Article:
Manneschi, L., Ellis, M.O.A., Gigante, G. et al. (2021) Exploiting multiple timescales in hierarchical echo state networks. Frontiers in Applied Mathematics and Statistics, 6. 616658. ISSN: 2297-4687

https://doi.org/10.3389/fams.2020.616658

frontiers in Applied Mathematics and Statistics

ORIGINAL RESEARCH
published: 17 February 2021
doi: 10.3389/fams.2020.616658

Exploiting Multiple Timescales in Hierarchical Echo State Networks

Luca Manneschi[1], *Matthew O. A. Ellis*[1], *Guido Gigante*[2], *Andrew C. Lin*[3,4], *Paolo Del Giudice*[2†] *and Eleni Vasilaki*[1*†]

[1]Department of Computer Science, The University of Sheffield, Sheffield, United Kingdom, [2]National Center for Radiation Protection and Computational Physics, Italian Institute of Health, Rome, Italy, [3]Department of Biomedical Science, The University of Sheffield, Sheffield, United Kingdom, [4]Neuroscience Institute, The University of Sheffield, Sheffield, United Kingdom

Echo state networks (ESNs) are a powerful form of reservoir computing that only require training of linear output weights while the internal reservoir is formed of fixed randomly connected neurons. With a correctly scaled connectivity matrix, the neurons' activity exhibits the echo-state property and responds to the input dynamics with certain timescales. Tuning the timescales of the network can be necessary for treating certain tasks, and some environments require multiple timescales for an efficient representation. Here we explore the timescales in hierarchical ESNs, where the reservoir is partitioned into two smaller linked reservoirs with distinct properties. Over three different tasks (NARMA10, a reconstruction task in a volatile environment, and psMNIST), we show that by selecting the hyper-parameters of each partition such that they focus on different timescales, we achieve a significant performance improvement over a single ESN. Through a linear analysis, and under the assumption that the timescales of the first partition are much shorter than the second's (typically corresponding to optimal operating conditions), we interpret the feedforward coupling of the partitions in terms of an effective representation of the input signal, provided by the first partition to the second, whereby the instantaneous input signal is expanded into a weighted combination of its time derivatives. Furthermore, we propose a data-driven approach to optimise the hyper-parameters through a gradient descent optimisation method that is an online approximation of backpropagation through time. We demonstrate the application of the online learning rule across all the tasks considered.

Keywords: reservoir computing (RC), echo state network (ESN), timescales, hyperparameter adaptation, backpropagation through time

OPEN ACCESS

Edited by:
Andre Gruning, University of Surrey, United Kingdom

Reviewed by:
Petia D. Koprinkova-Hristova, Institute of Information and Communication Technologies (BAS), Bulgaria
Igor Farkaš, Comenius University, Slovakia

***Correspondence:**
Eleni Vasilaki
e.vasilaki@sheffield.ac.uk

†These authors share senior authorship

Specialty section:
This article was submitted to Dynamical Systems, a section of the journal Frontiers in Applied Mathematics and Statistics

Received: *12 October 2020*
Accepted: *29 December 2020*
Published: *17 February 2021*

Citation:
Manneschi L, Ellis MOA, Gigante G, Lin AC, Del Giudice P and Vasilaki E (2021) Exploiting Multiple Timescales in Hierarchical Echo State Networks. Front. Appl. Math. Stat. 6:616658. doi: 10.3389/fams.2020.616658

1 INTRODUCTION

The high inter-connectivity and asynchronous loop structure of Recurrent Neural Networks (RNNs) make them powerful techniques for processing temporal signals [1]. However, the complex inter-connectivity of RNNs means that they cannot be trained using the conventional back-propagation (BP) algorithm [2] used in feed-forward networks, since each neuron's state depends on other neuronal activities at previous times. A method known as Back-Propagation-Through-Time (BPTT) [3], which relies on an unrolling of neurons' connectivity through time to propagate the error signal to earlier time states, can be prohibitively complex for large networks or time series. Moreover, BPTT is not considered biologically plausible as neurons must retain memory of their activation over the length of the input and the error signal must be propagated backwards with symmetric synaptic weights [4].

Many of these problems can be avoided using an alternative approach: reservoir computing (RC). In the subset of RC networks known as Echo State networks, a fixed "reservoir" transforms a temporal input signal in such a way that only a single layer output perceptron needs to be trained to solve a learning task. The advantage of RC is that the reservoir is a fixed system that can be either computationally or physically defined. Since it is fixed it is not necessary to train the reservoir parameters through BPTT, making RC networks much simpler to train than RNNs. Furthermore, the random structure of a RC network renders the input history over widely different time-scales, offering a representation that can be used for a wide variety of tasks without optimising the recurrent connectivity between nodes.

Reservoirs have biological analogues in cerebellum-like networks (such as the cerebellum, the insect mushroom body and the electrosensory lobe of electric fish), in which input signals encoded by relatively few neurons are transformed via "expansion re-coding" into a higher-dimensional space in the next layer of the network, which has many more neurons than the input layer [5–8]. This large population of neurons (granule cells in the cerebellum; Kenyon cells in the mushroom body) acts as a reservoir because their input connectivity is fixed and learning occurs only at their output synapses. The principal neurons of the "reservoir" can form chemical and electrical synapses on each other (e.g., Kenyon cells: [9–11]), analogous to the recurrent connectivity in reservoir computing that allows the network to track and transform temporal sequences of input signals. In some cases, one neuronal layer with recurrent connectivity might in turn connect to another neuronal layer with recurrent connectivity; for example, Kenyon cells of the mushroom body receive input from olfactory projection neurons of the antennal lobe, which are connected to each other by inhibitory and excitatory interneurons [12, 13]. Such cases can be analogised to hierarchically connected reservoirs. In biological systems, it is thought that transforming inputs into a higher-dimensional neural code in the "reservoir" increases the associative memory capacity of the network [5]. Moreover, it is known that for the efficient processing of information unfolding in time, which requires networks to dynamically keep track of past stimuli, the brain can implement ladders of neural populations with hierarchically organised "temporal receptive fields" [14].

The same principles of dimensional expansion in space and/or time apply to artificial RC networks, depending on the non-linear transformation of the inputs into a representation useful for learning the task at the single linear output layer. We focus here on a popular form of RC called Echo State Networks [15], where the reservoir is implemented as a RNN with a fixed, random synaptic connection matrix. This connection matrix is set so the input "echoes" within the network with decaying amplitude. The performance of an Echo State Network depends on certain network hyper-parameters that need to be optimised through grid search or explicit gradient descent. Given that the dependence of the network's performance on such hyper-parameters is both non-linear and task-dependent, such optimisation can be tedious.

Previous works have studied the dependence of the reservoir properties on the structure of the random connectivity adopted, studying the dependence of the reservoir performance on the parameters defining the random connectivity distribution, and formulating alternatives to the typical Erdos-Renyi graph structure of the network [16–18]. In this sense, in [17] a model with a regular graph structure has been proposed, where the nodes are connected forming a circular path with constant shortest path lengths equal to the size of the network, introducing long temporal memory capacity by construction. The memory capacity has been studied previously for network parameters such as the spectral radius (ρ) and sparsity; in general memory capacity is higher for ρ close to one and low sparsity, but high memory capacity does not guarantee high prediction [19, 20]. ESNs are known to perform optimally when at the "edge of criticality" [21], where low prediction error and high memory can be achieved through network tuning.

More recently, models composed of multiple reservoirs have gathered the attention of the community. From the two ESNs with lateral inhibition proposed in [22], to the hierarchical structure of reservoirs first analyzed by Jaeger in [23], these complex architectures of multiple, multilayered reservoirs have shown improved generalisation abilities over a variety of tasks [23–25]. In particular, the works [26, 27] have studied different dynamical properties of such hierarchical structures of ESNs, while [28] have proposed hierarchical (or deep) ESNs with projection encoders between layers to enhance the connectivity of the ESN layers. The partitioning (or modularity) of ESNs was studied by [29], where the ratio of external to internal connections was varied. By tuning this partitioning performance can be increased on memory or recall tasks. Here we demonstrate that one of the main reasons to adopt a network composed by multiple, pipelined sub-networks, is the ability to introduce multiple timescales in the network's dynamics, which can be important in finding optimal solutions for complex tasks. Examples of tasks that require such properties are in the fields of speech, natural language processing, and reward driven learning in partially observable Markov decision processes [30]. A hierarchical structure of temporal kernels [31], as multiple connected ESNs, can discover higher level features of the input temporal dynamics. Furthermore, while a single ESN can be tuned to incorporate a distribution of timescales with a prefixed mode, optimising the system hyper-parameters to cover a wide range of timescales can be problematic.

Here, we show that optimisation of hyper-parameters can be guided by analysing how these hyper-parameters are related to the timescales of the network, and by optimising them according to the temporal dynamics of the input signal and the memory required to solve the considered task. This analysis improves performance and reduces the search space required in hyper-parameter optimisation. In particular, we consider the case where an ESN is split into two sections with different hyper-parameters resulting in separate temporal properties. In the following, we will first provide a survey of timescales in ESNs before presenting the comparative success of these hierarchical ESNs on three different

tasks. The first is the non-linear auto-regressive moving average 10 (NARMA10) task which requires both memory and fast non-linear transformation of the input. Second, we explore the performance of the network in a reconstruction and state "perception" task with different levels of external white noise applied on the input signal. Finally, we apply the hierarchical ESN to a permuted sequential MNIST classification task, where the usual MNIST hand written digit database is serialised and permuted as a 1 dimensional time-series.

2 SURVEY OF TIMESCALES IN ECHO STATE NETWORKS

We begin by describing the operations of an ESN and present a didactic survey of their inherent timescales, which will be drawn upon in later sections to analyze the results.

As introduced in the previous section, an ESN is a recurrent neural network and the activity, $\mathbf{x}(t)$, of the neurons due to a temporal input signal $\mathbf{s}(t)$ is given by

$$\mathbf{x}(t+\delta t) = (1-\alpha)\mathbf{x}(t) + \alpha f(\mathbf{h}(t)), \tag{1}$$

$$\mathbf{h}(t) = \gamma \mathbf{W}_{\text{in}}\mathbf{s}(t) + \rho \mathbf{W}\mathbf{x}(t), \tag{2}$$

where $\mathbf{W}$ is a possibly sparse random matrix defining the connectivity of the network, $\mathbf{W}_{\text{in}}$ defines the input adjacency matrix, and γ is a rescaling factor of the input weights. $\alpha = \delta t/\tau$ is the leakage term of the node, and ρ is a scaling factor for the spectral radius of the connectivity matrix and will be discussed in more detail in the following. $f()$ is a non-linear function, which in this work we define as the hyperbolic tangent. To ensure that the network exhibits the Echo-State property, and so that the activity does not saturate, the initial random connectivity matrix, $\mathbf{W}$, is rescaled by its maximum eigenvalue magnitude (spectral radius), $\left|\lambda_{\mathbf{W}}^{\max}\right| = \max\left|\text{eig}(\mathbf{W})\right|$, thus ensuring a unitary spectral radius which can be tuned using ρ as a hyper-parameter. In practice, $\mathbf{W}$ is constructed from a matrix of Normally distributed random numbers and the sparseness is enforced by randomly setting to zero a fixed proportion of these elements. Typically 10 non-zero connections per node are retained in $\mathbf{W}$.

