Chaos

Multifunctionality in a reservoir computer

Cite as: Chaos **31**, 013125 (2021); https://doi.org/10.1063/5.0019974
Submitted: 26 June 2020 • Accepted: 18 December 2020 • Published Online: 12 January 2021

Andrew Flynn, Vassilios A. Tsachouridis and Andreas Amann

Multifunctionality in a reservoir computer

Cite as: Chaos **31**, 013125 (2021); doi: 10.1063/5.0019974
Submitted: 26 June 2020 · Accepted: 18 December 2020 ·
Published Online: 12 January 2021

Andrew Flynn,[1,a)] Vassilios A. Tsachouridis,[2] and Andreas Amann[1]

AFFILIATIONS

[1]School of Mathematical Sciences, University College Cork, Cork T12 XF62, Ireland
[2]Raytheon Technologies Research Center Ireland, Cork T23 XN53, Ireland

[a)]**Author to whom correspondence should be addressed:** andrew_flynn@umail.ucc.ie

ABSTRACT

Multifunctionality is a well observed phenomenological feature of biological neural networks and considered to be of fundamental importance to the survival of certain species over time. These multifunctional neural networks are capable of performing more than one task without changing any network connections. In this paper, we investigate how this neurological idiosyncrasy can be achieved in an artificial setting with a modern machine learning paradigm known as "reservoir computing." A training technique is designed to enable a reservoir computer to perform tasks of a multifunctional nature. We explore the critical effects that changes in certain parameters can have on the reservoir computers' ability to express multifunctionality. We also expose the existence of several "untrained attractors"; attractors that dwell within the prediction state space of the reservoir computer were not part of the training. We conduct a bifurcation analysis of these untrained attractors and discuss the implications of our results.

Advancements in machine learning often arise from a "two-way street" between neuroscientific observation and mathematical representation. In this paper, we stroll through such a street with inspiration from "multifunctional neural networks." These are networks of neurons whose activity patterns can change on the demand of performing a given duty, but synapses remain fixed. We conceptualize multifunctionality in the context of dynamical systems and machine learning by using a reservoir computer as a means to realize this neurological feat in an artificial setting. More specifically, we train a Reservoir Computer to imitate the dynamics of numerous chaotic attractors from different sources based on a given initial condition. To do this, we design a training technique that "blends" and weights data from these chaotic attractors. We explore how different weightings and changes in the memory of this artificial neural network effect the desired learning outcomes. In doing so, we uncover some "behind-the-scenes" bifurcations of several other attractors found to be lurking within the prediction state space that interfere with the network capacity to express multifunctionality. Above all, this paper identifies some new application areas suitable to a reservoir computer and broadens the current understanding of the dynamical capabilities inherent to this learning system.

I. INTRODUCTION

Multifunctionality is an essential element of biological neural networks.[1–3] These multifunctional networks are distinct pools of neurons capable of performing a multitude of mutually exclusive tasks. To elaborate with an example, it was found that a subset of the same bundle of neurons in the brain of the medicinal leech (*Hirudo medicinalis*) can switch their activity pattern once it senses a change in its surroundings to drive either a swimming or crawling motion.[4] It is reported that a cluster of neurons located in the *pre-Bötzinger complex* (a region of the mammalian brain stem) is responsible for regulating a switching between different respiratory patterns.[5] Depending on a particular input, the neurons in this region of the brain can alter their activity pattern accordingly to elicit a switching between eupneic (regular) breathing, sighing, or gasping. Furthermore, it is argued that multifunctionality in neural networks may naturally emerge from an efficient use of limited resources (in this case, neurons) and thus an evolutionary advantage in enduring environmental changes, reflecting the developmental history of certain organisms.[6]

Nevertheless, what is ubiquitous among these multifunctional neural networks is that they in principle resemble a system with more than one modus operandi. Based on a particular input, there is a distinct activity pattern expressed by the neurons in the network in order to perform one of the many tasks required of it. When needed, these multifunctional neurons switch to another activity pattern to collectively execute a different task while the network connections remain fixed. Therefore, if an artificial neural network was trained to sustain more than one desired activity pattern, it would in this sense be multifunctional. From a dynamical system perspective, this

type of behavior is akin to a multistable system or a system with a coexistence of attractors. For further reading on multistable systems, see Pisarchik and Feudel.[7]

There is much to be gained in the pursuit of artificial intelligence by articulating our current knowledge of biological neural networks and dynamical systems in machine learning environments. In this paper, we employ such a bilateral rationale to encapsulate multifunctionality in an artificial neural network by training a "Reservoir Computer"[8–10] (RC) to facilitate the coexistence of more than one desired activity pattern.

The RC approach to machine learning has been successfully applied to a number of problems, for example, time series prediction,[11] visual identification tasks,[12] real-time detection of epileptic seizures,[13] inferring from limited time series data, unmeasured state variables,[14] Lyapunov exponents,[15] and causal dependencies between variables.[16]

The basis of our research involves using a recent formulation of a continuous-time RC, presented by Lu *et al.*[17] Here, the RC was trained to perform short term predictions of a chaotic Lorenz system[18] and reconstruct the "climate" (qualitatively similar dynamical behavior) of its famous butterfly shaped chaotic attractor. Taking this result into account, we train a RC to promote a coexistence of reconstructed chaotic attractors in its prediction state space, thus becoming multifunctional. In order to demonstrate the flexibility of our approach, we consider training scenarios in which the climate of these chaotic attractors is reconstructed from a variety of sources. For example, we consider the case where these chaotic attractors are generated from two different systems entirely. We devise a training technique to "blend" data with a certain weighting parameter from these chaotic attractors. The choice of this weighting along with a parameter involved in tuning the memory of the RC is critical to achieving multifunctionality. We investigate the optimal setting of these parameters, from which we infer the regions in this parameter space where the RC achieves multifunctionality.

However, while we train the RC to realize more than one chaotic attractor in its prediction state space, we find several "untrained attractors" also residing here. These attractors inhabit the prediction state space but were not part of the training and limit the regions in which multifunctionality is obtained. A bifurcation analysis of these untrained attractors reveals some interesting dynamics where, for example, one of these attractors undergoes a period-doubling route to chaos.

The structure of the rest of the paper is as follows. In Sec. II, we provide details of the RC approach to attractor reconstruction and present the training procedure we use to achieve multifunctionality in a RC. Next, in Sec. III, the problem of epitomizing multifunctionality in a RC is further conceptualized. The trajectories on the chaotic attractors we use as our training data are also given here. In Sec. IV, we present our main findings and then provide an extended discussion of our results in Sec. V.

II. RESERVOIR COMPUTING

Echo-State Networks[8] (ESNs) and Liquid-State Machines[9] (LSMs) are two independently proposed designs of artificial neural networks with recurrent connections that can be trained to provide a self-sustained activity pattern. While ESNs were engineered in the context of machine learning and LSMs were developed from a computational neuroscience perspective, they share a similar philosophy and as a result have become increasingly synonymous under the umbrella term of "Reservoir Computing" or indeed a "Reservoir Computer" (RC).[10] Both are based upon the notion that as long as the internal connections, or in this case the "reservoir," possess certain characteristics, it is not necessary to adapt the internal weights of the network in order to achieve a desired learning outcome. Instead, it is sufficient to find an appropriate readout layer for a given task. Consequently, as the internal connections of the reservoir remain unaltered throughout the training, it can be represented physically. There are many interesting constructions of these "Physical Reservoir Computers," using, for example, an optoelectronic system[19] or an octopus inspired soft robotic arm.[20] For a recent review, see Tanaka *et al.*[21]

For the purpose of keeping this paper self-contained, we now outline the anatomy and implementation stages of the RC setup we use that was proposed by Lu *et al.*[17] We also present the technique we devised in order to train the RC to reconstruct the climate of more than one chaotic attractor based on a given initial condition.

A. Listening stage

In the listening stage, as illustrated in Fig. 1, the input data are used to drive the "listening reservoir" away from its initial state. This drive-response system evolves according to

$$\dot{\boldsymbol{r}}(t) = \gamma\left[-\boldsymbol{r}(t) + \tanh\left(\mathbf{M}\,\boldsymbol{r}(t) + \sigma\,\mathbf{W}_{in}\,\boldsymbol{u}(t)\right)\right]. \tag{1}$$

Here, $\boldsymbol{r}(t) \in \mathbb{R}^N$ is the state of the reservoir at a given time t and N denotes the number of artificial neurons. γ is a parameter arising from the transformation of the discrete-time formulation in Jaeger and Haas[11] to continuous-time. $\mathbf{M} \in \mathbb{R}^{N\times N}$ is the adjacency matrix representing the internal network connections of the reservoir. The input strength parameter, σ, and $\mathbf{W}_{in} \in \mathbb{R}^{N\times D}$, the input matrix, when multiplied together represent the weight given to the input vector $\boldsymbol{u}(t) \in \mathbb{R}^D$ as it is projected into the reservoir. Here, D is the dimension of the input data. We compute trajectories of Eq. (1) using the 4th order Runge–Kutta method with time step $\tau = 0.01$.

