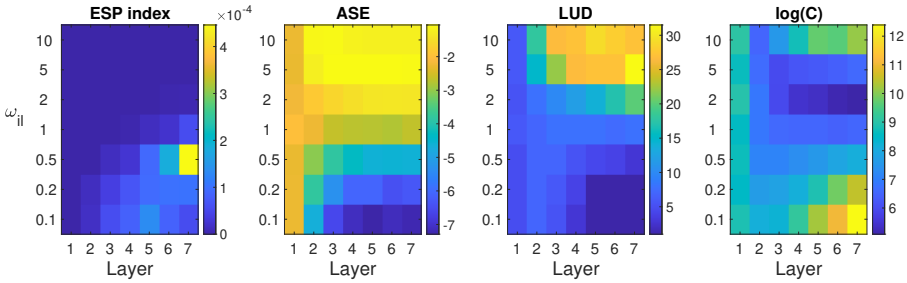


Fig. 9: **Inter-layer scaling variability**. Richness of reservoir dynamics measured in deeper reservoir layers (horizontal axis in each plot), varying the inter-layer scaling ω_{il} hyper-parameterization (vertical axis in each plot). The plots show ESP_{index} (the lower the better), ASE (the higher the better), LUD (the higher the better), and C (in $\log_{10}$ scale, the lower the better). Deviation across datasets and dataset-specific plots are reported in Appendix A.

From the analyzed results we can make some overall comments. In general, the degree of richness of the reservoir dynamics is sensitive to both the values of the reservoir hyper-parameters and the depth of the reservoir layer in the architecture. Our results show that in progressively deeper networks it is possible to observe either a progressive enrichment of the dynamics or their progressive impoverishment. This fundamental observation is interestingly linked to the empirical result that prediction accuracy in DeepESNs is generally non-monotonic with the addition of more reservoir layers (see e.g.,

[19,15]): adding layers is not per se sufficient to improve the goodness (nor the performance) of the reservoir system in applications. In this sense, the value of the scaling of inter-layer connections is crucial: values of $\omega_{il} > 1$ tend to result in more stable and richer dynamics, at the same time setting up a better conditioned learning problem for readout. In the sense of the progressive qualitative change of dynamics in deeper layers, the other scaling hyper-parameters seem to play a decidedly less prominent role.

Finally, a few hints on model selection in practical applications of Deep RC can be derived from the above observations and put forward for the RC practitioners. Most relevantly, the number of reservoir layers should be considered as a crucial hyper-parameter of the model, and thereby its value should be tuned (along with the values of the other hyper-parameters) on a validation set, rather than fixed to an arbitrary value. The interested reader can also find an automatic approach (rooted in the frequency analysis of the reservoir state content) to tune the number of layers in a DeepESN in one of our previous works [19]. Secondly, when tuning the scaling parameters of the deep reservoir, our empirical results suggest that it would be useful to start from exploring values of the inter-layer scaling ω_{il} larger than 1. Moreover, from a broader perspective, our analysis revealed that a non-negligible portion of the hyper-parameters space can lead to intrinsically poor quality reservoirs (e.g., with $ESP_{index} > 0$), and in such cases training the readout layer could be an avoidable cost. In applications, it would be then advisable to preliminarily sample the hyper-parameters space to identify the most promising regions (e.g., those corresponding to the richer resulting dynamics according to the measures proposed in this paper) and then deepen the tuning process by the standard supervised model selection phase only in the selected promising regions.

5 Conclusions

In this paper we have presented an empirical study on the richness of the hidden states representations developed in deeper layers of untrained deep RNNs, implemented according to the RC framework. Our analysis targeted a diverse range of metrics used to quantify the goodness of state dynamics from different perspectives. The stability regime was analyzed by the ESP index. The degree of diversity of reservoir's units activations was investigated by the Average State Entropy and by the number of Linearly Uncoupled Dynamics. Finally, the extent to which the operation of the reservoir facilitates the learning in the case of supervised problems was studied by the condition number of the resulting state matrix. The empirical analysis was conducted on datasets with different characteristics, including both common RC benchmarks and real-world data from diverse domains.

The outcomes of our experiments clearly point out that layering has an inherent effect on the quality of RNNs dynamics. Depending on the architectural construction (i.e., the layer's depth) and on the initialization of the

weight matrices involved in the state computation (analyzed here in terms of RC-based scaling coefficients), the quality of the recurrent representations can be enhanced or reduced even in the absence, or prior to, learning of the recurrent connections. In particular, our results suggest that an especially crucial role is played by the maximum singular value of the inter-layer weight matrices ω_{il}. To determine a progressive enrichment of the state dynamics the value of ω_{il} should be larger than 1.