The timescales of this dynamical system are closely linked to the specific structure of $\mathbf{W}$ and to the two hyper-parameters; α and ρ. Since α is the leakage rate, it directly controls the retention of information from previous time steps, while ρ specifies the maximum absolute magnitude of the eigenvalues and as such tunes the decay time of internal activity of the network. Thus, the basic hyper-parameters that need to be set are γ, α and ρ. Considering the nonlinear dependence of the network performance on these values and the task-dependent nature of an efficient parameterisation, this process can be challenging. Such hyper-parameters are commonly optimised through a grid search or through explicit gradient descent methods in online learning paradigms [32]. However, the fine tuning procedure can be guided, and the searchable space reduced, using a simple analysis of the hyper-parameters' relation to the timescales of the network, the external signal's temporal dynamics, and the memory required to solve the considered task.

Considering that the eigenvalues $\lambda_{\mathbf{W}}$ of the connectivity matrix are inside the imaginary unit circle due to the normalisation procedure described previously, and that α is a constant common to all neurons, the eigenvalues of the linearised system given by **Eq. 1** are

$$\lambda = 1 - \alpha(1 - \rho\lambda_{\mathbf{W}}). \tag{3}$$

This corresponds to a rescaling of value $\alpha\rho$ and to a translation of value $1-\alpha$ across the real axis of the original $\lambda_{\mathbf{W}}$. This operation on the eigenvalues of $\mathbf{W}$ is depicted in **Figure 1A**. Thus, considering that each eigenvalue λ_i can be decomposed in its corresponding exponential decaying part $\exp(-\delta t/\tau_i)$ and its oscillatory imaginary component, the timescales of the linearised system are

$$\tau = \frac{\delta t}{1 - \text{Re}(\lambda)} \tag{4}$$

$$= \frac{\delta t}{\alpha(1 - \rho\text{Re}(\lambda_{\mathbf{W}}))} \tag{5}$$

When the connectivity matrix, $\mathbf{W}$, is given by a sparse matrix with non-zero elements drawn randomly from a uniform distribution with the range $[-1, 1]$, then the corresponding eigenvalues will be uniformly distributed within a circle with a radius of $\max(|\lambda_{\mathbf{W}}|)$ in the complex plane [33]. These eigenvalues are then re-scaled by $\max(|\lambda_{\mathbf{W}}|)$ to ensure they are within the unit circle. The distribution of the eigenvalues then reveals the distribution of timescales of the linearised system. Indeed, given $p(\text{Re}(\lambda), \text{Im}(\lambda))$, the distribution of timescales can be found through computation of the marginal $p(\text{Re}(\lambda)) = \int p(\text{Re}(\lambda), \text{Im}(\lambda)) d\text{Im}(\lambda)$ and the change of variable defined in **Eq. 5**, giving

$$p(\tau) = \frac{2\delta t^2}{\pi\alpha^2\rho^2\tau^2}\sqrt{\alpha^2\rho^2 - (\alpha - \delta t/\tau)^2} \tag{6}$$

Importantly we note that while the eigenvalues are uniformly distributed over the unit circle, the timescales are not due to the inverse relationship between them. The resulting distribution of the linearised system, shown in **Figure 1B** (red line), is in excellent agreement with the numerically computed distribution for a single ESN (black points + shaded area).

The analytical form of the distribution, together with **Eq. 5**, allows us to explicitly derive how changes in α and ρ affect the network timescales. Notably we can obtain analytical expression for the minimum, maximum and most probable (peak of the distribution) timescale:

$$\tau_{\min} = \frac{\delta t}{\alpha(1+\rho)}, \tag{7}$$

$$\tau_{\max} = \frac{\delta t}{\alpha(1-\rho)}, \tag{8}$$

$$\tau_{\text{peak}} = \frac{5\delta t}{4\alpha(1-\rho^2)}\left(1 - \sqrt{1 - \frac{24}{25}(1-\rho^2)}\right) \tag{9}$$

where **Eqs. 8** and **7** can be derived directly from **Eq. 5**, while **Eq. 9** follows from maximisation of **Eq. 6**. As expected, α strongly

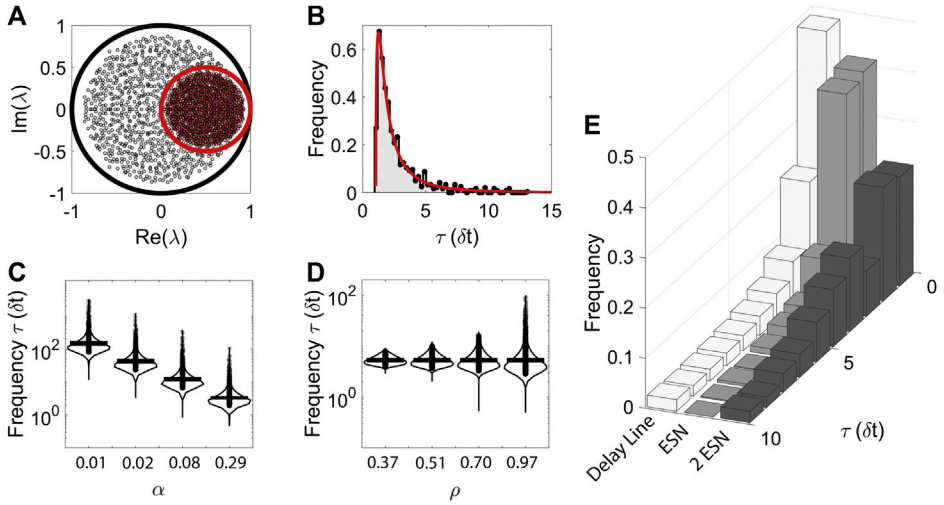

FIGURE 1 | The analysis of the timescales of the system in the linear regime can guide the search for the optimal values of the hyper-parameters α and ρ. **(A)**: Translation and scaling of the eigenvalues of the system due to the presence of the leakage factor. **(B)**: Example of distribution of timescales, computed analytically (red line) and computationally (black points) estimated from the eigenvalues of **W**. **(C)**: Pirate plot of the distributions of timescales as α increases. Both axes are logarithmic. Higher α values correspond to longer timescales and to a more compressed range of timescales (logarithmic *y*-axis). **(D)**: Pirate plot of the distributions of timescales: as ρ increases, the range of timescales expands. Again, both axes are logarithmic. **(E)**: Example distributions of timescales for reservoirs with different connectivity structure. From left to right, a delay line, single ESN, 2 ESNs (connected and unconnected, see text for the reason why the timescales for these two structures are the same in the linear regime). The higher complexity of the models reported is reflected in a richer distribution of timescales.

affects all these three quantities; interestingly, though, α does not influence the relative range of the distribution, $\tau_{max}/\tau_{min} = (1+\rho)/(1-\rho)$. Indeed α plays the role of a unit of measure for the τ's, and can then be used to scale the distribution in order to match the relevant timescales for the specific task. On the other hand, ρ does not strongly affect the shape of the distribution, but determines how dispersed the τ's are. Given the finite number of τ's expressed by a finite ESN, the hyper-parameter ρ can be used to balance the raw representation power of the network (how wide the range of timescales is) with the capacity to approximate any given timescale in that range. **Figures 1C,D** give a more detailed view of how the distribution of timescales changes as α and ρ, respectively, vary; note the logarithmic scale on the *y*-axis, that makes the dependence on α linear. The link between the eigenvalues and the reservoir dynamics can be shown through the analysis of the network response to an impulsive signal, shown in **Section 2 Supplementary Material** where the experimental activities are compared with the theoretical ones expected from the linearised system.

2.1 Hierarchical Echo-State Networks

Different studies have proposed alternatives to the random structure of the connectivity matrix of ESNs, formulating models of reservoirs with regular graph structures. Examples include a delay line [17], where each node receives and provides information only from the previous node and the following one respectively, and the concentric reservoir proposed in [18], where multiple delay lines are connected to form a concentric structure. Furthermore, the idea of a hierarchical architecture of ESNs, where each ESN is connected to the preceding and following one, has attracted the reservoir computing community for its capability of discovering higher level features of the external signal [34]. **Figure 2** schematically shows the architecture for **(A)** a single ESN, **(B)** 2 sub-reservoir hierarchical ESN for which the input is fed into only the first sub-reservoir which in turn feeds into the second and **(C)** a parallel ESN, where two unconnected sub-reservoirs receive the same input. These hierarchical ESNs are identical to the 2 layer DeepESN given by [27]. A general ensemble of interacting ESNs can be described by

$$\mathbf{x}^{(k)}(t+\delta t) = \left(1-\alpha^{(k)}\right)\mathbf{x}^{(k)} + \alpha^{(k)} f\left(\mathbf{h}^{(k)}(t)\right), \tag{10}$$

$$\mathbf{h}^{(k)}(t) = \gamma^{(k)}\mathbf{W}_{in}^{(k)}\mathbf{s}^{(k)}(t) + \sum_{l}^{N_{ESN}} \rho^{(kl)}\mathbf{W}^{(kl)}\mathbf{x}^{(l)}(t), \tag{11}$$

where the parameters have the similar definitions as in the case of a single ESN in **Eq. 1**. The index k indicates the network number and N_{ESN} is the total number of networks under consideration. In a hierarchical structure of ESNs $\mathbf{W}^{(kl)} \neq \mathbf{0}$ for $k = l$ or $k = l+1$ only, and $\mathbf{W}^{(kl)}$ can be drawn from any desirable distribution thanks to the absence of feedback connections to higher-order reservoirs. Indeed, in this case, the necessary condition for the Echo-State network property is that all the inner connectivity matrices $\mathbf{W}^{(kk)}$ have eigenvalues with an absolute value less than one. Furthermore, in the typical hierarchical structure proposed in previous works [23–25, 27, 35], the input is fed to the first network only, and $\mathbf{W}_{in}^{(k)} \neq \mathbf{0}$ if $k = 1$ only. We emphasise that the values of $\alpha^{(k)}$ and $\rho^{(k)}$, which are closely related to the timescales and repertoire of dynamics of network number k (and, in the case of hierarchical reservoirs, also to all subsequent networks), do not have to be equal for

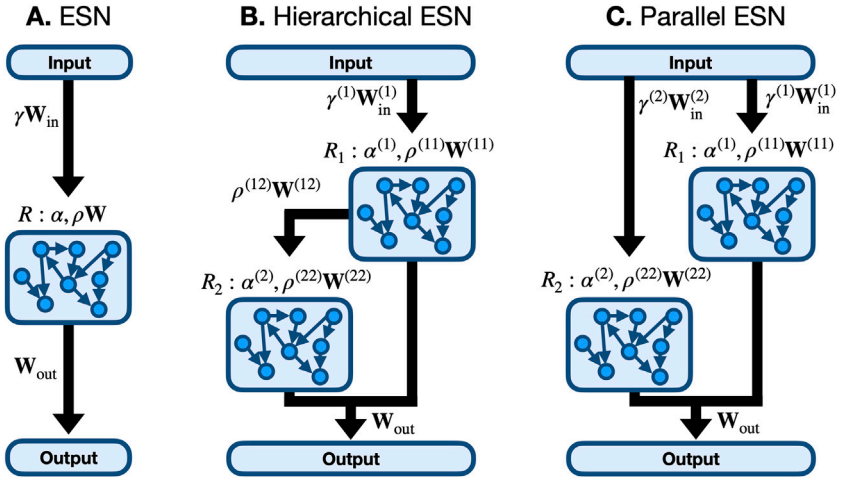

FIGURE 2 | Single and hierarchical echo-state network (ESN) architectures. **(A)**: A single ESN with internally connected nodes with a single set of hyper-parameters α and ρ. **(B)**: A hierarchical ESN composed of 2 connected reservoirs where the input is fed into reservoir one only and the connection is unidirectional from R1 to R2, which is identical to the 2 layer DeepESN of [27]. **(C)**: A parallel (or unconnected hierarchical) ESN where the network is partitioned into 2 reservoirs, R1 and R2, which each receive the input and provide output but have distinct hyper-parameters.

each ESN, but can be chosen differently to fit the necessity of the task. In particular, some tasks could require memory over a wide range of timescales that could not effectively be covered by a single ESN.