In the listening stage, we allow Eq. (1) to evolve from $t = 0$ to $t = t_{listen} = 200$ in order to remove any dependency the RC may have on its initial condition and synchronize to the input.

B. Training stage

In the training stage, we focus on the data generated from Eq. (1) and $\boldsymbol{u}(t)$ for $t_{listen} \le t \le t_{train} = 400$. The aim of training is to determine a post-processing function of the reservoir state, $\hat{\psi}\,(\boldsymbol{r}(t))$, which can replace the input. Like in Lu *et al.*,[17] we consider post-processing functions of the following form:

$$\hat{\psi}\,(\boldsymbol{r}(t)) = \mathbf{W}_{out}\,\boldsymbol{q}(\boldsymbol{r}(t)). \tag{2}$$

Here, $\boldsymbol{q}(\boldsymbol{r}(t)) \in \mathbb{R}^{2N}$ is a vector function that returns a vector where the first N elements are $\boldsymbol{r}(t)$ and the second N elements are $\boldsymbol{r}^2(t)$. While this step differs from many other manifestations of the readout layer, it is said to be advantageous over linear functions of $\boldsymbol{r}(t)$ and embeds further nonlinearity in the network. The "output

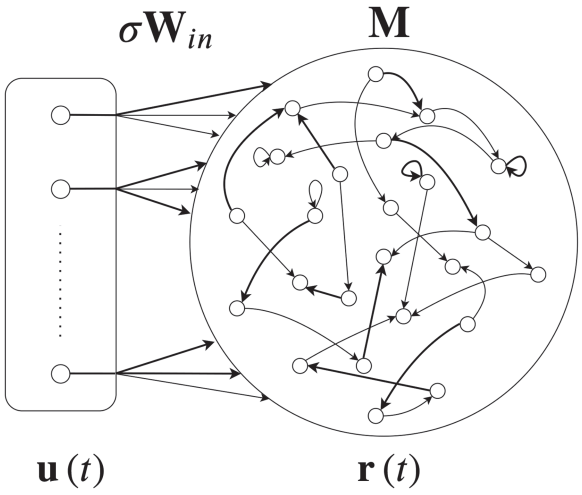


FIG. 1. Listening reservoir: The input signal $\boldsymbol{u}(t)$ is projected by $\sigma\mathbf{W}_{in}$ to drive a response from the reservoir state, $\boldsymbol{r}(t)$.

matrix," $\mathbf{W}_{out} \in \mathbb{R}^{D\times 2N}$, is determined by a ridge regression procedure which we now outline.

Each evaluation of $\boldsymbol{q}(\boldsymbol{r}(t))$ during the training time is stored in columns of the regularization matrix, $\mathbf{X}$,

$$\mathbf{X} = \left[\begin{array}{cccc} \boldsymbol{q}(\boldsymbol{r}(t_{listen})) & \boldsymbol{q}(\boldsymbol{r}(t_{listen}+\tau)) & \cdots & \boldsymbol{q}(\boldsymbol{r}(t_{train})) \end{array}\right]. \quad (3)$$

The target data matrix, $\mathbf{Y}$, is constructed in a similar manner,

$$\mathbf{Y} = \left[\begin{array}{cccc} \boldsymbol{u}(t_{listen}) & \boldsymbol{u}(t_{listen}+\tau) & \cdots & \boldsymbol{u}(t_{train}) \end{array}\right]. \quad (4)$$

Finally, $\mathbf{W}_{out}$ is calculated as

$$\mathbf{W}_{out} = \mathbf{Y}\mathbf{X}^T\left(\mathbf{X}\mathbf{X}^T + \beta\,\mathbf{I}\right)^{-1}; \quad (5)$$

here, β is the regularization parameter, and its role is to discourage overfitting and penalize large elements of $\mathbf{W}_{out}$ from occurring. $\mathbf{I}$ is the identity matrix of the appropriate dimension.

C. Predicting stage

In the predicting stage, we compute solutions of the "predicting reservoir" described by the following equation:

$$\dot{\hat{\boldsymbol{r}}}(t) = \gamma\left[-\hat{\boldsymbol{r}}(t) + \tanh\left(\mathbf{M}\,\hat{\boldsymbol{r}}(t) + \sigma\mathbf{W}_{in}\mathbf{W}_{out}\,\boldsymbol{q}(\hat{\boldsymbol{r}}(t))\right)\right], \quad (6)$$

with $\hat{\boldsymbol{r}}(0) = \boldsymbol{r}(t_{train})$. This setup is illustrated in Fig. 2.

If the training was successful, the readout from the reservoir, $\mathbf{W}_{out}\,\boldsymbol{q}(\hat{\boldsymbol{r}}(t))$, should be an approximation of the original input which we denote as $\hat{\boldsymbol{u}}(t) \approx \boldsymbol{u}(t)$.

In our numerical experiments, we set $N = 1000$ and the reservoir state is initialized at the beginning of the listening stage as $\boldsymbol{r}(0) = (0, 0, \ldots, 0)^T = \mathbf{0}^T$ for all simulations. Here, T denotes the transpose operation. $\mathbf{M}$ is constructed as a random matrix of sparse Erdös–Renyi connectivity with a specific spectral radius, ρ. To elaborate, the matrix is designed such that each element is chosen independently to be nonzero with probability $P = 0.04$ (i.e., sparsity $= 0.04$ or degree $= 40$), and these nonzero elements are chosen

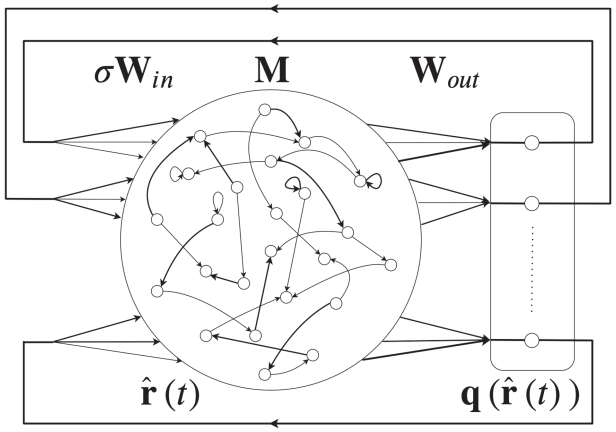


FIG. 2. Predicting reservoir: The readout layer $\mathbf{W}_{out}\,\boldsymbol{q}(\hat{\boldsymbol{r}}(t))$ replaces the external input to the reservoir.

uniformly from $(-1, 1)$. This random sparse matrix is rescaled such that the magnitude of its largest eigenvalue is ρ. For example, if ρ is close to 1, this in effect means that the input takes a long time to die out within the reservoir, which is more preferable in tasks requiring a large memory.[22] The $\mathbf{W}_{in}$ matrix is designed such that each row has only one nonzero randomly assigned element, chosen uniformly from $(-1, 1)$. The time damping factor is kept constant at $\gamma = 10$, while ρ, σ, and β are varied in order to achieve a desired learning outcome.

We remark that it is difficult to choose ρ, σ, γ, and β for a specific task. To combat this, gradient[23] and Bayesian[24] based parameter optimization algorithms have been developed to reduce the error in time series prediction problems. Such methods go beyond the scope of the current paper but may be useful in further studies. Instead, it is our focus to understand the mechanisms that give rise to multifunctionality in a RC. We do this by exploring the dynamics exhibited by the RC in a range of parameter values. As we will see, a multifunctional RC requires that for a given set of parameters, it can reconstruct the climate of more than one attractor. However, certain attractors may require different parameter settings to be reconstructed. In this paper, we investigate the effects in performance that changing ρ has in terms of the RC reconstructing a pair of attractors and, therefore, achieving multifunctionality. We choose to work with a random Erdös–Renyi topology in order to provide the RC with enough dynamical flexibility to solicit multistable dynamics.

Next, we present the training technique we designed that combines data from different sources in order to construct a single output matrix that allows for the reconstruction of more than one chaotic attractor.