Our study paves the way to several future research directions. First of all, it would be interesting to study the impact of the organization of the untrained network connections on the richness of the resulting reservoir dynamics. A starting point in this direction would consist in exploring diverse setups for the sparsity of input, inter-layer and reservoir connections, including topological organization of special interest, like cycle, permutation or small-world reservoirs [38,30]. Moreover, extending the analysis presented in this paper to settings with possibly different reservoir sizes in different layers of the Deep RC network would also allow exploring the most suitable architectural organization of an RC system on a given computational budget, i.e., to address the question of finding the convenient number of layers in which to arrange a given total number of reservoir neurons. The analysis presented here would also stimulate the development of input-driven pre-training strategies aiming at tailoring the architectural construction of deep RNNs towards regions of the hyper-parameters space where the developed dynamics are rich by construction, hence facilitating training in downstream supervised learning tasks. Besides, from an RC perspective, an interesting side effect of our analysis is the observation that a proper construction of deep reservoir architectures leads to a better-conditioned learning problem for the readout (as testified by a smaller value of the condition number). The consequence is that Deep RC neural networks lend themselves more easily to an effective application of efficient iterative readout training algorithms based on least mean square error, which is traditionally ineffective in shallow RC due to bad conditioning [27, 32]. We believe that this direction is particularly attractive for a successful design of distributed deep learning systems for temporal data embedded in low-powerful devices. Furthermore, extensions of our analysis to the case of deep reservoirs for graphs [16] is also of a great appeal.

References

1. Atiya, A.F., Parlos, A.G.: New results on recurrent network training: unifying the algorithms and accelerating convergence. IEEE Transactions on Neural Networks **11**(3), 697–709 (2000)
2. Bacciu, D., Barsocchi, P., Chessa, S., Gallicchio, C., Micheli, A.: An experimental characterization of reservoir computing in ambient assisted living applications. Neural Computing and Applications **24**(6), 1451–1464 (2014)
3. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks **5**(2), 157–166 (1994)
4. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The ucr time series classification archive (2015). `www.cs.ucr.edu/~eamonn/time_series_data/`

5. Colla, V., Matino, I., Dettori, S., Cateni, S., Matino, R.: Reservoir computing approaches applied to energy management in industry. In: International Conference on Engineering Applications of Neural Networks, pp. 66–79. Springer (2019)
6. Cover, T.M.: Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. IEEE transactions on electronic computers (3), 326–334 (1965)
7. Dettori, S., Matino, I., Colla, V., Speets, R.: Deep echo state networks in industrial applications. In: IFIP International Conference on Artificial Intelligence Applications and Innovations, pp. 53–63. Springer (2020)
8. Dias, D.B., Madeo, R.C., Rocha, T., Biscaro, H.H., Peres, S.M.: Hand movement recognition for brazilian sign language: a study using distance-based neural networks. In: 2009 international joint conference on neural networks, pp. 697–704. IEEE (2009)
9. Dua, D., Graff, C.: UCI machine learning repository (2017). URL `http://archive.ics.uci.edu/ml`
10. Gallicchio, C.: Chasing the echo state property. In: 27th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2019, pp. 667–672. ESANN (i6doc. com) (2019)
11. Gallicchio, C., Micheli, A.: A markovian characterization of redundancy in echo state networks by pca. In: Proc. of the 18th European Symposium on Artificial Neural Networks (ESANN). d-side publi. (2010)
12. Gallicchio, C., Micheli, A.: Architectural and markovian factors of echo state networks. Neural Networks **24**(5), 440–456 (2011)
13. Gallicchio, C., Micheli, A.: Deep echo state network (deepesn): A brief survey. arXiv preprint arXiv:1712.04323 (2017)
14. Gallicchio, C., Micheli, A.: Echo state property of deep reservoir computing networks. Cognitive Computation **9**(3), 337–350 (2017)
15. Gallicchio, C., Micheli, A.: Reservoir topology in deep echo state networks. In: International Conference on Artificial Neural Networks, pp. 62–75. Springer (2019)
16. Gallicchio, C., Micheli, A.: Fast and deep graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3898–3905 (2020)
17. Gallicchio, C., Micheli, A.: Deep reservoir computing. In: K. Nakajima, I. Fischer (eds.) Reservoir Computing, pp. 77–95. Springer (2021)
18. Gallicchio, C., Micheli, A., Pedrelli, L.: Deep reservoir computing: A critical experimental analysis. Neurocomputing **268**, 87–99 (2017). DOI https://doi.org/10.1016/j.neucom.2016.12.089
19. Gallicchio, C., Micheli, A., Pedrelli, L.: Design of deep echo state networks. Neural Networks **108**, 33–47 (2018)
20. Gallicchio, C., Scardapane, S.: Deep randomized neural networks. Recent Trends in Learning From Data pp. 43–68 (2020)
21. Graves, A., Mohamed, A.r., Hinton, G.: Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing, pp. 6645–6649. Ieee (2013)
22. Haber, E., Ruthotto, L.: Stable architectures for deep neural networks. Inverse Problems **34**(1), 014004 (2017)
23. Hermans, M., Schrauwen, B.: Training and analysing deep recurrent neural networks. Advances in neural information processing systems **26**, 190–198 (2013)
24. Hu, H., Wang, L., Lv, S.X.: Forecasting energy consumption and wind power generation using deep echo state network. Renewable Energy **154**, 598–613 (2020)
25. Jaeger, H.: 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)
26. Jaeger, H.: Short term memory in echo state networks. Tech. rep., GMD-German National Research Institute for Computer Science (2002)
27. Jaeger, H.: Reservoir riddles: Suggestions for echo state network research. In: Proceedings of the 2005 IEEE International Joint Conference on Neural Networks (IJCNN), vol. 3, pp. 1460–1462. IEEE (2005)
28. Jaeger, H., Haas, H.: Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. Science **304**(5667), 78–80 (2004)