In **Figure 1E** we show examples of the timescale distributions of the corresponding linearised dynamical systems for different ESN structures, from the simple delay line model to the higher complexity exhibited from two hierarchical ESNs. In order from left to right, the histograms of timescales are for a delay line, a single ESN, and two ESNs (whether hierarchically connected or unconnected; see below for clarification). All the models share an ESN with $\rho = 0.9$ and $\alpha = 0.9$; where present, the second reservoir has $\alpha = 0.2$. By construction, the richness and range of timescales distributions reported increases with the complexity of the models. However, we note how a simple delay line could exhibit longer temporal scales than the other structures analyzed thanks to its constant and high value of minimum path length between any pairs of nodes. Nevertheless, its limited dynamics restricts its application to simple tasks. The cases with two ESNs show a bimodal distribution corresponding to the two values of α.

Yet, the spectrum of the eigenvalues of the linearised system is only partially informative of the functioning and capabilities of an ESN. This is clearly demonstrated by the fact that a hierarchical and a parallel ESN share the same spectrum in the linear regime. Indeed, for a hierarchical ESN, whose connectivity matrix of the linearised dynamics is given by:

$$W = \begin{bmatrix} \mathbf{W}^{(11)} & 0 \\ \mathbf{W}^{(21)} & \mathbf{W}^{(22)} \end{bmatrix}, \tag{12}$$

it is easy to demonstrate that every eigenvalue of $\mathbf{W}^{(11)}$ and $\mathbf{W}^{(22)}$ is also an eigenvalue of W, irrespective of $\mathbf{W}^{(21)}$, not unlike what happens for a parallel ESN (where $\mathbf{W}^{(21)} = \mathbf{0}$, and hence the demonstration follows immediately). Nonetheless, as we will see in the next sections, the hierarchical ESN has better performance on different tasks compared to the other structures considered, including the parallel ESN.

It is interesting to note, in this respect, that the success of the hierarchical ESN is generally achieved when the leakage term of the first reservoir is higher than the leakage term of the second (or, in other words, when the first network has much shorter timescales). Such observation opens the way to an alternative route to understand the functioning of the hierarchical structure, as the first reservoir expanding the dimensionality of the input and then feeding the enriched signal into the second network. Indeed, in the following, we will show how, in a crude approximation and under the above condition of a wide separation of timescales, the first ESN extracts information on the short term behavior of the input signal, notably its derivatives, and the second ESN integrates such information over longer times.

We begin with the (continuous time) linearized dynamics of a Hierarchical ESN given by

$$\dot{\boldsymbol{x}}^{(1)}(t) = -\mathbf{M}^{(1)}\mathbf{x}^{(1)}(t) + \mathbf{W}_{\text{in}}^{(1)}\mathbf{s}(t), \tag{13}$$

$$\dot{\boldsymbol{x}}^{(2)}(t) = -\mathbf{M}^{(2)}\mathbf{x}^{(2)}(t) + \mathbf{W}^{(21)}\mathbf{x}^{(1)}(t), \tag{14}$$

where, for simplicity, we have reabsorbed the $\rho^{(kl)}$ and $\gamma^{(k)}$ factors into the definitions of $\mathbf{W}^{(kl)}$ and $\mathbf{W}_{\text{in}}^{(k)}$ respectively, and the new constants can be derived with reference to **Eqs 1,2**; for example:

$$\mathbf{M}^{(k)} = \frac{\alpha^{(k)}}{\delta t}\left[\mathbf{I} - f'(0)\ \mathbf{W}^{(kk)}\right]. \tag{15}$$

The neuron activity can be projected on to the left eigenvector of each of the $\mathbf{M}^{(i)}$ matrices. As such we define the eigenvector matrices, $\mathbf{V}^{(i)}$, where each row is a left eigenvector and so satisfies the equation $\mathbf{V}^{(i)}\mathbf{M}^{(i)} = \mathbf{\Lambda}^{(i)}\mathbf{V}^{(i)}$. $\mathbf{\Lambda}^{(1)}$ and $\mathbf{\Lambda}^{(22)}$ are the diagonal

matrices of the eigenvalues of the two **M** matrices. Using these we can define $\mathbf{y}^{(k)} \equiv \mathbf{V}^{(k)}\mathbf{x}^{(\mathbf{k})}$, and so the dynamical equations can be expressed as

$$\dot{\boldsymbol{y}}^{(1)}(t) = -\boldsymbol{\Lambda}^{(1)}\mathbf{y}^{(1)}(t) + \tilde{\mathbf{W}}_{\text{in}}^{(1)}\mathbf{s}(t), \tag{16}$$

$$\dot{\boldsymbol{y}}^{(2)}(t) = -\boldsymbol{\Lambda}^{(2)}\mathbf{y}^{(2)}(t) + \tilde{\boldsymbol{W}}^{(21)}\mathbf{y}^{(1)}(t), \tag{17}$$

where $\tilde{\boldsymbol{W}}_{\text{in}}^{(1)} = \mathbf{V}^{(1)}\mathbf{W}_{\text{in}}^{(1)}$ and $\tilde{\boldsymbol{W}}^{(21)} = \mathbf{V}^{(2)}\mathbf{W}^{(21)}\ (\mathbf{V}^{(1)})^{-1}$ are the input and connection matrices expanded in this basis. Taking the Fourier transform on both sides of **Eq. 16**, such that $FT[\mathbf{y}^{(1)}(t)] = \tilde{\boldsymbol{y}}^{(1)}(\omega)$ and $FT[\dot{\boldsymbol{y}}^{(1)}(t)] = -i\omega\tilde{\boldsymbol{y}}^{(1)}(\omega)$, where i is the imaginary unit. The transform $\tilde{\boldsymbol{y}}^{(2)}(\omega)$ of $\mathbf{y}^{(2)}(t)$ can now be expressed as a function of the transform of the signal $\tilde{\mathbf{s}}(\omega)$ giving

$$\left(\boldsymbol{\Lambda}^{(1)} - i\omega\mathbf{I}\right)\tilde{\boldsymbol{y}}^{(1)}(\omega) = \tilde{\boldsymbol{W}}_{\text{in}}^{(1)}\tilde{\mathbf{s}}(\omega) \tag{18}$$

where **I** is the identity matrix of the same size as $\boldsymbol{\Lambda}^{(1)}$. If the second ESN's timescale are much longer than that of the first one (i.e., $\boldsymbol{\Lambda}^{(1)} \gg \boldsymbol{\Lambda}^{(2)}$), then we can expand the inverse of the $\tilde{\boldsymbol{y}}^{(1)}$ coefficient on the LHS of **Eq. 18** when $\boldsymbol{\Lambda}^{(1)} \to \infty$ as

$$\left(\boldsymbol{\Lambda}^{(1)} - i\omega\mathbf{I}\right)^{-1} = \left(\boldsymbol{\Lambda}^{(1)}\right)^{-1}\left(1 - i\omega\left(\boldsymbol{\Lambda}^{(1)}\right)^{-1}\right)^{-1}, \tag{19}$$

$$\approx \left(\boldsymbol{\Lambda}^{(1)}\right)^{-1}\sum_{n=0}^{\infty}\left(i\omega\left(\boldsymbol{\Lambda}^{(1)}\right)^{-1}\right)^{n}, \tag{20}$$

By applying this approximation to **Eq. 18**, and by defining the diagonal matrix of characteristic times $\mathbf{T}^{(1)} \equiv -\left(\boldsymbol{\Lambda}^{(1)}\right)^{-1}$, the relation between the activity of reservoir one and the input in Fourier space is given by

$$\tilde{\boldsymbol{y}}^{(1)}(\omega) = -\mathbf{T}^{(1)}\sum_{n=0}^{\infty}\left(-i\omega\mathbf{T}^{(1)}\right)^{n}\tilde{\mathbf{W}}_{\text{in}}^{(1)}\tilde{\mathbf{s}}(\omega). \tag{21}$$

The coefficients of this series are equivalent to taking successive time derivatives in Fourier space, such that $(-i\omega)^n\tilde{\mathbf{s}} = d^{(n)}\tilde{\mathbf{s}}/dt^{(n)}$. So by taking the inverse Fourier transform we find the following differential equation for $\boldsymbol{y}^{(1)}$

$$\mathbf{y}^{(1)}(t) = -\mathbf{T}^{(1)}\sum_{n=0}^{\infty}\left(\mathbf{T}^{(1)}\right)^{n}\tilde{\mathbf{W}}_{\text{in}}^{(1)}\frac{d^{(n)}\mathbf{s}(t)}{dt^{(n)}}, \tag{22}$$

which can be inserted into **Eq. 17** to give

$$\dot{\boldsymbol{y}}^{(2)} = \boldsymbol{\Lambda}^{(2)}\mathbf{y}^{(2)} - \tilde{\boldsymbol{W}}^{(21)}\boldsymbol{T}^{(1)}\left[\tilde{\boldsymbol{W}}_{\text{in}}^{(1)}\mathbf{s}(t) + \sum_{n=1}^{\infty}\left(\mathbf{T}^{(1)}\right)^{n}\tilde{\boldsymbol{W}}_{\text{in}}^{(1)}\frac{d^{(n)}\mathbf{s}(t)}{dt^{(n)}}\right]. \tag{23}$$

Thus the second ESN integrates the signal with a linear combination of its derivatives. In other words, the first reservoir expands the dimensionality of the signal to include information regarding the signal's derivatives (or, equivalently in discretized time, the previous values assumed by the signal). In this respect, **Eq. 23** is key to understanding how the hierarchical connectivity between the two reservoirs enhances the representational capabilities of the system. The finite-difference approximation of the time derivatives appearing in **Eq. 23** implies that a combination of past values of the signal appears, going back in time as much as the retained derivative order dictates.