D. Training with the "blending technique"

We adapt the regression procedure from Sec. II B to instead use data from two input sources and the corresponding reservoir output in both training stages. From a philosophical perspective, it is necessary that the matrices $\mathbf{M}$ and $\mathbf{W}_{in}$ and parameters ρ, σ, and β remain identical when generating both training data sets. These

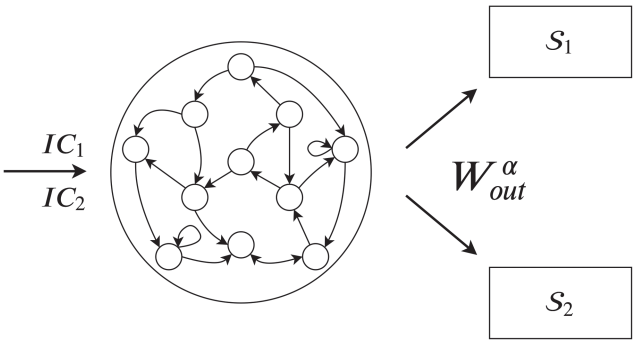


FIG. 3. Illustration of a multifunctional RC.

data are stitched together in what we call the "blending technique" with the "blending" or "weighting" parameter, $\alpha \in [0, 1]$. The particular construction of the regularization matrix, $\mathbf{X}$, and the target data matrix, $\mathbf{Y}$, used in the regression are now outlined.

First, the training data collected from the reservoir regarding each individual attractor, $\mathbf{X}_{\mathcal{S}_1}$ and $\mathbf{X}_{\mathcal{S}_2}$, for $\mathcal{S}_1$ and $\mathcal{S}_2$ some arbitrary attractors, are grouped together and "blended" in the following concatenation:

$$\mathbf{X}_C = \left(\alpha \mathbf{X}_{\mathcal{S}_1},\, (1-\alpha)\, \mathbf{X}_{\mathcal{S}_2}\right). \tag{7}$$

Based on this construction, when $\alpha = 0$ or $\alpha = 1$, one data set completely dominates the training. The same procedure is applied to the corresponding target data matrices in order to obtain the equivalent $\mathbf{Y}_C$.

To avoid any biases in the concatenation step, we take advantage of the memory-less based readout layer by randomly reordering each column of the matrices, $\mathbf{X}_C$ and $\mathbf{Y}_C$, corresponding to the input and reservoir state at a given time. These matrices are used in the ridge regression formula in Eq. (5).

The aim is to find an α that will give rise to an output matrix, $\mathbf{W}^{\alpha}_{out}$, which, depending on the initial condition (IC), allows the RC to reconstruct one or the other attractor. We provide a schematic of the desired outcome in Fig. 3.

We remark that recently, a RC was used in a chaotic source separation problem where it was trained using a different method of blending signals from various chaotic sources.[25] In contrast to our method, Krishnagopal *et al.*[25] consider training a RC to separate a blended input formed by a linear combination of two differently weighted chaotic signals into its constituents. Once the RC is trained on this sum of mixed signals, the aim is to suppress one of the signals and continue to predict the evolution of the other chaotic time series. This work differs from the results of the current paper as we aim to train a single RC on a weighted and randomly blended training data set from two different chaotic attractors and predict the evolution of either chaotic attractor based on a given IC.

Related to the concept of multifunctionality in artificial neural networks is the notion of "systems within systems." In this approach, distinct sub-networks are assigned particular duties within a larger network to collectively perform different behaviors. For example, the modular selection and identification control technique proposed by Wolpert and Kawato[26] focuses on a "modular neural network" setup whereby a given trained network is singled out among others and brought into operation by a responsibility signal in order to generate the specific behavior required from it. Furthermore, it was demonstrated by Tani *et al.*[27] that multiple temporal patterns can be learned using the recurrent neural network with parametric biases setup. Here, the parametric biases act as bifurcation parameters that change the dynamical regime of the network in order to generate a specific behavior. In contrast, our approach is not modular or requires a conscious change in parameters in order to reconstruct different attractors using the same network. When the training is successful, the multifunctional RC setup presented in this paper operates on a global scale where the artificial neurons are specifically organized in the network to perform more than one task based on a given IC.

III. MULTIFUNCTIONALITY AND DYNAMICAL SYSTEMS

The characterization of neuronal behavior regularly invokes the language of dynamical systems theory.[6,28] Conceptualizing certain traits of neurons from this perspective can act as a bridge between biological and artificial neural networks where advancements in one can contribute to the other.

Considering the claim made in Sec. I that a multifunctional neural network in principle resembles a system with a coexistence of attractors and that a RC can be trained to reconstruct the climate of a chaotic attractor, we pose the following: can a RC be trained to permit a coexistence of reconstructed attractors? Such a RC can be said to, in the spirit of our claim, express multifunctionality.

To demonstrate this, we consider the following tasks that require multifunctionality by training a RC to reconstruct a coexistence of chaotic attractors from

- Case I: a multistable system.
- Case II: a system with different parameter settings.
- Case III: two different systems entirely.

We remark that such tasks closely resemble the chaotic and highly variable behavior expressed by certain multifunctional neural networks as observed in nature.[29,30] Furthermore, coexisting chaotic attractors are also found to occur in some low-dimensional models of neuronal systems.[31,32]

In order to provide an adequate testing ground, we require input data from a system that allows for the generation of multiple coexisting chaotic attractors under a wide range of parameters. An example of such a system was presented in Guan *et al.*[33] and described by the following set of equations:

$$\begin{aligned} \dot{x}_1(t) &= a\, x_1(t) - x_2(t)\, x_3(t) - x_2(t) + d, \\ \dot{x}_2(t) &= -b\, x_2(t) + x_1(t)\, x_3(t), \\ \dot{x}_3(t) &= -c\, x_3(t) + x_1(t)\, x_2(t). \end{aligned} \tag{8}$$

Here, $\boldsymbol{x}(t) = (x_1(t), x_2(t), x_3(t))^T$ defines the state of the system at a given time t. a, b, c, and d are the system parameters.

For example, when setting $(a, b, c, d) = (5, 15, 3, 12)$, there is a coexistence of two single-scroll chaotic attractors as illustrated in Fig. 4. Initializing Eq. (8) with $\boldsymbol{x}^{\mathcal{A}_1}(0) = (1, 1, 1)^T$ results in the

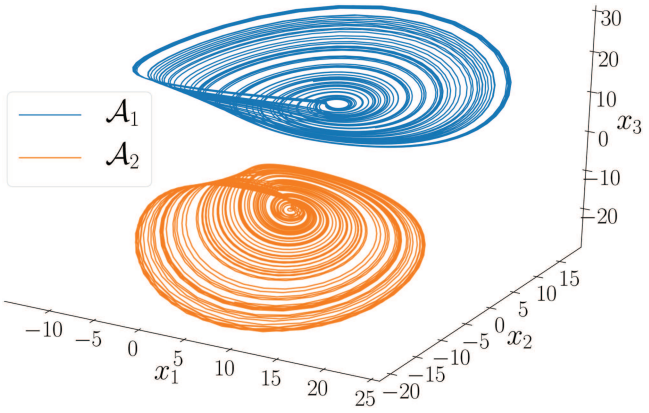


FIG. 4. Coexisting attractors in the state space of Eq. (8) for $(a, b, c, d) = (5, 15, 3, 12)$ and $\boldsymbol{x}^{\mathcal{A}_1,\mathcal{A}_2}(0) = (1, 1, \pm 1)^T$.

state of the system settling on the attractor $\mathcal{A}_1$, seen as the blue trajectory in Fig. 4. The other attractor, $\mathcal{A}_2$, indicated by the orange trajectory in Fig. 4 is arrived at once Eq. (8) is initialized from $\boldsymbol{x}^{\mathcal{A}_2}(0) = (1, 1, -1)^T$.

Trajectories on these attractors, $\mathcal{A}_1$ and $\mathcal{A}_2$, are considered the training data for Case I.

In Case II, we set the problem of reconstructing a double-scroll chaotic attractor in addition to the attractor $\mathcal{A}_2$ from Fig. 4. This particular double-scroll chaotic attractor, which we call $\mathcal{B}_1$, is reached by initializing Eq. (8) with $\boldsymbol{x}^{\mathcal{B}_1}(0) = (1, 1, 1)^T$ and setting the system parameters as $(a, b, c, d) = (5, 8, 2, 2)$ as shown in Fig. 5.