29. Jaeger, H., Lukoševičius, M., Popovici, D., Siewert, U.: Optimization and applications of echo state networks with leaky-integrator neurons. Neural Networks **20**(3), 335–352 (2007)
30. Kawai, Y., Park, J., Asada, M.: A small-world topology enhances the echo state property and signal propagation in reservoir computing. Neural Networks **112**, 15–23 (2019)
31. Kim, T., King, B.R.: Time series prediction using deep echo state networks. Neural Computing and Applications **32**(23), 17769–17787 (2020)
32. Lukoševičius, M., Jaeger, H.: Reservoir computing approaches to recurrent neural network training. Computer Science Review **3**(3), 127–149 (2009)
33. Olszewski, R.T.: Generalized feature extraction for structural pattern recognition in time-series data. Tech. rep., CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE (2001)
34. Ozturk, M., Xu, D., Principe, J.: Analysis and design of echo state networks. Neural computation **19**(1), 111–138 (2007)
35. Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y.: How to construct deep recurrent neural networks. arXiv preprint arXiv:1312.6026 (2013)
36. Principe, J., Xu, D., Fisher, J., Haykin, S.: Information theoretic learning. unsupervised adaptive filtering. Unsupervised Adaptive Filtering **1** (2000)
37. Principe, J.C.: Information theoretic learning: Renyi's entropy and kernel perspectives. Springer Science & Business Media (2010)
38. Rodan, A., Tiňo, P.: Minimum complexity echo state network. IEEE transactions on neural networks **22**(1), 131–144 (2010)
39. Scardapane, S., Wang, D.: Randomness in neural networks: an overview. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **7**(2), e1200 (2017)
40. Tiňo, P., Hammer, B., Bodén, M.: Markovian bias of neural-based architectures with feedback connections. In: Perspectives of neural-symbolic integration, pp. 95–133. Springer (2007)
41. Verstraeten, D., Schrauwen, B., d'Haene, M., Stroobandt, D.: An experimental unification of reservoir computing methods. Neural networks **20**(3), 391–403 (2007)
42. Weigend, A.S.: Time series prediction: forecasting the future and understanding the past. Routledge (2018)
43. Werbos, P.J.: Backpropagation through time: what it does and how to do it. Proceedings of the IEEE **78**(10), 1550–1560 (1990)
44. Williams, B.H., Toussaint, M., Storkey, A.J.: Extracting motion primitives from natural handwriting data. In: International Conference on Artificial Neural Networks, pp. 634–643. Springer (2006)
45. Xue, Y., Yang, L., Haykin, S.: Decoupled echo state networks with lateral inhibition. Neural Networks **20**(3), 365–376 (2007)
46. Yildiz, I., Jaeger, H., Kiebel, S.: Re-visiting the echo state property. Neural networks **35**, 1–9 (2012)

Conflict of interest

The authors declare that they have no conflict of interest.

Acknowledgments

This work has been partially supported by the European Union's Horizon 2020 Research and Innovation program, under project TEACHING (Grant agreement ID: 871385), URL `https://www.teaching-h2020.eu`, and by the project BrAID under the Bando Ricerca Salute 2018 – Regional public call for research and development projects aimed at supporting clinical and organisational innovation processes of the Regional Health Service – Regione Toscana.