2.2 Online Learning of Hyper-Parameter

Selecting the hyper-parameters of such systems can be challenging. Such selection process can be informed by the knowledge of the natural timescales of the task/signal at hand. Alternatively one can resort to a learning method to optimise the parameters directly. The inherent limitation of these methods is the same as learning the network weights with BPTT: the whole history of network activations is required at once. One way to bypass this issue is to approximate the error signal by considering only past and same-time contributions, as suggested by Bellec *et al.* [4] in their framework known as *e-prop* (see also [36]), and derive from this approximation an online learning rule for the ESN hyper-parameters. Following their approach, we end up with a novel learning rule for the leakage terms of connected ESNs that is similar to the rule proposed by Jaeger *et al.* [32] but extended to two hierarchical reservoirs. The main learning rule is given by:

$$\frac{dE}{d\alpha^{(i)}}(t) = \sum_{k=1}^{N_{\text{ESN}}}\frac{\partial E}{\partial\mathbf{x}^{(k)}(t)}\mathbf{e}^{(ki)}(t), \tag{24}$$

where $\mathbf{e}^{(ki)}(t) = d\mathbf{x}^{(k)}(t)/d\alpha^{(i)}$ is known as the eligibility trace which tracks the gradient of neuron activities in the reservoir number k with respect to the ith leakage rate. Given the closed form for the hierarchical ESNs in **Eqs 10,11** these terms can be readily calculated. For our N_{ESN} sub-reservoirs in the hierarchical structure there will be N_{ESN}^2 eligibility traces to track how each sub-reservoir depends on the other leakage rates. In the hierarchical case of a fixed feed-forward structure some of these traces will be zero, and the number of non-zero eligibility traces would be $N(N+1)/2$. Since the update of the neuron's activity depends on its previous values, so do the eligibility traces; therefore, they can be calculated recursively through

$$\mathbf{e}^{(ki)}(t+\delta t) = \left(1-\alpha^{(k)}\right)\mathbf{e}^{(ki)}(t) + \delta_{ki}\left(f\left(\mathbf{h}^{(k)}(t)\right) - \mathbf{x}^{(k)}(t)\right) + \alpha^{(k)}f'\left(\mathbf{h}^{(k)}(t)\right)\sum_{l\neq k}\rho^{(kl)}\mathbf{W}^{(kl)}\mathbf{e}^{(li)}(t), \tag{25}$$

where $\delta_{ki} = 1$ if $k = i$ and 0 otherwise, i. e the Kronecker delta. The update of **Eq. 25** for each k-i pair needs to follow the order of dependencies given by the structure of connected reservoirs considered. The eligibility trace is an approximation that only includes same-time contributions to the gradient but has the advantage that is can be easily computed online. A complete description of our method is given in the **Supplementary Material**. For an example where the mean squared error function $E(t) = \frac{1}{2}[\tilde{y}(t) - y(t)]^2$ is used in a regression task and a structure composed by two reservoirs, the updating equations on the leakage terms are

$$\begin{aligned}\alpha^{(1)} &\leftarrow \alpha^{(1)} - \eta_\alpha[\tilde{y}(t) - y(t)]\mathbf{W}_{\text{out}}\begin{pmatrix}\mathbf{e}^{(11)}(t)\\ \mathbf{e}^{(12)}(t)\end{pmatrix}\\ \alpha^{(2)} &\leftarrow \alpha^{(2)} - \eta_\alpha[\tilde{y}(t) - y(t)]\mathbf{W}_{\text{out}}\begin{pmatrix}\mathbf{e}^{(21)}(t)\\ \mathbf{e}^{(22)}(t)\end{pmatrix}\end{aligned} \tag{26}$$

where η_α is the learning rate on the leakage terms and $(\mathbf{e}^{(k1)}(t), \mathbf{e}^{(k2)}(t))$ ($k = 1, 2$ in this case with two reservoirs) is a vector composed by the juxtaposition of the eligibility traces, which can be computed through **Eq. 25**. Of course, the gradient can be combined with existing gradient learning techniques, among which we adopt the Adam optimiser, described in the **Supplementary Material**. In all online learning simulations, training is accomplished through minibatches with updates at each time step. Training is stopped after convergence. When learning α′s and the output weights simultaneously, the learning rates corresponding to these hyper-parameters need to be carefully set, since the weights need to adapt quickly to the changing dynamic of the network, but a fast convergence of $\mathbf{W}_{\text{out}}$ can trap the optimisation process around sub-optimal values of the leakage terms. For a reservoir with trained and converged output weights, a further variation of α′s, even in the right direction, could correspond to an undesirable increase in the error function. We found that this problem of local minimum can be avoided by applying a high momentum in the optimisation process of α and randomly re-initialising the output weights when the α′s are close to convergence. The random re-initialisation functions to keep the output weights from being too close to convergence. Thus, we defined the convergence of the algorithm for α′s as when the α′s do not change considerably after re-initialisation. When this happens, it is possible to turn off the learning on the leakage terms and to optimise the read-out only. More details about online training can be found in the discussions related to each task.

3 RESULTS

The following sections are dedicated to the study of the role of timescales and the particular choices of α and ρ in various tasks, with attention on networks composed by a single ESN, 2 unconnected ESNs and 2 hierarchical ESNs. The number of trainable parameters in each task for the different models will be preserved by using the same total number of neurons in each model. The results analyzed will be consequently interpreted through the analysis of timescales of the linearised systems.

3.1 NARMA10

A common test signal for reservoir computing systems is the non-linear auto-regressive moving average sequence computed with a 10 step time delay (NARMA10) [37, 38]. Here we adopt a discrete time formalism where $n = t/\delta t$ and the internal state of the reservoir is denoted as $\mathbf{x}_n = \mathbf{x}(n\delta t)$. The input, s_n, is a uniformly distributed random number in the range $[0, 0.5]$ and the output time-series is computed using

$$y_n = y_{n-1}\left(a + b\sum_{k=1}^{D} y_{n-k}\right) + cs_{n-1}s_{n-D} + d, \tag{27}$$

where $D = 10$ is the memory length, $a = 0.3$, $b = 0.05$, $c = 1.5$, and $d = 0.1$. The task for the network is to predict the NARMA10 output y_n given the input s_n. We have adapted this to also generate a NARMA5 task where $D = 5$ but the other parameters are unchanged. This provides an almost identical task but with different timescales for comparison.

The task of reconstructing the output of the NARMA10 sequence can be challenging for a reservoir as it requires both a memory (and average) over the previous 10 steps and fast variation with the current input values to produce the desired output. A typical input and output signal is shown in **Figure 3A** and the corresponding auto-correlation function of the input and output in B. Since the input is a random sequence it does not exhibit any interesting features but for the output the auto-correlation shows a clear peak at a delay of 9 δt in accordance with the governing equation. For a reservoir to handle this task well it is necessary to include not only highly non-linear dynamics on a short timescale but also slower dynamics to handle the memory aspect of the task.

This regression task is solved by training a set of linear output weights to minimise the mean squared error (MSE) of the network output and true output. The predicted output is computed using linear output weights on the concatenated network activity ($\mathbf{x}_n = (\mathbf{x}_n^{(1)}, \mathbf{x}_n^{(2)})^T$), such that

$$\tilde{y}_n = \mathbf{x}_n^T \mathbf{W}_{\text{out}} \tag{28}$$

where $\mathbf{W}_{\text{out}}$ is the weight vector of length N+1 when an additional bias unit is included. The MSE is minimised by using the ridge regression method [39] such that the weights are computed using

$$\mathbf{W}_{\text{out}} = \left(\mathbf{x}^T\mathbf{x} - \lambda\mathbf{I}\right)^{-1}\mathbf{x}^T\mathbf{y} \tag{29}$$

where $\mathbf{x}$ is a matrix formed from the activation of the internal states with a shape of number of samples by number of neurons, $\mathbf{y}$ is the desired output vector, λ is the regularisation parameter that is selected using a validation data set and $\mathbf{I}$ the identity matrix. To analyze the performance of the ESNs on the NARMA10 task we use the normalised root mean squared error as

$$\text{NRMSE} = \sqrt{\frac{1}{N_s}\frac{\sum_n^{N_s}\left(\tilde{y}_n - y_n\right)^2}{\text{Var}(\mathbf{y})}}, \tag{30}$$

where $\tilde{y}_n$ is the predicted output of the network and y_n is the true output as defined by **Eq. 27**.

To test the effectiveness of including multiple time-scales in ESNs, we simulate first a single ESN with $N = 100$ neurons and vary both α and ρ to alter the time-scale distribution. Secondly, we simulate a hierarchical ESN split into 2 reservoirs each with $N = 50$ neurons, where we vary $\alpha^{(1)}$ and $\alpha^{(2)}$ with $\rho^{(1)} = \rho^{(2)} = 0.95$. The input factor was set as $\gamma^{(1)} = 0.2$ and $\gamma^{(2)} = 0$ for the connected hierarchical ESN but when they are unconnected the input is fed into both, such that $\gamma^{(1)} = \gamma^{(2)} = 0.2$. In all cases the NRMSE is computed on an unseen test set and averaged over 20 initialisations of the ESN with a running median convolution is applied to the error surfaces to reduce outliers. In parallel to this we have also applied the online training method for the α hyper-parameters. The hyper-parameters used for the gradient descent learning are summarised in **Table 1**.