In Case III, we consider reconstructing $\mathcal{A}_2$ and the chaotic butterfly attractor, $\mathcal{L}$, generated by the Lorenz system.[18]

We remark that the Lorenz attractor $\mathcal{L}$ and the attractors chosen from Eq. (8) share no common region in the state space. Otherwise, this adds a further level of difficulty in training a RC to distinguish which attractor a given trajectory belongs to. Alternatively, to expose further criteria needed for the RC to exhibit multifunctionality, we suggest studying the effect of decreasing the distance between two disjoint attractors.

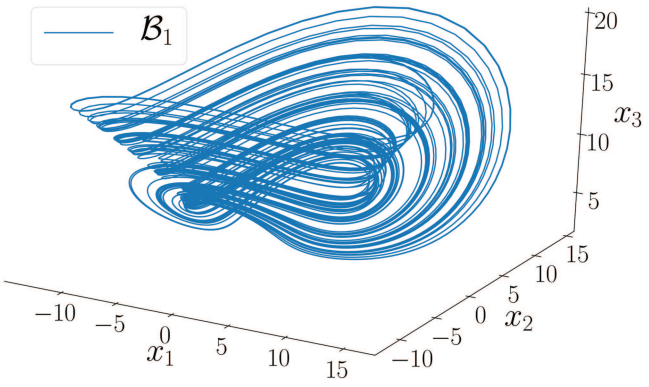


FIG. 5. The attractor, $\mathcal{B}_1$, arrived at by computing solutions of the system [Eq. (8)] initialized from $\boldsymbol{x}^{\mathcal{B}_1}(0) = (1, 1, 1)^T$ with parameter values set to $(a, b, c, d) = (5, 8, 2, 2)$.

IV. RESULTS

In this section, we illustrate the events in which a RC was trained to exhibit multifunctionality by successfully reconstructing the coexistence of attractors as specified in Cases I–III. We focus on establishing the optimal blending of the training data with respect to ρ so as to determine the regions in which multifunctionality was achieved. Following these observations, we examine the circumstances in which the reconstruction of a given attractor fails and detect a number of "untrained attractors"; attractors residing within the prediction state space were not part of the training. We then track the evolution of these attractors with respect to α and ρ and uncover a number of "behind-the-scenes" bifurcations.

A. Attractor reconstruction

There are many means of assessing the accuracy of a predicted time series. In this work, we choose to calculate $\theta_{\mathcal{S}}(\mathbf{W}_{out}^{\mathcal{S}})$ as the Normalized Root Mean Square Error (NRMSE) of the prediction in comparison with the target time series averaged over all state variables of a given attractor $\mathcal{S}$. $\theta_{\mathcal{S}}(\mathbf{W}_{out}^{\mathcal{S}})_i$ for the ith state variable is calculated as

$$\theta_{\mathcal{S}}(\mathbf{W}_{out}^{\mathcal{S}})_i = \frac{\sqrt{\frac{1}{t_{predict}-t^*}\sum_{t=t_{predict}-t^*}^{t_{predict}}\left(u_i(t)-\hat{u}_i(t)\right)^2}}{|\max(u_i(t))-\min(u_i(t))|}. \tag{9}$$

In our results, we set $t_{predict} = 600$ as the prediction end time and $t_{predict} - t^*$ is the time that error sampling begins from. $u_i(t)$ and $\hat{u}_i(t)$ are the ith state variables of the target and predicted time series. The $\max(\cdot)$ and $\min(\cdot)$ functions are measures of the maximum and minimum value of the time series evaluated from $t_{predict} - t^*$ to $t_{predict}$. The closer $\theta_{\mathcal{S}}(\mathbf{W}_{out}^{\mathcal{S}})$ is to 0, the more accurate the prediction of $\mathcal{S}$. It was found empirically that for $\theta_{\mathcal{S}}(\mathbf{W}_{out}^{\mathcal{S}}) > \delta = 0.35$, attractor reconstruction fails. Throughout our work, we keep $\beta = 10^{-2}$, $\sigma = 0.2$ in Cases I and III and let $\sigma = 0.4$ in Case II.

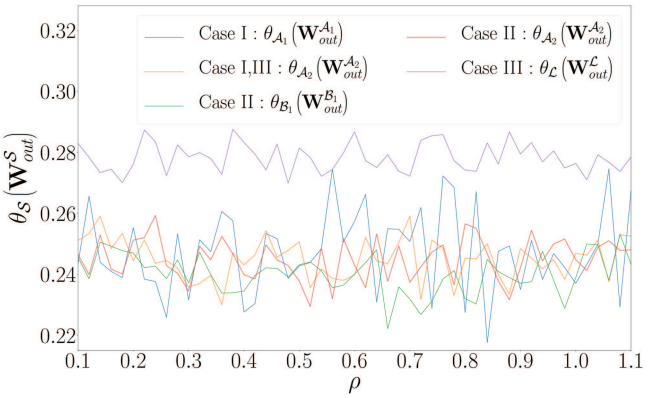
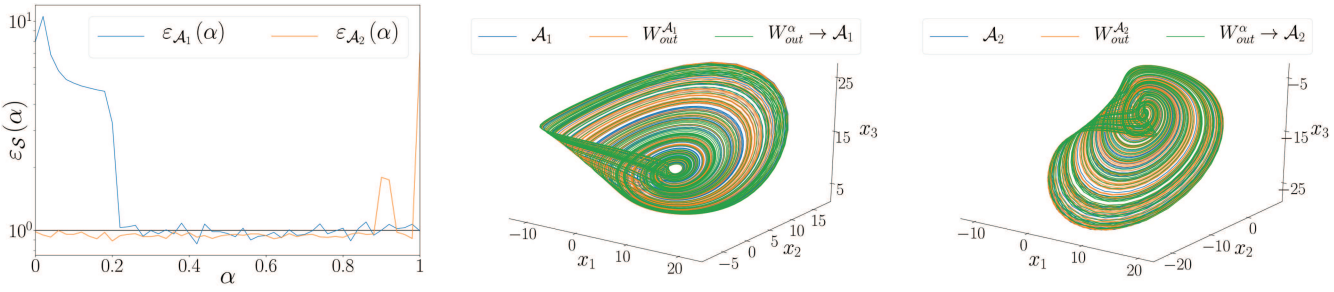


FIG. 6. $\theta_{\mathcal{S}}(\mathbf{W}_{out}^{\mathcal{S}})$ vs ρ when using the task specific matrices to reconstruct the attractor $\mathcal{S}$ from Cases I–III.



(a) $\varepsilon_{\mathcal{S}}(\alpha)$ vs. α (b) Reconstruction of $\mathcal{A}_1$ (c) Reconstruction of $\mathcal{A}_2$

FIG. 7. Case I ($\rho = 0.7$): (a) $\varepsilon_{\mathcal{A}_1}$ and $\varepsilon_{\mathcal{A}_2}$ vs α. (b) and (c) Attractor reconstruction using $\mathbf{W}_{out}^{\mathcal{A}_1}$ and $\mathbf{W}_{out}^{\mathcal{A}_2}$ and $\mathbf{W}_{out}^{\alpha}$ for $\alpha = 0.5$.

First, we conduct an error analysis of the predicted time series when using the task specific matrices, $\mathbf{W}_{out}^{\mathcal{A}_1}$, $\mathbf{W}_{out}^{\mathcal{A}_2}$, $\mathbf{W}_{out}^{\mathcal{B}_1}$, and $\mathbf{W}_{out}^{\mathcal{L}}$, to determine if the attractors can be reconstructed in the usual sense. In Fig. 6, we provide a picture, where having set $t^* = t_{train}$, the error analysis indicates that attractor reconstruction was achieved for all $\rho \in [0.1, 1.1]$ using each of the task specific matrices as $\theta_{\mathcal{S}}(\mathbf{W}_{out}^{\mathcal{S}}) < \delta$ in Cases I–III.