A Further Results

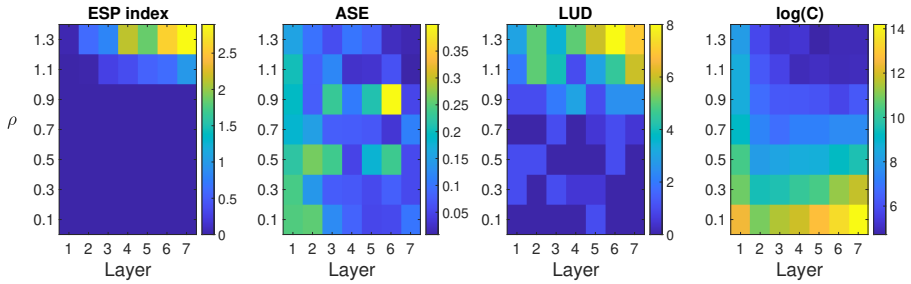
Here, we report additional experimental results that provide further support to the analysis conducted in the paper.

In particular, in Figure 10, we show the deviation of the results across the different datasets, expressed in terms of median average deviation (MAD)[8], for the spectral radius, the input scaling and the inter-layer variability. The provided plots match the corresponding median results given in Figures 7, 8, and 9. We can observe that in general the deviation is rather small and, varying the reservoir configurations and the number of layers, shows a trend generally in line with the observations made in Section 4.3, with tendentially lower values for richer reservoirs.

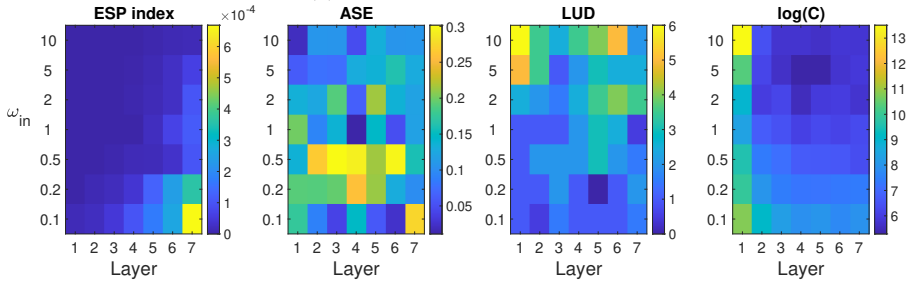
Next, we detail the outcomes of the experiments reported in Section 4.3, aggregated on the individual datasets. We provide the values of ESP_{index}, ASE, LUD and C, achieved under the experimental settings illustrated in Section 4.2, varying the value of the spectral radius ρ in Figure 11, varying the value of the input scaling ω_{in} in Figure 12, and varying the value of the inter-layer scaling ω_{il} in Figure 13. In each figure, each row corresponds to a different dataset. The values shown in the plots evidently confirm the same trends analyzed in Section 4.3 (see Figures 7, 8, and 9).

Finally, we report the outcomes of the further experiments conducted in the same conditions as in Section 4.2, but with a preliminary re-scaling in $[-1, 1]$ of the input along each dimension individually, for each dataset. Results are given in Figure 14, and, as it is apparent from the plots, are qualitatively in line with those without re-scaling shown in Figures 7, 8, and 9, thereby confirming the analysis discussed in Section 4.3.

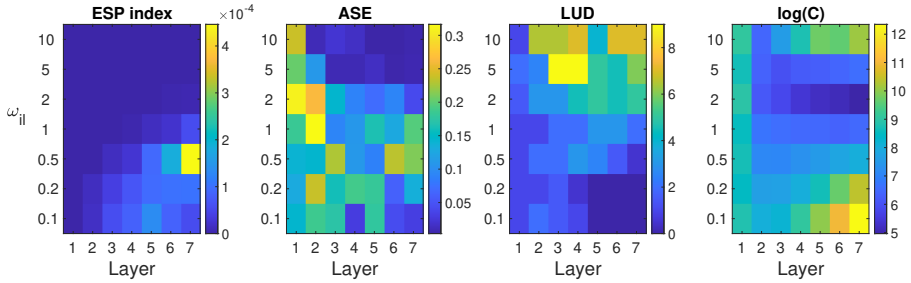
[8] Given a vector $\mathbf{v}$, $\text{MAD}(\mathbf{v}) = \text{MEDIAN}(|\mathbf{v} - \text{MEDIAN}(\mathbf{v})|)$.



(a) Spectral radius variability.