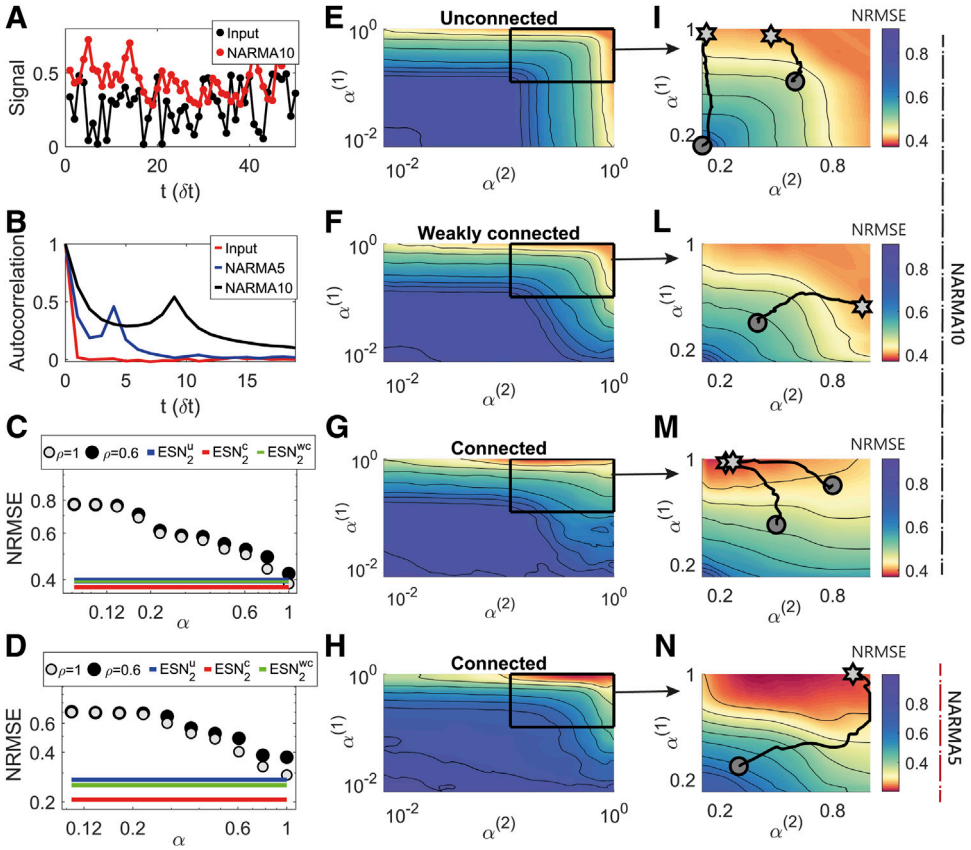

FIGURE 3 | Performance of single or hierarchical ESNs on the NARMA10 and NARMA5 task. **(A)**: Example input signal (black) and desired output (red) for the NARMA10 task. **(B)**: The auto-correlation function of the (black) input (red) NARMA10 and (blue) NARMA5 desired output signals, showing a second peak at about 9 delay steps for the NARMA10 and 4 for the NARMA5. **(C)**: The NRMSE for a single ESN for with $\rho = 1.0$ and 0.63 over a range of α. The NRMSE is lower for $\rho \approx 1$ and $\alpha = 1$. The solid lines show the minimum NRMSE for the unconnected (blue line) and connected (red line); for the unconnected case the minimum NRMSE is similar to the single ESN while the connected case has a smaller NRMSE by about 10%. **(D)**: Average NRMSE of a single ESN for various α compared to the hierarchical ESNs for the NARMA5 task. **(E–K):** The average NRMSE surface using a hierarchical ESN computed for varying the leakage rates $\alpha^{(k)}$ of both the reservoir components for **(E)** and **(I)** (no coupling, $\rho^{(21)} = 0$), **(F)** and **(J)** (weak coupling, $\rho^{(21)} = 0.1$), and **(G)** and **(K)** (strong coupling, $\rho^{(21)} = 1$). Panels (**I-L**) show a close up in region for the range $\alpha^{(k)} = [0.1, 1]$ to highlight the changing behaviors. The lines on these panels show the trajectory of the $\alpha^{(k)}$ values trained directly using the online method. For each case of the coupling the online learning trends toward the approximate error minimum. **(H)** shows the NRMSE surface for the NARMA5 task using a strongly connected hierarchical ESN, with **(N)** again showing a zoom of the $\alpha = [0.1, 1]$ region. The region of best performance is with $\alpha^{(2)} \approx 0.5$ which matches the shorter timescale demonstrated in the auto-correlation in **(B)**.

Figures 3E–G and I–M show the NRMSE depending on $\alpha^{(1)}$ and $\alpha^{(2)}$ for 3 variations of the hierarchical ESN connection strength on the NARMA10 task. In the unconnected case ($\rho^{(21)} = 0$, panels E and I), we find that the NRMSE drops by increasing both leakage rates but the minimum is when one of the leakage rates is ≈ 0.5. This is in agreement with the online learning method for the α's in shown in I but the error minimum is shallow and prone to noise in the signal or ESN structure. For the weakly connected hierarchical ESN ($\rho^{(21)} = 0.1$, panels F and J) we find again that when the sub-reservoirs have different timescales the NRMSE is reduced. In comparison to the unconnected case the error surface is asymmetric with a minimum at approximately $\alpha^{(1)} = 1.0$ and $\alpha^{(2)} \approx 0.5$. As the strength of the connection is increased ($\rho^{(21)} = 1.0$, Panel G and K), the minimum error moves to a lower leakage rate in the second reservoir ($\alpha^{(2)} \approx 0.2$) which reflects a better separation of the timescale distributions. This is a gradual effect with respect to the connection strength since stronger connection allows for a relative increase of the expanded input from the first reservoir compared to the base input signal. Since the input feeds into reservoir 1, a high α provides a transformation on the input over short time-scales, expanding the dimensionality of the signal, offering a representation that preserves much of the dynamic of the driving input and that is fed to the second reservoir. Then, since the latter does not have a direct connection to the input it performs a longer timescale transformation of the internal states of reservoir 1. In this way the reservoirs naturally act on different parts of the task, i.e., reservoir one provides a fast non-linear transformation of the input while reservoir 2 follows the slower varying 10-step average of the signal, and thus returning a lower NRMSE. As a side note, we can demonstrate the validity of the theoretical analysis in **Section 2.1** by replacing the first reservoir by **Eq. 23** on the NARMA task (see

TABLE 1 | Table of the hyper-parameters adopted in the online learning process. η is the learning rate in each case, while β_1, β_2 and ϵ are parameters for the Adam optimiser (further details are given in the **Supplementary Material**).

Learning hyper-parameters		
	NARMA/Telegraph	**psMNIST**
Network size N	100	1200
Minibatch size N_{batch}	10	50
	Learning $\mathbf{W}_{out}$	
η_W	10^{-3}	10^{-3} [10^{-4}][a]
β_1	0.9	0.9
β_2	0.999	0.999
ϵ	10^{-8}	10^{-8}
	Learning α	
η_α	5×10^{-6}	10^{-3}
β_1	0.99	0.999
β_2	0.999	0.999
ϵ	10^{-8}	10^{-8}

[a]*symbol indicates that the learning rate 10^{-3} is for the case with 4 hidden states, while the learning rate [10^{-4}] is for the case with 28 hidden states. This decrease of η is due to the increase in the dimensionality of the representation for the latter case in comparison to the situation where the read-out is composed by four concatenated values of activity. Furthermore, such learning rates are 10 times higher than the case in which only the read-out is trained (only in the psMNIST task). Thus, the high learning rate adopted has the purpose to introduce noise in the learning process and to avoid local minima in the complex case where α and $\mathbf{W}_{out}$ are optimised simultaneously.*

Section 3 Supplementary Material), resulting in a similar landscape as in **Figure 3G** and a similar optimal value for $\alpha^{(2)}$.

Figure 3C shows the relative performance of the single ESN to the minimum values for the unconnected (ESN_2^u) and connected (ESN_2^c) hierarchical reservoirs. The single ESN shows the similar decrease in NRMSE with increasing α and reaches a similar minimum NRMSE as the unconnected case. In comparison with the connected cases the multiple timescales provides a more optimised result. If we consider the analysis of the timescales discussed in the previous section the choice of these hyper-parameters becomes more evident. With $\alpha = 1$ the timescale distribution of the network is sharply peaked close to the minimum timescale of one discrete step while when $\alpha = 0.1$ this peak is broader and the peak of the distribution is closer to the second peak present in the auto-correlation function shown in Panel **B**. We note that while the most likely timescale is $\tau_{peak} \approx 6$ for $\alpha = 0.1, \rho = 0.95$ which is lower than the natural timescale of the problem, the increased width of the distribution increases the number of timescales at $\tau = 10$ dramatically which maybe why a lower α is not necessary.

To further investigate the effect of the inherent timescale of the task on the timescales we performed a similar analysis on the NARMA5 task. **Figures 3H,L** show the NRMSE surface for the strongly connected case. The minimum error occurs at $\alpha^{(1)} \approx 1.0$ (similar to the NARMA10 results in G and K) but $\alpha^{(2)} \approx 0.5$ (as opposed to ≈ 0.2 for NARMA10). This is due to the shorter timescales required by the NARMA5 task and the peak timescale for these values is much closer to the peak in the auto-correlation shown in **B**. Panel D shows the performance of the single ESN where again the optimal leakage rate is $\alpha = 1$ and similar to the unconnected cases but the NRMSE is higher than the connected cases.

In this theoretical task where the desired output is designed *a priori*, the memory required and the consequent range of timescales necessary to solve the task are known. Consequently, considering the mathematical analysis in **Section 2.1**, and that for hierarchical ESNs the timescales of the first ESN should be faster than those of the second **Figure 3**), the best-performing values of the leakage terms can be set *a priori* without the computationally expensive grid search reported in **Figures 3E–L**. However, it can be difficult to guess the leakage terms in the more complex cases where the autocorrelation structure of the signal is only partially informative of the timescales required.

This problem can be solved using the online learning approach defined through **Eq. 24**. In this case, learning is accomplished through minibatches and the error function can be written explicitly as

$$E(t) = \frac{1}{2N_{batch}} \sum_{m=1}^{N_{batch}} \left[\tilde{y}(t,m) - y(t,m)\right]^2 \tag{31}$$

where N_{batch} is the minibatch size and m is its corresponding index. A minibatch is introduced artificially by dividing the input sequence into N_{batch} signals or by generating different NARMA signals. Of course, the two methods lead to equivalent results if we assure that the N_{batch} sequences are temporally long enough. A learning rate $\eta_\alpha/\eta_W \approx 10^{-2} - 10^{-3}$ was adopted. The optimiser used for this purpose is Adam, with the suggested value of $\beta_1 = 0.9$ adopted for the output weights and a higher first momentum $\beta_1 = 0.99$ adopted for the leakage terms. Instead, we set $\beta_2 = 0.999$ of the second momentum for both types of parameters (See **Section 2.2** for a description of the updating rules). Panels I–L show a zoomed in region of the error surface with the lines showing the online training trajectory of the α hyper-parameters. In each case the trajectory is moving toward the minimum NRMSE of the α phase space.