With this established, we now search for values of α that give rise to a single readout matrix, $\mathbf{W}_{out}^{\alpha}$, which allows the RC to reconstruct either of the attractors specified in Cases I–III. After applying the blending technique, we calculate the following error function of the particular readout matrix used to reconstruct a given attractor:

$$\varepsilon_{\mathcal{S}}(\alpha) = \theta_{\mathcal{S}}\left(\mathbf{W}_{out}^{\alpha}\right) / \theta_{\mathcal{S}}\left(\mathbf{W}_{out}^{\mathcal{S}}\right). \tag{10}$$

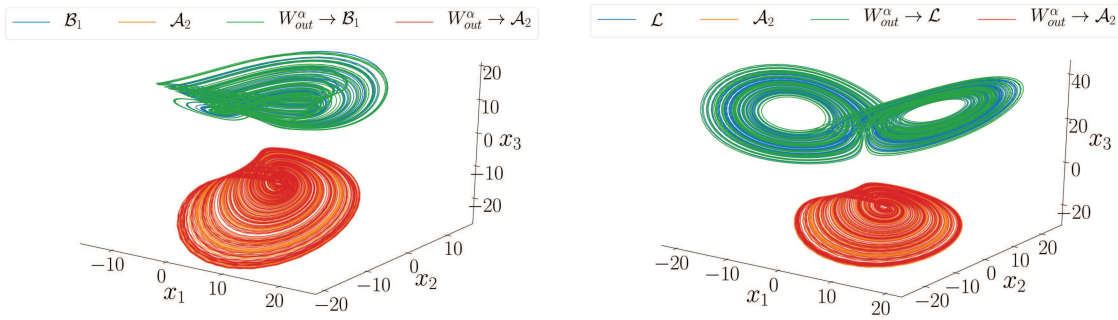
Here, $\theta_{\mathcal{S}}\left(\mathbf{W}_{out}^{\alpha}\right)$ is the NRMSE when using the blending technique to reconstruct an attractor $\mathcal{S}$ with a certain α. This measure of error implies that if $\varepsilon_{\mathcal{S}}(\alpha) < 1$, the prediction of $\mathcal{S}$ was more accurate when using $\mathbf{W}_{out}^{\alpha}$ over $\mathbf{W}_{out}^{\mathcal{S}}$ with the opposite being said if $\varepsilon_{\mathcal{S}}(\alpha) > 1$.

Starting with the task of reconstructing attractors from a multistable system in Case I, we set $\rho = 0.7$ with the resulting $\varepsilon_{\mathcal{S}}(\alpha)$ vs α plot shown in Fig. 7(a).

Here, we see that on one hand as α is increased from 0, there is a large decrease in $\varepsilon_{\mathcal{A}_1}$ and stays relatively close to 1 for $\alpha \gtrsim 0.22$, while on the other hand, $\varepsilon_{\mathcal{A}_2}$ remains relatively near 1 for $\alpha \lesssim 0.88$ and then grows as α is increased thereafter. From this, we deduce that for $0.22 \lesssim \alpha \lesssim 0.88$, multifunctionality is expressed by the RC. We illustrate in Figs. 7(b) and 7(c) a successful implementation of the blending technique for $\alpha = 0.5$. The predicted trajectory on $\mathcal{A}_1$ and $\mathcal{A}_2$ when using the task specific matrices, $\mathbf{W}_{out}^{\mathcal{A}_1}$ and $\mathbf{W}_{out}^{\mathcal{A}_2}$, is plotted in orange, and the predictions using the multifunctional output matrix, $\mathbf{W}_{out}^{\alpha}$, are plotted in green. A trajectory on the actual attractors from Eq. (8) in each figure is plotted in blue.

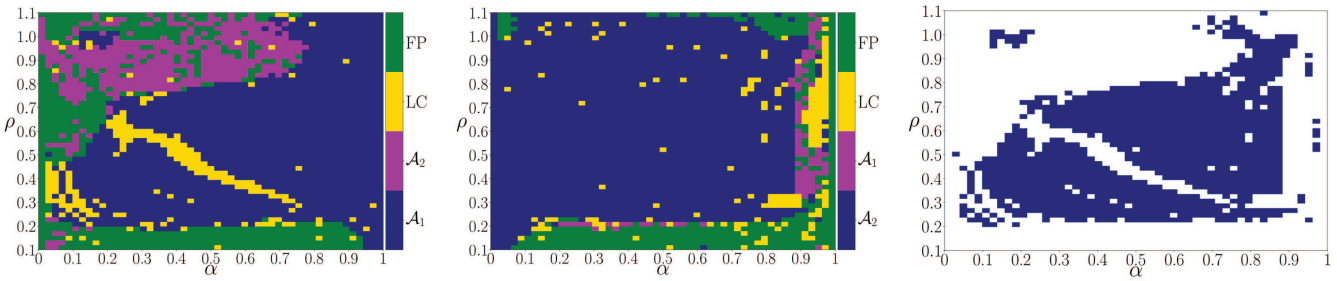
We conduct a similar analysis for Cases II and III with Fig. 8 illustrating examples where the RC was successfully trained to be multifunctional. We see in Fig. 8(a) that when training the RC to reconstruct both $\mathcal{B}_1$ and $\mathcal{A}_2$ in Case II, the desired coexistence of chaotic attractors is achieved for $\rho = 0.3$ and $\alpha = 0.5$. We find that in Case III, when setting $\rho = 0.85$, the RC is successfully trained to express multifunctionality as it can reconstruct both $\mathcal{L}$ and $\mathcal{A}_2$ for $\alpha = 0.65$ as seen in Fig. 8(b).

The results illustrated in Figs. 7 and 8 show that a RC can be trained to exhibit multifunctionality. Furthermore, this broadens the current set of applications a RC is capable of. We show that instead of having to change parameters in a system for it to exhibit a



(a) Case II: Reconstructing attractors $\mathcal{B}_1$ and $\mathcal{A}_2$. (b) Case III: Reconstructing the chaotic Lorenz attractor $\mathcal{L}$ and $\mathcal{A}_2$.

FIG. 8. Illustration of attractor reconstruction in (a) Case II for $(\alpha, \rho) = (0.5, 0.3)$ and (b) Case III for $(\alpha, \rho) = (0.65, 0.85)$.



(a) IC: $\mathcal{A}_1$ (b) IC: $\mathcal{A}_2$ (c) Regions of multifunctionality

FIG. 9. (a) and (b) Long-term behavior of prediction in the (α, ρ)-plane for Case I. Each color characterizes the attractor the RC eventually settles to starting from a particular IC: Initial Condition. (c) Plotted in blue are the regions of multifunctionality.

different behavior, one can merge separate modes of operation from various parameter choices of the system to coexist in the prediction state space of a multifunctional RC. We also demonstrate that the combination of attractors is not limited to a single system as it is possible to combine attractors from different systems to coexist. We remark that given this ability, there is the prospect of designing an appropriate controller for Eq. (6) to switch between attractors. For further reading on "inter-attractor control," see Richter.[34]

Although we have illustrated instances in which appropriate values of α were found to give rise to multifunctionality, this cannot be said for all α values or for other choices of γ, ρ, σ, and β. Moreover, relying on an error analysis alone is not sufficient enough to classify the parameter regions in which multifunctionality is achieved.

This RC is designed such that if attractor reconstruction fails, then the predicted trajectory will not blow up to infinity in a finite amount of time. However, the prediction can decay toward some other stable attractor. Furthermore, given the nature of our study, it is also possible that the RCs predicted trajectory on one chaotic attractor can switch to the other chaotic attractor.

Following this argument, in Sec. IV B, we employ a means of characterizing the resultant attractor that the prediction settles to and from this identify the regions in the (α, ρ)-plane where multifunctionality is achieved.

B. Exploring multifunctionality in the (α, ρ)-plane

In this section, we analyze the long-term behavior of the RC in the prediction stage [Eq. (6)] initialized with $\hat{\mathbf{r}}(0)$ corresponding to $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{B}_1$, or $\mathcal{L}$ (for the appropriate case) and trained for a given $\alpha \in [0, 1]$ and $\rho \in [0.1, 1.1]$.