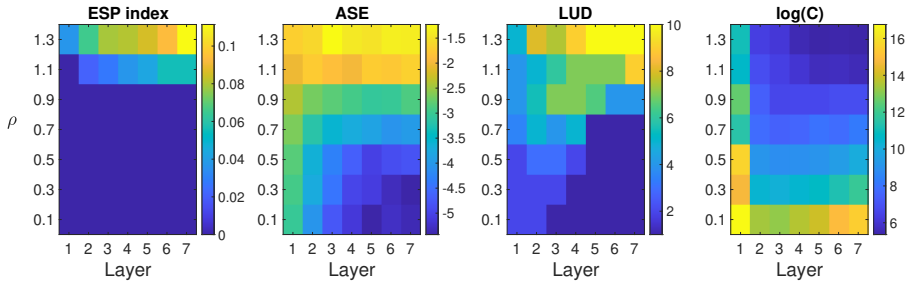


(b) Input scaling variability.

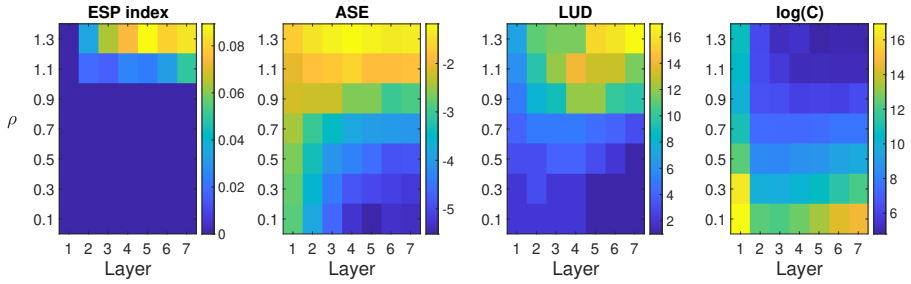


(c) Inter-layer scaling variability.

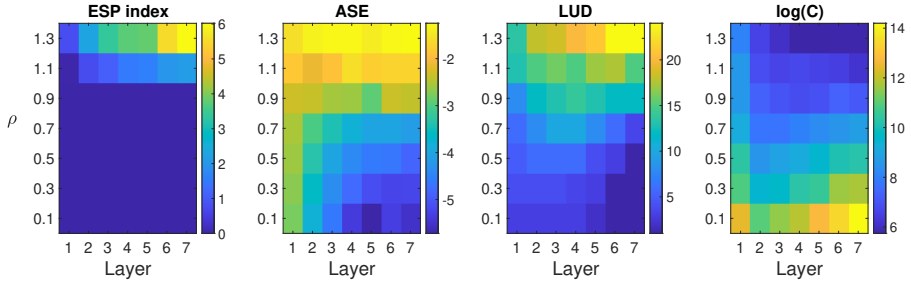
Fig. 10: Median absolute deviation across the results for the different datasets considered in Figures 7, 8 and 9. Results are reported for ESP_{index}, ASE, LUD and C (in $\log_{10}$ scale) measured in deeper reservoir layers, varying the values of the spectral radius ρ (Figure 10a), of the input scaling ω_{in} (Figure 10b), and of the inter-layer scaling ω_{in} (Figure 10b).



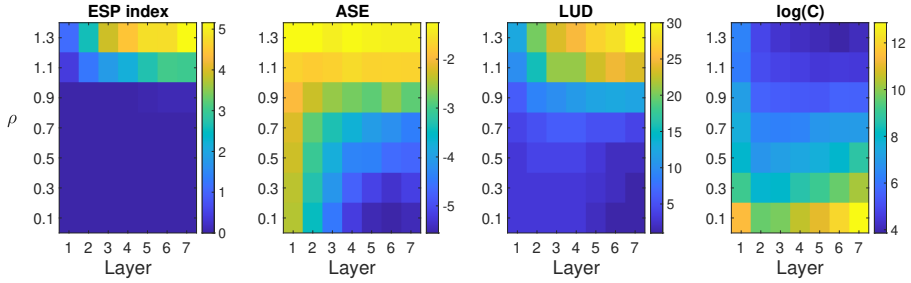
(a) RANDOM dataset.



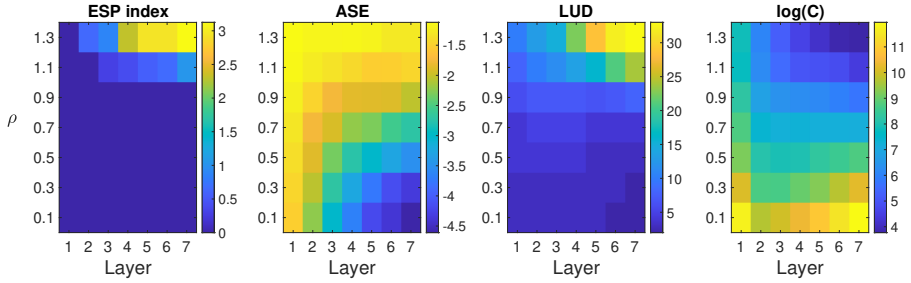
(b) LASER dataset.