3.2 A Volatile Environment

We now turn to study the reservoir performance on a task of a telegraph process in a simulated noisy environment. The telegraph process $s^{(1)}(t)$ has two states that we will call *up* 1) and *down* (0), where the probability of going from a *down* state to an *up* state $p(s = 1|s = 0)$ (or the opposite $p(s = 0|s = 1)$) is fixed for any time step. The environment is also characterised by a telegraph process $s^{(2)}(t)$, but the transition probability is much lower and controls the transition probability of the first signal. To simplify the notation in the following we denote the probability of the signal i transitioning from state a to state b as $p(s^{(i)}(t) = a|s^{(i)}(t - \delta t) = b) = p_{ab}^{(i)}(t)$. The signal taken under consideration is then composed by a fast telegraph process with probabilities $p_{01}^{(1)}(t)$ and $p_{10}^{(1)}(t)$, whose values are interchanged by following the dynamic of a slower telegraph process $s^{(2)}(t)$. Every time the slower environment signal changes its state, the probabilities of the first signal are changed, i. e., $p_{01}^{(1)}(t) \leftrightarrow p_{10}^{(1)}(t)$. The resulting signal is then characterised by

$$p_{10}^{(1)}(t) = \begin{cases} p_1, & \text{if } s^{(2)}(t) = 0 \\ p_2, & \text{if } s^{(2)}(t) = 1 \end{cases} \tag{32}$$

$$p_{01}^{(1)}(t) = \begin{cases} p_2, & \text{if } s^{(2)}(t) = 0 \\ p_1, & \text{if } s^{(2)}(t) = 1 \end{cases} \tag{33}$$

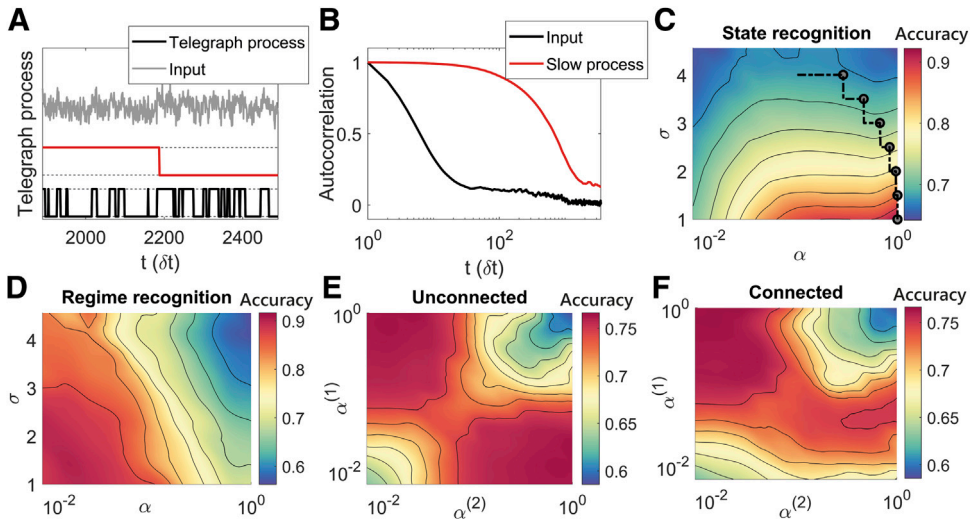

FIGURE 4 | The best structure and parameters of the model depend on the specific environment considered, that is different values of the additive noise in the input signal, and on the specific desired output. **(A)**: Example of input signal and of its generative processes, which have a faster and a slower dynamic respectively. When the slower process (red line) is *up* (*down*), the other signal is in a regime where the average time in the zero (one) state is greater than the average time spent in the other state. The input signal (gray line) corresponds to the faster process (black line) with additional white noise. **(B)**: Auto-correlation structure of the two generative processes. **(C)**: The accuracy surface for a single ESN on the state recognition sub-task for varying level of noise (σ) and leakage rate of the network showing that for increasing levels of noise a lower leakage rate is needed to determine the state. The line shows the trajectory of α using the online learning method when the strength of the noise is changed. **(D)**: The accuracy for a single ESN on the regime recognition sub-task for varying noise and leakage rate. In this case the low leakage rate is preferred for all values of noise. **(E)**: Accuracy surface for the state recognition sub-task for an unconnected hierarchical ESN showing how either of the leakage rates must be low while the other is high. **(F)**: Accuracy surface for the regime recognition sub-task for a hierarchical ESN showing the first reservoir must have a high leakage rate and the second a low leakage rate.

The transition probabilities of the second signal are fixed and symmetric such that

$$p_{01}^{(2)}(t) = p_{10}^{(2)}(t) = p_3, \tag{34}$$

The probabilities p_1, p_2 and p_3 are fixed parameters of the signal that define the process. Given that the second signal controls the probabilities of the first telegraph process, we say that it defines the regime of the input, while we refer to the *up* and *down* values of the first process simply as states. Thus, the reconstruction of $s^{(1)}(t)$ from the input will be called state reconstruction, while reconstruction of $s^{(2)}(t)$ will be called regime reconstruction. These reconstructions can be considered separately or as a joint task requiring the system to be modeled on long and short timescales simultaneously. Due to the probability transition caused by $s^{(2)}(t)$, both states and regime will be equally present over a infinitely long signal. The values adopted for the simulation are $p_1 = 0.05$, $p_2 = 0.1$ and $p_3 = 0.0005$.

The input signal corresponds to $s^{(1)}(t) + \sigma\mathcal{N}(0,1)$, that is the faster telegraph process with additional white noise. The input signal constructed is a metaphor of a highly stochastic environment with two states and two possible regimes that define the probability of switching between the two states. The reservoir will be asked to understand in which state ($s^{(1)}(t) = 1$ or 0) and/or regime ($s^{(2)}(t) = 1$ or 0) it is for each time t, measuring the understanding of the model to estimate the state of the input signal. The input signal and telegraph processes is shown in **Figure 4A**, while B shows the corresponding auto-correlation structure of the processes. The auto-correlation shows that the input has a temporal structure of around 10 δt while the slow 'environment' process has a structure close to 1000 δt. This corresponds directly to the timescales defined by the probabilities of the signals.

Panels C and D of **Figure 4** show the performance of a single ESN when it is tasked to reconstruct the processes $s^{(1)}(t)$ (state recognition) and $s^{(2)}(t)$ (regime recognition) respectively. In this simulation, learning is always accomplished online and the error function is the same as **Eq. 31**. First, panel C demonstrates how the leakage term, α, must be tuned to the level of noise of the environment, and how lower values of α are desirable for noisier signals, in order to solve the state recognition problem. Indeed, the need to smooth the fluctuations of the input signal increases with σ, while for low values of noise the network should simply mimic the driving input. Second, panel D shows how the desirable values of α must be lower in the case where the network is asked to reproduce the slower dynamic of $s^{(2)}(t)$ independently of having to output the fast signal, in order to solve the regime recognition problem. This result exemplifies how the timescales of the network must be tuned depending on the desired output. It demonstrates that, even in this relatively simple environment, it is crucial to adopt multiple timescales in the network to obtain results that are robust with respect to a variation of the additional white noise σ.

Finally, panels E and F of **Figure 4** show the accuracy of two unconnected (E) and connected (F) reservoirs when the network

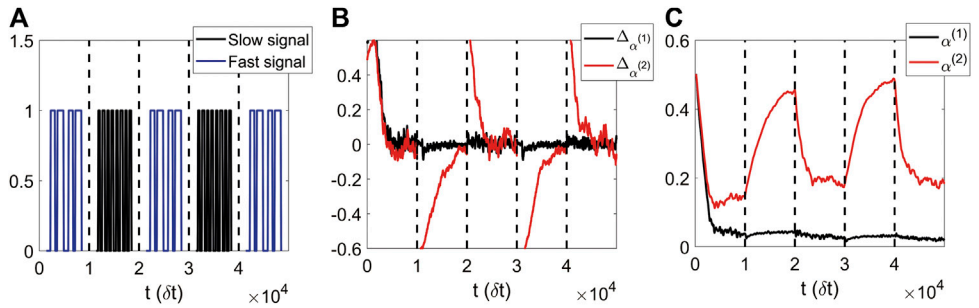

FIGURE 5 | The online training of the leakage terms can adapt to the changing environment, that is the signal probabilities are increased or decreased periodically. **(A)**: Scheme of the change of the values of probabilities, where high probabilities of switching are referred to as fast phase of the telegraph process, while low probabilities as slow phase. **(B)**: Running average of the gradients of $\alpha^{(1)}$ and $\alpha^{(2)}$ as time varies. **(C)**: Online adaptation of the leakage terms.

has to classify the state and the regime of the input signal at the same time. In this case, the desired output corresponds to a four dimensional signal that encodes all the possible combinations of states and regimes; for instance, when the signal is in the state one and in the regime one, we would require the first dimension of the output to be equal to one and all other dimensions to be equal to zero, and so on. The best performance occurs when one leakage term is high and the other one is low and in the range of significant delays of the auto-correlation function. This corresponds to one network solving the regime recognition and the other network solving the state recognition. For the unconnected reservoirs, it does not matter which reservoir has high vs. low leakage terms, reflected by the symmetry of **Figure 4E**, while for the connected reservoirs, the best performance occurs when the first reservoir has the high leakage term and the second the low leakage terms, see **Figure 4F**, similar to **Figure 3**. Both two-reservoir networks can achieve accuracy 0.75, but the single ESN can not solve the task efficiently, since it cannot simultaneously satisfy the need for high and low αs, reporting a maximum performance of about 0.64.

The path reported in panel C of **Figure 4** and all panels in **Figure 5** show the application of the online training algorithm in this environment. The values of the hyper-parameters adopted in the optimisation process through the Adam optimiser are the same as in **Section 3.1**, where we used a slower learning rate and a higher first momentum on the leakage terms in comparison to the values adopted for the output weights. The line of panel C (**Figure 4**) shows the online adaptation of α for a simulation where the external noise increases from one to four with six constant steps of 0.5 equally spaced across the computational time of the simulation. The result shows how the timescales of the network decrease for each increase in σ, depicted with a circle along the black line. The path of online adaptation reports a decrease of the α value for noisier external signals. This result occurs because as the signal becomes noisier (σ rises), it becomes more important to dampen signal fluctuations. This result also shows that the online algorithm can adapt in environments with varying signal to noise ratio. **Figure 5** shows the online training of $\alpha^{(1)}$ and $\alpha^{(2)}$ for an environment composed by a faster and a slower composition of telegraph processes. This specific simulation is characterised by the alternation of two signals defined by **Eqs 32**, **33** and **34**, each with different values of p_1 and p_2. In particular, while $p_1 = 0.5$ and $p_2 = 0.1$ for the 'fast' phase of the external signal, $p_1 = 0.1$ and $p_2 = 0.05$ for the "slow" phase. In contrast, the slower timescale of the task defined by $p_3 = 0.0005$ remains invariant across the experiment. Panel C shows the adaptation of the leakage terms for this task in the case of a hierarchical structure of ESNs. While $\alpha^{(2)}$ adapts to the change of p_1 and p_2 following the transition between the two phases of the external signals, the relatively constant value of $\alpha^{(1)}$ indicates how the first network sets its timescales to follow the slower dynamic of the signal, characterised by the constant value of p_3. Thus, the composed network exploits the two reservoirs separately, and the first (second) reservoir is used to represent the information necessary to recognise the regime (state) of the external signal.