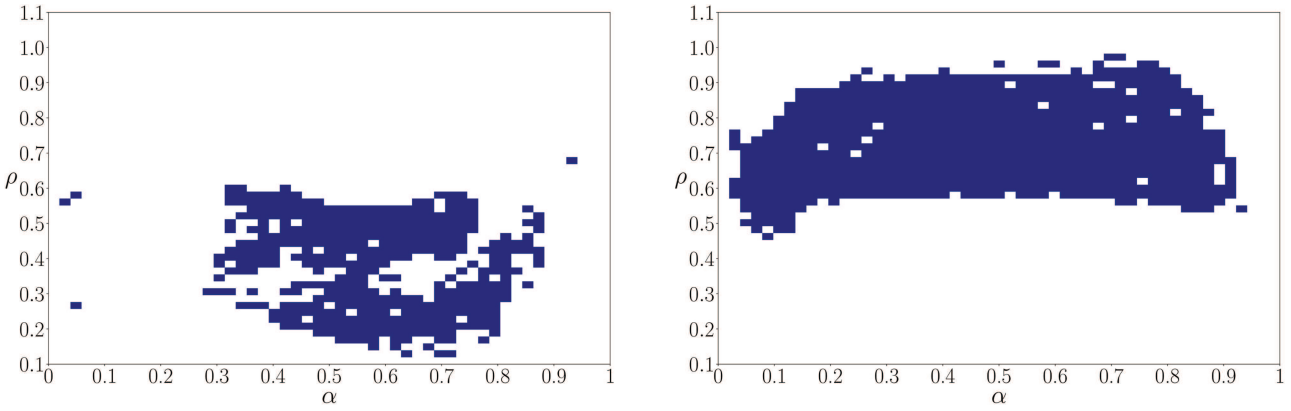
We choose to assign a color to each point in the (α, ρ)-plane that characterizes the prediction of a given attractor. More specifically, a point in the (α, ρ)-plane is colored:

1. Yellow—if the predicted time series is periodic for $t_{predict} - 40 \leq t \leq t_{predict}$, we say that the prediction has decayed to some limit cycle.
2. Green—if the predicted time series remains constant for $t_{predict} - 10 \leq t \leq t_{predict}$, we say that the prediction has decayed to some fixed point.
3. Purple—if for $t^* = 40$, $\theta_{\mathcal{S}}\left(\mathbf{W}_{out}^{\alpha}\right) \leq \delta$ for the predicted time series in comparison with the target time series of the other chaotic attractor, we then say that the prediction has switched from one chaotic attractor to the other.
4. Blue—if conditions 1–3 are not fulfilled and if for $t^* = 40$ that $\theta_{\mathcal{S}}\left(\mathbf{W}_{out}^{\alpha}\right) \leq \delta$ for a given attractor $\mathcal{S}$, we then say that the climate of $\mathcal{S}$ was well reconstructed.
5. Red—if otherwise to signal that closer inspection of the prediction is needed.

The result of this analysis for Case I is shown in Fig. 9.

As expected, we see that attractor reconstruction is impossible when initializing the RC in a prediction mode with $\hat{\mathbf{r}}(0)$ corresponding to $\mathcal{A}_1$ in Fig. 9(a) [$\mathcal{A}_2$ in Fig. 9(b)] for $\alpha = 0$ (1). We now know that the prediction in this scenario consistently decays toward some fixed point for all values of ρ. However, when increasing (decreasing) α from 0 (1), there is a more varied sequence of events for a given ρ. Prior to achieving attractor reconstruction, the predicted trajectory on $\mathcal{A}_1$ ($\mathcal{A}_2$) at times tends toward some limit cycle or switches to $\mathcal{A}_2$ ($\mathcal{A}_1$). Figure 9(a) also shows that for large ρ, the RC favors the reconstruction of $\mathcal{A}_2$ as opposed to $\mathcal{A}_1$ where the prediction of $\mathcal{A}_1$ mainly switches to $\mathcal{A}_2$ until some critical α value where attractor reconstruction is achieved. Nevertheless, there is some middle ground where both attractors can be successfully reconstructed for a given pair of α and ρ, and this common blue area between Figs. 9(a)–9(b) is the regions in which we say multifunctionality was achieved. We provide a separate picture in Fig. 9(c) to explicitly show these regions. Through the same reasoning, we show in Figs. 10(a) and 10(b) the regions in the (α, ρ)-plane in which the RC was successfully trained to express multifunctional behavior for both Cases II and III.

We see a relatively large common blue region in the (α, ρ)-plane for Case I in Fig. 9(c). However, for Cases II and III, the regions of multifunctionality in Figs. 10(a) and 10(b) are relatively much smaller. Cases II and III require a higher level of dynamical flexibility from the RC as not only do the chaotic attractors come from different settings of Eq. (8) and different systems entirely but also vary characteristically, i.e., single-scroll and double-scroll. A particular setup of the RC may happen to favor reconstructing one



(a) Case II (b) Case III

FIG. 10. Regions of multifunctionality in the (α, ρ)-plane for (a) Case II and (b) Case III.

flavor of attractor over the other, thus a contributing factor toward the reduced regions of multifunctionality seen here.

While in each of the explored cases, the target chaotic attractors are separated in the state space, they do share some dynamical similarities that may be just as vital in order to achieve multifunctionality. For example, the chosen attractors in all three of the studied cases evolve along a similar timescale. We now consider the pair of chaotic attractors in Case I and investigate the relationship between the RC capacity to exhibit multifunctionality when changing the timescale of $\mathcal{A}_1$ while keeping the timescale of $\mathcal{A}_2$ fixed. To do this, we introduce a new parameter, $\Delta^{\mathcal{A}_1}$, which is used to control the timescale of $\mathcal{A}_1$. We multiply the RHS of Eq. (8) by $\Delta^{\mathcal{A}_1}$ and initialize the system with $\boldsymbol{x}^{\mathcal{A}_1}(0) = (1, 1, 1)^T$ to generate solutions of $\mathcal{A}_1$ with a modified timescale. If $\Delta^{\mathcal{A}_1} < 1$, the dynamics are slowed down, and if $\Delta^{\mathcal{A}_1} > 1$, the dynamics are sped up.

We now investigate and identify the regions in the $(\Delta^{\mathcal{A}_1}, \rho)$-plane where multifunctionality is achieved. We do this by employing the previous method of characterizing the long-term behavior of the RC prediction of either $\mathcal{A}_1$ or $\mathcal{A}_2$ when using W^{α}_{out} with $\alpha = 0.5$ and take the common regions where the RC can successfully reconstruct either attractor depending on the IC. The result of this is shown in Fig. 11.

Here, in Fig. 11, we see that if $\mathcal{A}_1$ evolves along a relatively longer or shorter timescale in comparison with $\mathcal{A}_2$, then there is a point where multifunctionality is lost for all ρ values. This reveals the limits upon which multifunctionality can be achieved in this scenario by exposing the RC dynamical capacity to facilitate the coexistence of increasingly dissimilar chaotic attractors in its prediction state space.

Now, if we focus on the prediction of $\mathcal{A}_1$ for $\rho = 0.7$ as α is increased from 0 in Fig. 9(a), we see that before multifunctionality is achieved, the prediction repeatedly falls to a fixed point. We plot these fixed points for various values of α in Fig. 12. Also seen here is a case where the prediction switches from $\mathcal{A}_1$ to $\mathcal{A}_2$ when $\alpha = 0.2$.

We ask, what happens to these fixed points in Fig. 12 as multifunctionality is achieved? Is it the case that the successful reconstruction of $\mathcal{A}_1$ entirely takes the place of these fixed points in the prediction state space or do these fixed points still exist and we are unable to see them? Furthermore, are there other attractors lurking in this prediction state space that we are not immediately aware of? We delve further into these questions in Sec. IV C and highlight the important role of ICs in the prediction stage of the RC.

C. Detecting untrained attractors

To get a broader picture of the prediction state space for a given ρ and α in Case I, we initialize the RC in the prediction mode with many (1000) random initial conditions (RICs) and observe the resultant trajectories.

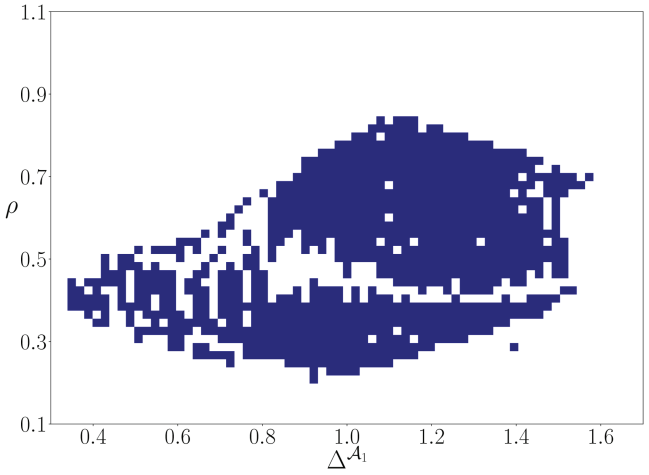


FIG. 11. Regions of multifunctionality in the $(\Delta^{\mathcal{A}_1}, \rho)$-plane when changing the timescale of $\mathcal{A}_1$ in Case I with timescale parameter $\Delta^{\mathcal{A}_1}$.