(c) ECG dataset.

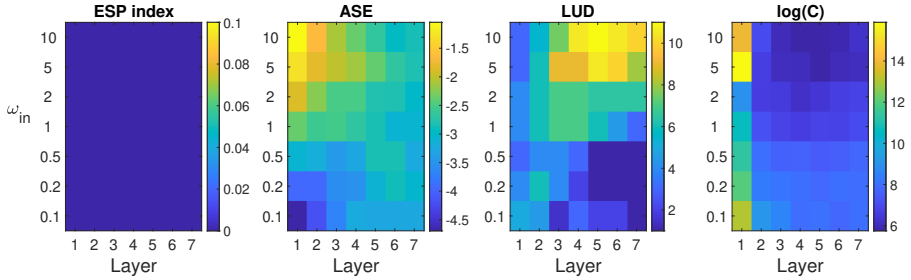


(d) LIBRAS dataset.

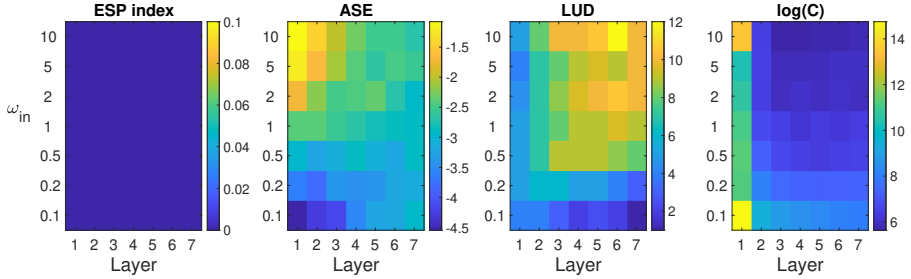


(e) CHAR dataset.

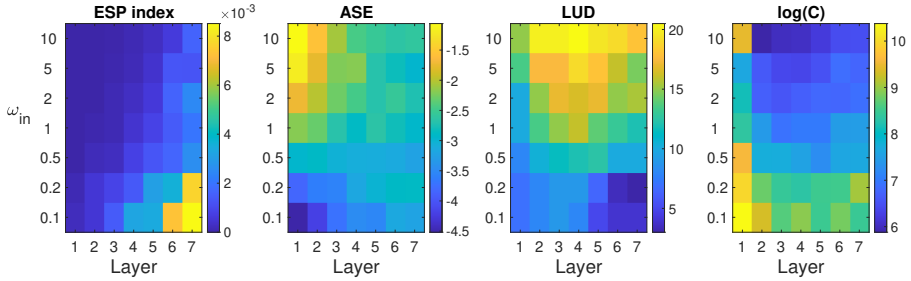
Fig. 11: **Spectral radius variability**. ESP_{index}, ASE, LUD and C (in $\log_{10}$ scale) measured in deeper reservoir layers (horizontal axis in each plot), varying the value of the spectral radius ρ (vertical axis in each plot). Results are presented individually for each dataset.



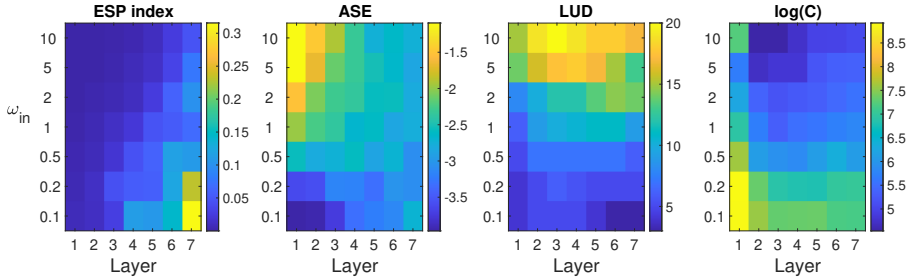
(a) RANDOM dataset.



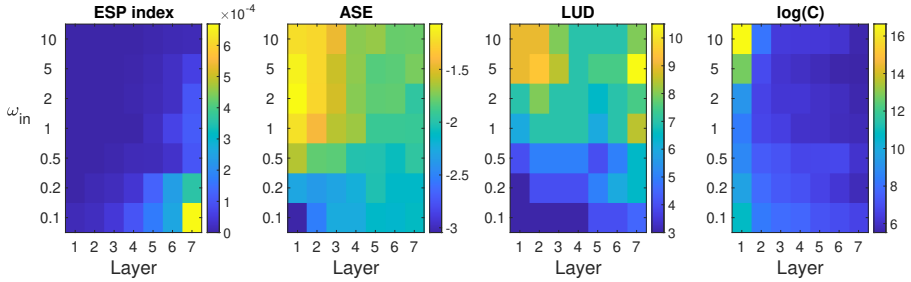
(b) LASER dataset.