3.3 Permuted Sequential MNIST

The Permuted Sequential MNIST (psMNIST) task is considered a standard benchmark for studying the ability of recurrent neural networks to understand long temporal dependencies. The task is based on the MNIST dataset, which is composed of 60,000 handwritten digits digitised to 28×28 pixel images. In the standard MNIST protocol every pixel is presented at the same temporal step so a machine has all the information of the image available at once and needs to classify the input into one out of ten classes. In contrast, in the psMNIST task, the model receives each pixel sequentially once at a time, so that the length of the one dimensional input sequence is 784. Thus, the machine has to rely on its intrinsic temporal dynamic and consequent memory ability to classify the image correctly. Furthermore, each image in the dataset is transformed through a random permutation of its pixels in order to include temporal dependencies over a wide range of input timescales and to destroy the original images' structure. Of course, the same permutation is applied on the entire dataset. The performance of ESNs on the MNIST dataset, where each columns of pixels in a image is fed to the network sequentially (each image corresponds to a 28

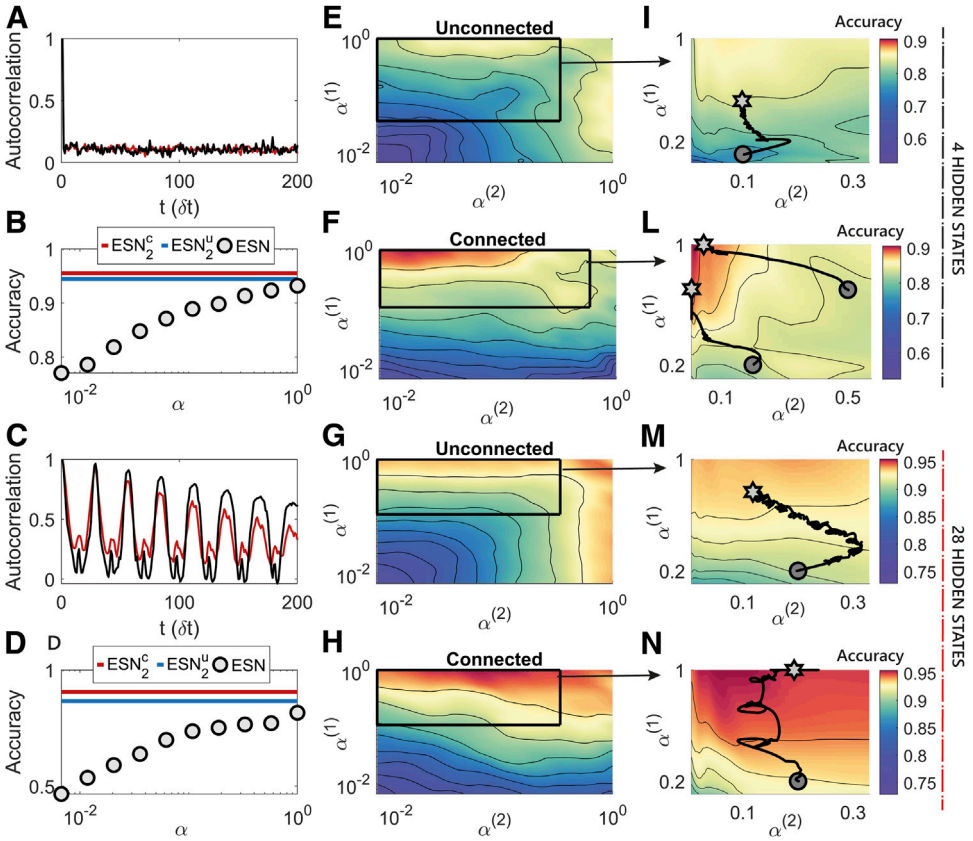

FIGURE 6 | The additional non linearity added by the hierarchical reservoir structure is responsible for a relevant modification and increase of the performance surface. **(A,C)**: Auto-correlation structure of the MNIST dataset for two examples of digits, where each pixel is presented one after the other **(C)**, and auto-correlation structure of the data after the random permutation **(A)**. The oscillatory trend in **C** reflects the form of the written digits, when this is seen one pixel after the other. The auto-correlation function of the permuted data is low, but not negligible, for all the temporal steps, showing the necessity to have a wide repertoire of timescales in the interval corresponding to the image size. **(B,D)**: Accuracy of a single ESN for various α values compared the maximum accuracy of the hierarchical ESNs with 4 hidden states **(B)** or 28 hidden states **(D)**. **(E–F)**: case with low sampling frequency of the ESNs which corresponds to a higher demand of internal memory in the reservoir. While the best region of accuracy for the unconnected reservoirs is characterised by intermediate values of the leakage factors, the hierarchically connected network structure reports the best performance when the second network has slower dynamics. **(G–H)**: The utilisation of a high sampling frequency alleviates the need for long term memory, and the reservoirs prefer the regions with fast timescales. In both cases analyzed, the additional complexity of the hierarchical model leads to a considerable boost in performance. **(I–N)**: Paths (black line, starting from the circle and ending in the star) that describe the online changes of the leakage terms achieved through the online training algorithm in a zoomed region of the performance surface of $\alpha^{(1)}$ and $\alpha^{(2)}$. The paths are smoothed through a running average.

dimensional signal of length 28 time steps), has been analyzed in [40] and in [41]. In [40] the original dataset was preprocessed through reshaping and rotating the original image to enhance the network's ability to understand high level features of the data. In this case, the original dataset is used. In [41], the addition of thresholds and the introduction of sparse representation in the read-out of the reservoir was used to improve the performance of the network in the online learning of the standard MNIST task through reservoir computing. This section is focused on the analysis of the performance of ESNs on the psMNIST task and on their dependence on the range of timescales available in the network, i.e. the values of α and ρ chosen. In contrast to the previous sections where ESNs are trained through ridge regression, we have applied an online gradient descent optimisation method. The cost function chosen to be minimised is the cross entropy loss

$$E = -\frac{1}{N_{batch}} \sum_{m=1}^{N_{batch}} \sum_{j=1}^{N_{class}} \left[y_j(m) \log\left(\tilde{y}_j(m)\right) + \left(1 - y_j(m)\right) \log\left(1 - \tilde{y}_j(m)\right) \right], \tag{35}$$

where m is the minibatch index, N_{batch} corresponds to the minibatch size and N_{class} is the number of classes. For this task the desired output, y_j, is a one-hot encoded vector of the correct classification while the desired output is a sigmoid function of the readout of the reservoir nodes. Furthermore, instead of reading out the activity of the reservoir from the final temporal step of each sequence only, we have expanded the reservoir representation by using previous temporal activities of the network. In practice, given the sequence of activities $\mathbf{x}(0), \mathbf{x}(\delta t), \ldots, \mathbf{x}(\delta t T)$ $(T = 784)$ that defines the whole temporal dynamic of the network subjected to an example input sequence, we trained the network by reading

out from the expanded vector $\mathbf{X} = [\mathbf{x}(M\delta t), \mathbf{x}(2M\delta t), \ldots, \mathbf{x}(T\delta t)]$, where M defines the "time frame" used to sample the activities of the evolution of the system across time.

$$\tilde{\boldsymbol{y}} = \mathrm{sigm}\left(\sum_{n=1}^{T/M} \mathbf{W}_{\mathrm{out}}^{(n)} \mathbf{x}(nM\delta t) \right) \tag{36}$$

where sigm stands for sigmoid activation function. We then repeat the simulation for two different time frames of sampling for each different model, that is a single ESN and a pair of unconnected or connected ESNs, as in the previous sections.

The two values of M used are 28 and 196, corresponding to a sampling of 28 and 4 previous representations of the network respectively. Of course, a higher value of M corresponds to a more challenging task, since the network has to exploit more its dynamic to infer temporal dependencies. We note, however, that none of the representation expansions used can guarantee a good understanding of the temporal dependencies of the task, or in other words, can guarantee that the system would be able to discover higher order features of the image, considering that these features depend on events that could be distant in time.

In **Figure 6** we again analyze the performance of two connected or unconnected ESNs varying $\alpha^{(1)}$ and $\alpha^{(2)}$ for both $M = 28$ and 196. In contrast to the previous sections, we now use gradient descent learning on the output weights instead of ridge regression and increase the total number of neurons in each model to $N = 1200$ due to the complexity of the task. The Adam optimiser is used; its parameters, for both the output weights and α learning, are in **Table 1**. As previously, we have trained the output weights over a range of fixed αs and report the performance on an unseen test data set. In parallel to this we have trained both the output weights and α values which, as shown by the lines on the contour plots, converge toward the minimum computed using the fixed α′s.

As in the other simulations, we found that the values of ρ corresponding to the best performance was approximately one, which maximises the range of timescales and the memory available in the network. **Figures 6E,F** shows the case with $M = 28$, while **Figures 6G,H** reports the accuracy for the simulation with $M = 196$ where E and G are unconnected and F and H connected reservoirs. The accuracy surface demonstrates how, in the case of the unconnected ESNs with a fast sampling rate in panel G, the best performance is achieved when at least one of the two values of α is close to one. The result is due to the fast changing dynamic of the temporal sequence that is introduced through the random permutation of the pixels. On the contrary, in the case of the unconnected ESNs with a slow sampling rate in panel E the best accuracy is in a range of intermediate timescales since both partitions must respond to both fast and slow timescales.

This relatively simple behavior of the dependence of the accuracy on the setting of the hyper-parameters changes in the cases of two connected ESNs, whose additional complexity corresponds to a considerable increase in the performance. **Figure 6H** reports how the network prefers a regime with a fast timescale in the first reservoir and a intermediate timescale in the second, which acts as an additional non-linear temporal filter of the input provided by the first network. The need of memory of events distant in time is emphasised in 6F, where the best performing network is composed by reservoirs with fast and slow dynamics respectively. The performance boost from the panels E–G to the ones F-H has only two possible explanations: first, the timescales of the second network are increased naturally thanks to the input from the first reservoir; second, the connections between the two reservoirs provide an additional non-linear filter of the input that can be exploited to discover higher level features of the signal. Thus, we can conclude once again that achieving high performance in applying reservoir models requires 1) additional non-linearity introduced through the interconnections among the reservoirs and 2) an appropriate choice of timescales, reflecting the task requirements in terms of external signal and memory.

Panels I, L, M and N show the application of the online training of αs for the various cases analyzed. In the psMNIST task we found that the major difficulties in the application of an iterative learning rule on the leakage terms are: the possibility to get trapped in local minima, whose abundance can be caused by the intrinsic complexity of the task, the intrinsic noise of the dataset, the randomness of the reservoir and of the applied permutation; the high computational time of a simulation that exploits an iterative optimisation process on *αs* arising from a practical constraint in the implementation. Indeed, while the activities of the reservoir can be computed once across the whole dataset and then saved in the case of untrained values of αs, the activities of the nodes need to be computed every time the leakage terms change in the online learning paradigm. However, we found that using a higher learning rate η_W on the output weights, compared to the value adopted in the paradigm where the leakage terms are not optimised (as in Panels E, F, G and H), can introduce beneficial noise in the learning process and help to avoid local minima. Furthermore, a higher value of the learning rate on the output weights corresponds to an increased learning rate on the thresholds, as shown from **Supplementary Equation S7** and from the dependence of the updating equations on $\mathbf{W}_{\mathrm{out}}$. As in the previous simulations of **Sections 3.1 and 3.2**, the output weights are randomly reinitialised after the convergence of αs, helping the algorithm to avoid an undesirable quick convergence of weights. The online process is then ended when the leakage terms remain approximately constant even after the re-initialisation. Following this computational recipe, it possible to avoid the difficulties found and train the leakage terms efficiently.

Finally, we note how the best accuracy of 0.96 reached throughout all the experiments on the psMNIST is comparable to the results obtained by recurrent neural networks trained with BPTT, whose performance on this task are analyzed in [42] and can vary from 0.88 to 0.95. In comparison to recurrent structures trained through BPTT, a network with two interacting ESNs provide a cheap and easily trainable model. However, this comparison is limited by the necessity of recurrent neural networks to carry the information from the beginning to the end of the sequence, and to use the last temporal state only or to adopt attention mechanisms.