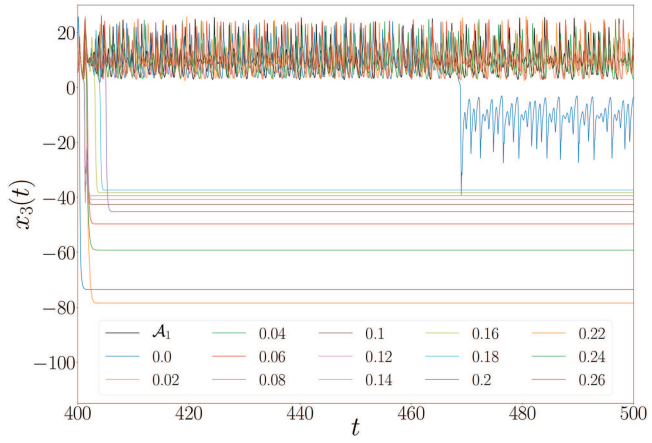


FIG. 12. Case I ($\rho = 0.7$, IC: $\mathcal{A}_1$): $x_3(t)$ vs t for $\mathcal{A}_1$ colored in black and the predicted trajectories for various values of α as indicated in the plot legend.

In Fig. 13, we set $\rho = 0.7$ and train the RC for $\alpha = 0.45$, 0.50, and 0.55. We plot $x_3^{\xi_i}$ as the trajectories of the x_3 variable as predicted by the RC starting from the ith RIC.

For $\alpha = 0.5$, we see that there exists a stable fixed point located within the middle of both reconstructed chaotic attractors. As we begin to apply more weight to the data from $\mathcal{A}_2$ with $\alpha = 0.45$, we see another stable fixed point appearing closer to the reconstructed $\mathcal{A}_2$. Similarly, for $\alpha = 0.55$, we find a stable fixed point

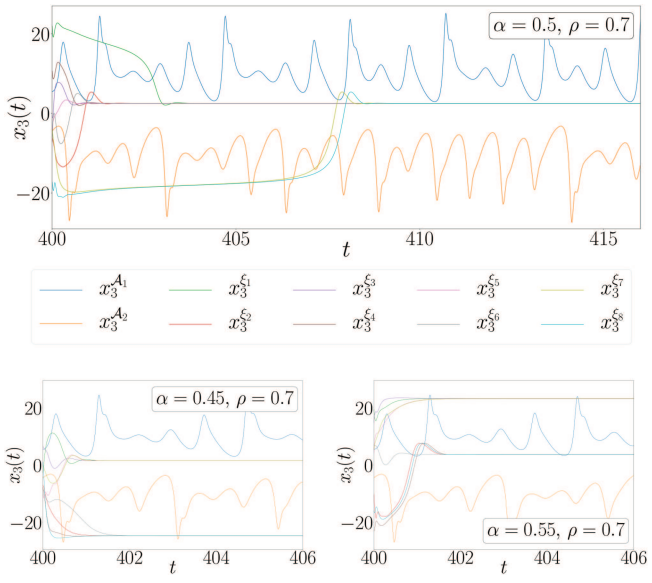


FIG. 13. Time trace of the predicted x_3 variable, with the RC trained for a given α and ρ as indicated above each plot, starting from the random initial condition ξ_i for $i = 1, 2, \ldots$.

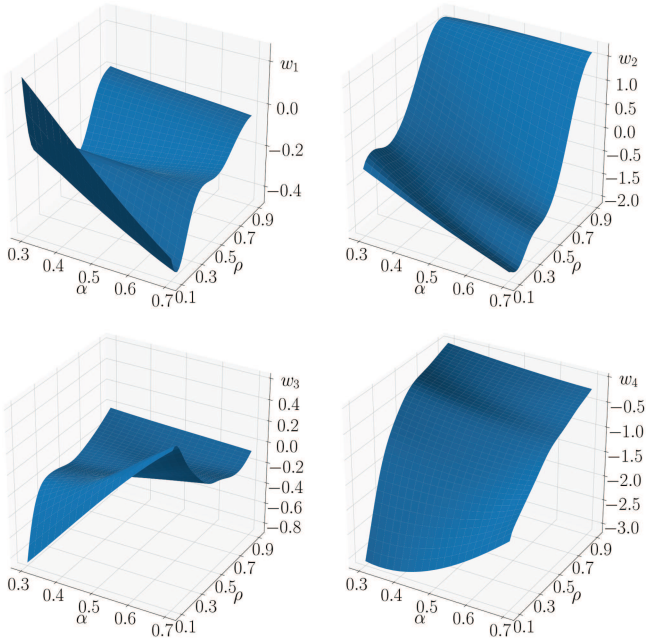


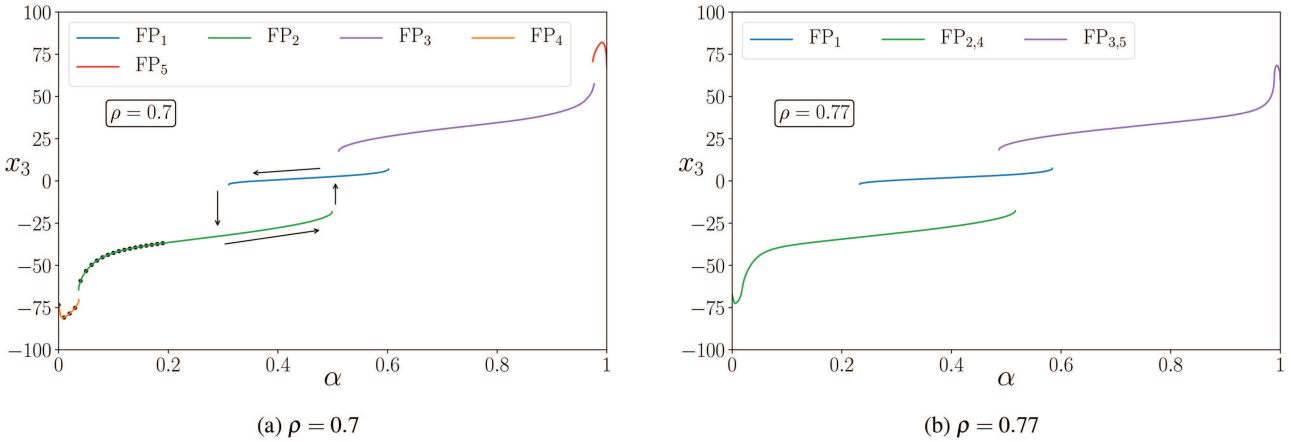
FIG. 14. The evolution of four randomly chosen elements of the $\mathbf{W}_{out}^{\alpha}$ matrix with respect to changes in α and ρ.

closer to the reconstructed $\mathcal{A}_1$. Therefore, while the RC was successfully trained to reconstruct the climate of both $\mathcal{A}_1$ and $\mathcal{A}_2$, there are additional attractors populating the prediction state space that were not involved in the training, and we call these the "untrained attractors." The results in Fig. 13 show that small changes in α can give rise to a bistability of untrained attractors. This suggests that there are some "behind-the-scenes" bifurcations taking place in the prediction state space. However, further discussion is needed to consider α as a bifurcation parameter of the RC.

While ρ is a parameter of the RC itself, α is a parameter that is strictly involved in the training procedure. Therefore, to consider α as a bifurcation parameter of the RC, we need to assess if for a small change in α, there is a relatively smooth change in the elements of the $\mathbf{W}_{out}^{\alpha}$ matrix generated from a specific combination of α and ρ. In other words, a continuous function that maps $(\alpha, \rho) \rightarrow \mathbf{W}_{out}^{\alpha}$ needs to exist.

To expand upon this notion, we consider some randomly chosen elements of $\mathbf{W}_{out}^{\alpha}$ and observe their evolution with respect to α and ρ. The result of this is shown in Fig. 14 for four randomly chosen elements of $\mathbf{W}_{out}^{\alpha}$.

The smoothness of these surface plots indicates that a relatively small change in α or ρ in turn gives rise to a small change in the elements of $\mathbf{W}_{out}^{\alpha}$ to which we generalize contributes to an overall change in the dynamics of the RC. From this, we now consider α as a bifurcation parameter of the RC. We can view our training procedure as a smooth map from the hyper-parameters ρ



(a) $\rho = 0.7$ (b) $\rho = 0.77$

FIG. 15. Evolution of untrained attractors (stable branches of fixed points) in (α, x_3)-plane for (a) $\rho = 0.7$ and (b) $\rho = 0.77$.

and α to a matrix $\mathbf{W}_{out}^{\alpha}$ in the $\mathbb{R}^{D\times 2N}$ space. Therefore, by considering these hyper-parameters, ρ and α, this mapping generates a two-dimensional structure in the $\mathbb{R}^{D\times 2N}$ space.

Our study now moves toward tracking the evolution of these untrained attractors and identifying some of these "behind-the-scenes" bifurcations.