(c) ECG dataset.

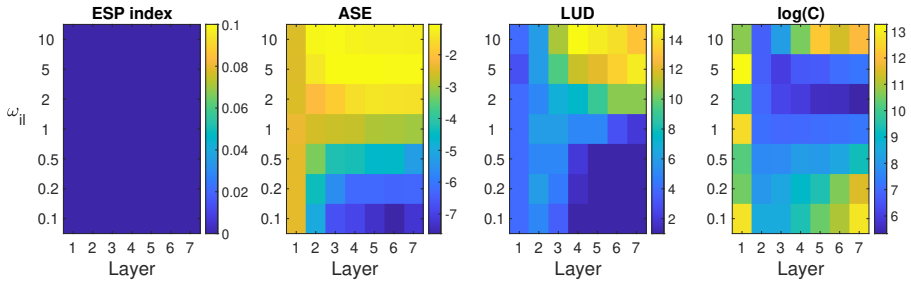


(d) LIBRAS dataset.

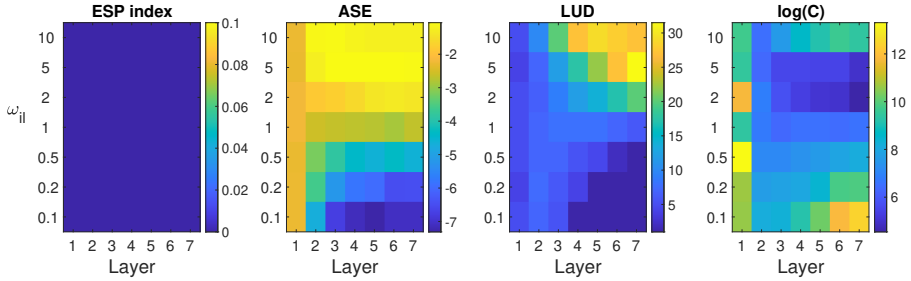


(e) CHAR dataset.

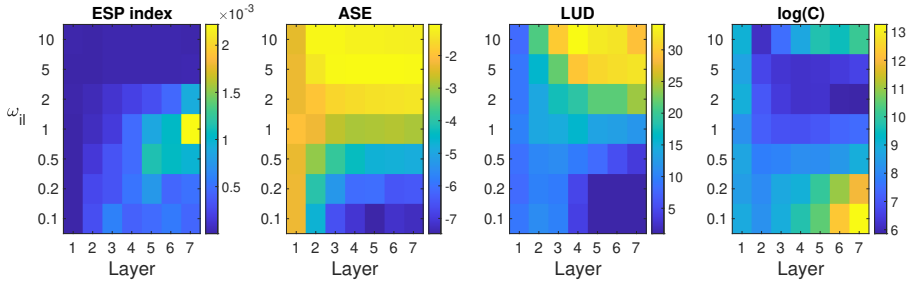
Fig. 12: **Input scaling variability**. ESP_{index}, ASE, LUD and C (in $\log_{10}$ scale) measured in deeper reservoir layers (horizontal axis in each plot), varying the value of the input scaling ω_{in} (vertical axis in each plot). Results are presented individually for each dataset.



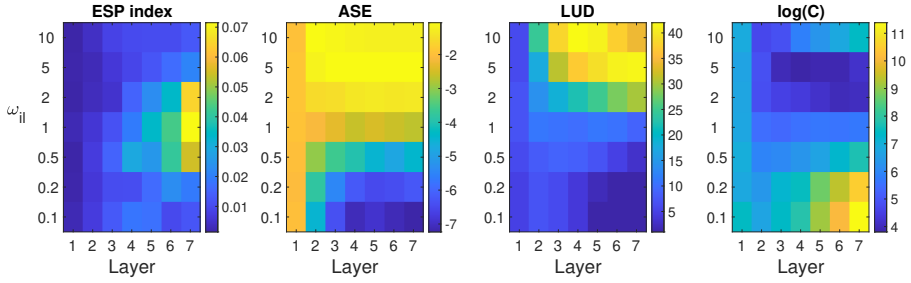
(a) RANDOM dataset.