4 CONCLUSION

In summary, ESNs are a powerful tool for processing temporal data, since they contain internal memory and time-scales that can be adjusted via network hyper-parameters. Here we have highlighted that multiple internal time-scales can be accessed by adopting a split network architecture with differing hyper-parameters. We have explored the performance of this architecture on three different tasks: NARMA10, a benchmark composed by a fast-slow telegraph process and PSMNIST. In each task, since multiple timescales are present the hierarchical ESN performs better than a single ESN when the two reservoirs have separate slow and fast timescales. We have demonstrated how choosing the optimal leakage terms of a reservoir can be aided by the theoretical analysis in the linear regime of the network, and by studying the auto-correlation structure of the input and/or desired output and the memory required to solve the task. The theoretical analysis developed needs to be considered as a guide for the tuning of the reservoir hyper-parameters, and in some specific applications it could be insufficient because of the lack of information about the nature of the task. In this regard, we showed how to apply a data-driven online learning method to optimise the timescales of reservoirs with different structures, demonstrating its ability to find the operating regimes of the network that correspond to high performance and to the best, task-dependent, choice of timescales. The necessity of adopting different leakage factors is emphasised in the case of interactive reservoirs, whose additional complexity leads to better performance in all cases analyzed. Indeed, the second reservoir, which acts as an additional non linear filter with respect to the input, is the perfect candidate to discover higher temporal features of the signal, and it consequently prefers to adopt longer timescales in comparison to the first reservoir, which has instead the role of efficiently representing the input. We believe such hierarchical architectures will be useful for addressing complex temporal problems and there is also potential to further optimise the connectivity between the component reservoirs by appropriate adaptation of the online learning framework presented here.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. This data can be found here: http://yann.lecun.com/exdb/mnist/. The code for this study is available from: https://github.com/LucaManneschi/EchoStatesNetwork.

AUTHOR CONTRIBUTIONS

All the authors contributed to the paper conceptually and to the writing of the article. EV proposed the original concept of the paper. LM, ME, GG, PD, EV contributed to the design of the case studies and to the analysis of the results. LM and ME designed and performed the simulations.

FUNDING

EV and AL acknowledge the support from a Google Deepmind Award. EV and ME were funded by the Engineering and Physical Sciences Research Council (Grant No. EP/S009647/1). EV was also supported by EPSRC (Grant Nos. EP/S030964/1 and EP/P006094/1). PD and GG were supported by the European Union Horizon 2020 Research and Innovation program under the FET Flagship Human Brain Project (SGA2 Grant agreement No. 785907 and SGA3 Grant agreement No. 945539). AL was supported by the European Research Council (639489) and the Biotechnology and Biological Sciences Research Council (BB/S016031/1).

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fams.2020.616658/full#supplementary-material.

REFERENCES

1. Ludik J, Prins W, Meert K, Catfolis T. A comparative study of fully and partially recurrent networks. *Proc Int Conf Neural Netw* (1997) 1(ICNN'97):292–7. doi:10.1109/ICNN.1997.611681
2. Rumelhart DE, Hinton GE, Williams RJ. Technical Report. Learning internal representations by error propagation. California Univ San Diego La Jolla Inst for Cognitive Science (1985)
3. Werbos PJ. Backpropagation through time: what it does and how to do it. *Proc IEEE* (1990) 78:1550–60. doi:10.1109/5.58337
4. Bellec G, Scherr F, Subramoney A, Hajek E, Salaj D, Legenstein R, et al. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nat Commun* (2020) 11:3625. doi:10.1038/s41467-020-17236-y
5. Marr D. A theory of cerebellar cortex. *J Physiol* (1969) 202:437–70. doi:10.1113/jphysiol.1969.sp008820
6. Farris SM. Are mushroom bodies cerebellum-like structures? *Arthropod Struct Dev* (2011) 40:368–79. doi:10.1016/j.asd.2011.02.004
7. Laurent G. Olfactory network dynamics and the coding of multidimensional signals. *Nat Rev Neurosci* (2002) 3:884–95. doi:10.1038/nrn964
8. Warren R, Sawtell NB. A comparative approach to cerebellar function: insights from electrosensory systems. *Curr Opin Neurobiol* (2016) 41:31–7. doi:10.1016/j.conb.2016.07.012
9. Takemura SY, Aso Y, Hige T, Wong A, Lu Z, Xu CS, et al. A connectome of a learning and memory center in the adult Drosophila brain. *eLife* (2017) 6:5643. doi:10.7554/eLife.26975
10. Zheng Z, Lauritzen JS, Perlman E, Robinson CG, Nichols M, Milkie D, et al. A complete electron microscopy volume of the brain of adult *Drosophila melanogaster*. *Cell* (2018) 174:730–43. doi:10.1016/j.cell.2018.06.019
11. Liu Q, Yang X, Tian J, Gao Z, Wang M, Li Y, et al. Gap junction networks in mushroom bodies participate in visual learning and memory in Drosophila. *eLife* (2016) 5:e13238. doi:10.7554/eLife.13238
12. Shang Y, Claridge-Chang A, Sjulson L, Pypaert M, Miesenböck G. Excitatory local circuits and their implications for olfactory processing in the fly antennal lobe. *Cell* (2007) 128:601–12. doi:10.1016/j.cell.2006.12.034
13. Olsen SR, Wilson RI. Lateral presynaptic inhibition mediates gain control in an olfactory circuit. *Nature* (2008) 452:956–60. doi:10.1038/nature06864
14. Yeshurun Y, Nguyen M, Hasson U. Amplification of local changes along the timescale processing hierarchy. *Proc Natl Acad Sci U S A* (2017) 114:9475–80. doi:10.1073/pnas.1701652114

15. Jaeger H. *The "echo state" approach to analysing and training recurrent neural networks-with an erratum note*. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report (2001) 148:13.
16. Deng Z, Zhang Y. Collective behavior of a small-world recurrent neural system with scale-free distribution. *IEEE Trans Neural Netw* (2007) 18:1364–75. doi:10.1109/tnn.2007.894082
17. Rodan A, Tino P. Minimum complexity echo state network. *IEEE Trans Neural Netw* (2010) 22:131–44. doi:10.1109/TNN.2010.2089641
18. Bacciu D, Bongiorno A. Concentric esn: assessing the effect of modularity in cycle reservoirs. In: 2018 International Joint Conference on Neural Networks (IJCNN); 2018 July 8–13; Rio, Brazil . (IEEE) (2018), 1–8.
19. Farkaš I, Bosák R, Gergeľ P. Computational analysis of memory capacity in echo state networks. *Neural Netw* (2016) 83:109–20. doi:10.1016/j.neunet.2016.07.012
20. Marzen S. Difference between memory and prediction in linear recurrent networks. *Phys Rev E* (2017) 96:032308. doi:10.1103/PhysRevE.96.032308
21. Livi L, Bianchi FM, Alippi C. Determination of the edge of criticality in echo state networks through Fisher information maximization. *IEEE Trans Neural Netw Learn Syst* (2018) 29:706–17. doi:10.1109/TNNLS.2016.2644268
22. Xue Y, Yang L, Haykin S. Decoupled echo state networks with lateral inhibition. *Neural Netw* (2007) 20:365–76. doi:10.1016/j.neunet.2007.04.014
23. Jaeger H. Technical Report. Discovering multiscale dynamical features with hierarchical echo state networks. Bremen, Germany: Jacobs University Bremen (2007)
24. Gallicchio C, Micheli A, Pedrelli L. Deep echo state networks for diagnosis of Parkinson's disease (2018a) arXiv preprint. Available from: https://arxiv.org/abs/1802.06708 (Accessed February 19, 2018).
25. Malik ZK, Hussain A, Wu QJ. Multilayered echo state machine: a novel architecture and algorithm. *IEEE Trans Cybernetics* (2016) 47:946–59. doi:10.1109/TCYB.2016.2533545
26. Gallicchio C, Micheli A. Echo state property of deep reservoir computing networks. *Cogn Comp* (2017) 9:337–50. 10.1007/s12559-017-9461-9
27. Gallicchio C, Micheli A, Pedrelli L. Design of deep echo state networks. *Neural Netw* (2018b) 108:33–47. doi:10.1016/j.neunet.2018.08.002
28. Ma Q, Shen L, Cottrell GW. Deepr-esn: a deep projection-encoding echo-state network. *Inf Sci* (2020) 511:152–71. doi:10.1016/j.ins.2019.09.049
29. Rodriguez N, Izquierdo E, Ahn YY. Optimal modularity and memory capacity of neural reservoirs. *Netw Neurosci* (2019) 3:551–66. doi:10.1162/netn_a_00082
30. Szita I, Gyenes V, Lőrincz A. Reinforcement learning with echo state networks. In: International Conference on Artificial Neural Networks; December 5–8, 2006; Athens, Greece (Springer) (2006) 830–9.
31. Hermans M, Schrauwen B. Recurrent kernel machines: computing with infinite echo state networks. *Neural Comput* (2012) 24:104–33. doi:10.1162/NECO_a_00200
32. Jaeger H, Lukoševičius M, Popovici D, Siewert U. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Netw* (2007) 20:335–52. doi:10.1016/j.neunet.2007.04.016
33. Girko VL. Circular law. *Theory Probab Appl* (1985) 29:694–706. doi:10.1038/s41586-019-1763-5
34. Gallicchio C, Micheli A, Pedrelli L. Deep reservoir computing: a critical experimental analysis. *Neurocomputing* (2017) 268:87–99. doi:10.1016/j.neucom.2016.12.089
35. Sun X, Li T, Li Q, Huang Y, Li Y. Deep belief echo-state network and its application to time series prediction. *Knowl Based Syst* (2017) 130:17–29. doi:10.1016/j.knosys.2017.05.022
36. Manneschi L, Vasilaki E. An alternative to backpropagation through time. *Nat Mach Intell* (2020) 2:155–6. doi:10.1002/mp.14033
37. Atiya AF, Parlos AG. New results on recurrent network training: unifying the algorithms and accelerating convergence. *IEEE Trans Neural Networks* (2000) 11:697–709. doi:10.1109/72.846741
38. Goudarzi A, Banda P, Lakin MR, Teuscher C, Stefanovic D. A comparative study of reservoir computing for temporal signal processing (2014) arXiv preprint. Available from: https://arxiv.org/abs/1401.2224 (Accessed January 10, 2014).
39. Lukoševičius M, Jaeger H. Reservoir computing approaches to recurrent neural network training. *Computer Science Review* (2009) 3:127–49. doi:10.1016/j.cosrev.2009.03.005
40. Schaetti N, Salomon M, Couturier R. Echo state networks-based reservoir computing for mnist handwritten digits recognition. In: 2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES); 2016 August 24–26; Paris, France (IEEE) (2016) 484–91.
41. Manneschi L, Lin AC, Vasilaki E. Sparce: sparse reservoir computing (2019) arXiv preprint. Available at: https://arxiv.org/abs/1912.08124 (Accessed December 4, 2019).
42. Chandar S, Sankar C, Vorontsov E, Kahou SE, Bengio Y. Towards non-saturating recurrent units for modelling long-term dependencies. *Proc AAAI Conf Artif Intell* (2019) 33:3280–7. doi:10.1609/aaai.v33i01.33013280

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.