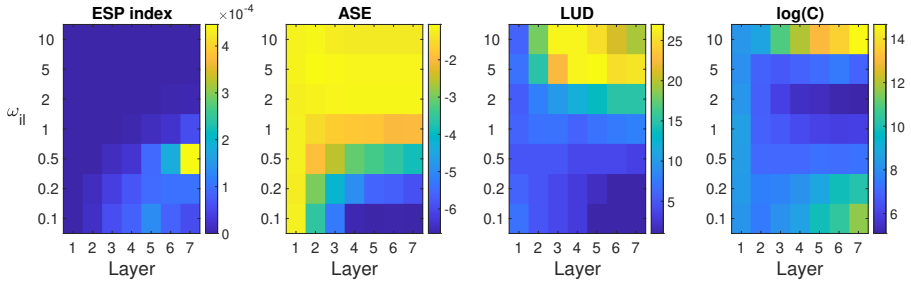
(b) LASER dataset.



(c) ECG dataset.

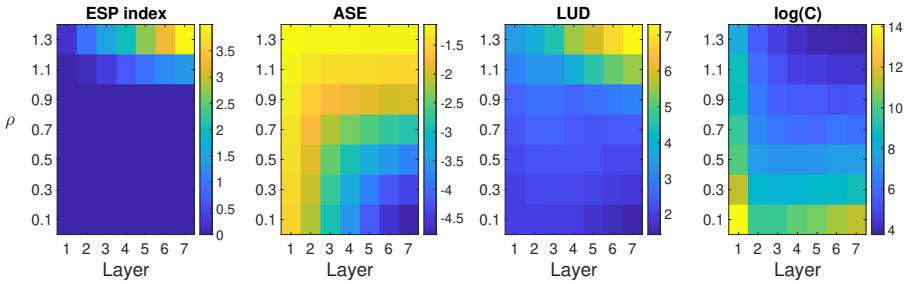


(d) LIBRAS dataset.

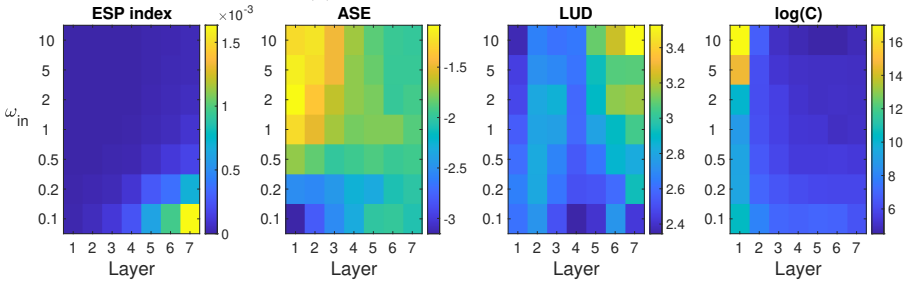


(e) CHAR dataset.

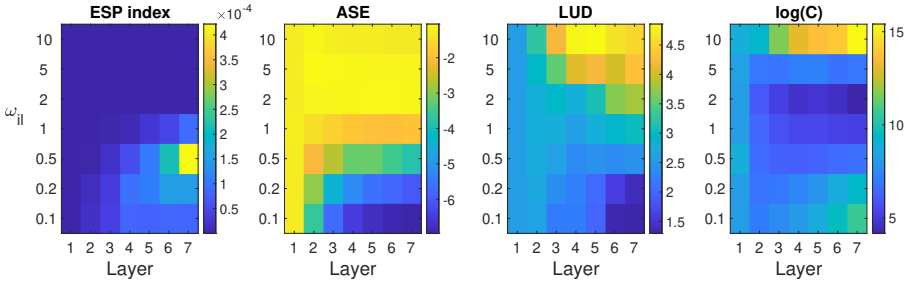
Fig. 13: **Inter-layer scaling variability**. ESP_{index}, ASE, LUD and C (in $\log_{10}$ scale) measured in deeper reservoir layers (horizontal axis in each plot), varying the value of the inter-layer scaling ω_{il} (vertical axis in each plot). Results are presented individually for each dataset.



(a) Spectral radius variability.



(b) Input scaling variability.



(c) Inter-layer scaling variability.

Fig. 14: Richness of reservoir dynamics measured in deeper reservoir layers, varying the values of the spectral radius ρ (Figure 14a), of the input scaling ω_n (Figure 14b), and of the inter-layer scaling (Figure 14c). The plots show ESP_{index} (the lower the better), ASE (the higher the better), LUD (the higher the better), and C (in log_{10} scale, the lower the better). Results are aggregated across all datasets. Differently from the results shown in Figures 11, 12, and 13, the input is re-scaled to a common range $[-1, 1]$ for all the datasets.