D. Bifurcation analysis of untrained attractors

In this section, we track the evolution of these untrained attractors first in the (α, x_3)-plane for a given ρ and then in the (α, ρ)-plane.

We do this by initializing the state of the RC, trained for a certain α and ρ, with one of the corresponding fixed points shown in Fig. 13. We track the evolution of this fixed point with respect to α by repeating the process of incrementally changing α, retraining and initializing the state of the RC with the fixed point corresponding to the previous α. The result of this for $\rho = 0.7$ is shown in Fig. 15(a).

Here, we see the extent of the bistabilities found in Fig. 13. We also find branches of fixed points and bistabilities closer to the end points of α. The evolution with respect to α of the previously found fixed point located in the middle of the chaotic attractors is now labeled as the branch FP_1, and the evolution of the fixed points closer to $\mathcal{A}_2$ and $\mathcal{A}_1$ is labeled as FP_2 and FP_3, respectively. We label the evolution of the fixed points closest to $\alpha = 0$ as the branch FP_4 and those closest to $\alpha = 1$ as FP_5.

Giving rise to these bistabilities are the hysteresis cycles created here. As indicated by the arrows in Fig. 15(a), when moving along the FP_2 branch, as α is increased, there is a point where the state of the RC jumps to the FP_1 branch. After this transition, if α was instead decreased, we then remain and continue to track along the FP_1 branch to the point where the state of the RC returns to the FP_2 branch and so on.

Figure 15(a) also demonstrates that when the RC fails to reconstruct $\mathcal{A}_1$, then the predicted trajectory falls onto the FP_4 and FP_2 branches. The black points plotted in Fig. 15(a) are the fixed points that the predictions of $\mathcal{A}_1$ decay toward in Fig. 12. As these black points line up directly with the branches of FP_4 and FP_2, we conclude that as attractor reconstruction of $\mathcal{A}_1$ is achieved that these fixed points do not suddenly disappear, but that their existence is intrinsic to the dynamics of this RC setup. In modes of failure, these branches of fixed points provide routes to stability should the predicted trajectory fall off $\mathcal{A}_1$. This also gives greater insight to the behavior of the RC at the boundaries of multifunctionality. It is also important to highlight that the fixed points located at $\alpha = 0$ and 1 also occur in the task specific systems. More specifically, if the RC was successfully trained on data from only $\mathcal{A}_1$, then within this prediction state space exists the reconstructed attractor $\mathcal{A}_1$ and the same fixed point, FP_5, that we find in the multifunctional setup. The same can be said in relation to $\mathcal{A}_2$ and FP_4. Furthermore, these untrained attractors have dynamics of their own and interact among themselves giving rise to these "behind-the-scenes" bifurcations.

These stable branches of equilibria would ordinarily be connected by branches of unstable equilibria, but given the nature of our approach, these cannot be explicitly determined. However, as the end points of these branches are seemingly being drawn together, i.e., the right and left end points of both FP_4 and FP_2 and likewise for FP_3 and FP_5, we infer that this is a signature of the existence of unstable branches of equilibria and evidence that at these end points are saddle-node (SN) bifurcations. To help strengthen this claim, we increase ρ to 0.77 and track the evolution of the fixed points as before. The result of this is shown in Fig. 15(b) where the right and left end points of FP_4 and FP_2 as well as FP_3 and FP_5 have connected together resulting in two branches of fixed points which we call $FP_{2,4}$ and $FP_{3,5}$. This particular behavior is indicative of two cusp bifurcations taking place about $\alpha \approx 0.034$ and 0.988 as ρ is increased from 0.7 to 0.77. In addition, we see a region of "tristability" here for $0.4867 \leq \alpha \leq 0.5165$ where there is an overlap between all three branches.

We continue exploring and characterizing the behavior of these untrained attractors by tracking their evolution in the (α, ρ)-plane. The result of this is depicted in Fig. 16.

In this picture, we see the previously mentioned cusp bifurcations taking place at $(\alpha, \rho) = (0.0337, 0.71)$ and $(0.9875, 0.76)$.

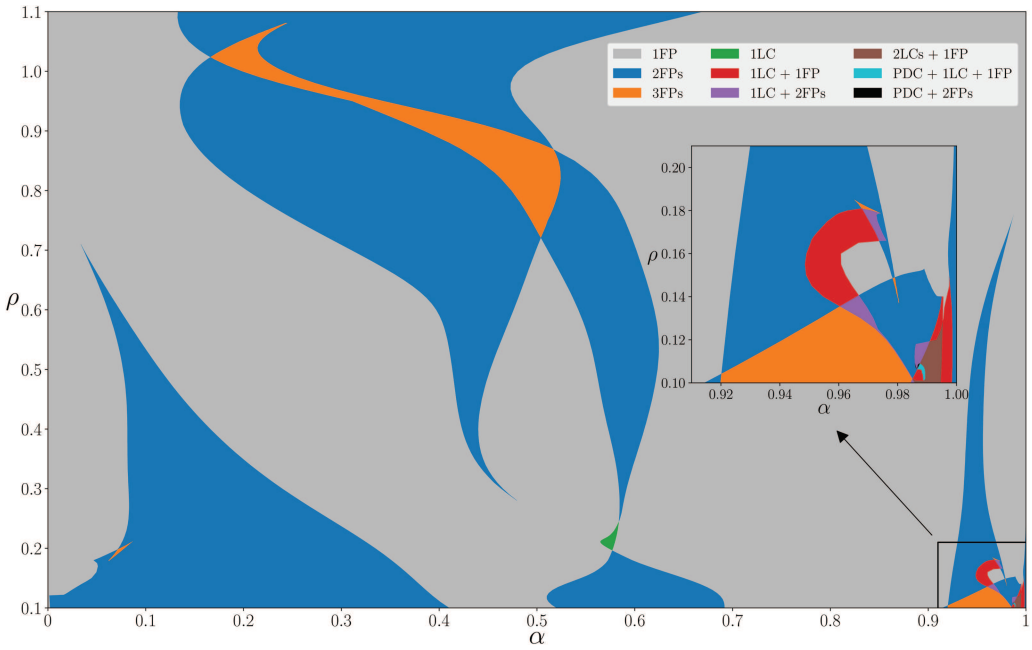


FIG. 16. Classification of "untrained attractors" in the $(\alpha,\ \rho)$-plane. FP, fixed point; LC, limit cycle; PDC, period doubling cascade).

We also find a cusp bifurcation of the FP_1 and FP_2 branches taking place at $(\alpha,\ \rho) \approx (0.48,\ 0.27)$. We denote $FP_{1,2}$ as the branch of stable fixed points emerging from this cusp bifurcation.

Figure 16 also reveals the existence of several limit cycles. As ρ is decreased from 0.27, the bistability between FP_3 and $FP_{1,2}$ is lost at $(\alpha,\ \rho) \approx (0.5837,\ 0.2449)$. As these branches begin to drift apart, we show in Fig. 17 for $\rho = 0.21$ that a period-1 limit cycle, LC_1, is born in this gap. Plotted here is the evolution of the maximum and minimum values of LC_1 vs α. As ρ is decreased further, the bistability between the FP_3 and $FP_{1,2}$ branches resumes from $(\alpha,\ \rho) \approx (0.577,\ 0.1965)$ resulting in the death of LC_1.

We also see a relatively small region of tristability between $FP_{1,2}$, FP_7, and FP_4 in Fig. 17. While the left end of $FP_{1,2}$ had at one time appeared to be growing toward and potentially connecting to FP_4 in another cusp bifurcation, instead it has broken off due to a cusp bifurcation taking place on $FP_{1,2}$ at $(\alpha,\ \rho) \approx (0.212,\ 0.087)$ resulting in a new branch of fixed points FP_7 as seen in Fig. 17. The existence of FP_7 is relatively brief as its own end points are quickly drawn together and disappear entirely at $(\alpha,\ \rho) \approx (0.062,\ 0.179)$ in Fig. 16. This particular behavior also occurs on FP_3 where we will later provide a more detailed picture and explanation of these events. We also find the beginnings of a branch of stable fixed points emerging from the right hand side of Fig. 17 which we label as FP_6. Later, we will discuss some interesting properties emanating from this branch.

Given the abundant variety of behavior exhibited by the untrained attractors for large α and small ρ, as seen in the inset plot of Fig. 16, we provide a more expansive picture in Fig. 18 depicting how certain bifurcations arise.

The cusp bifurcation taking place at $(\alpha, \rho) = (0.965, 0.185)$ gives rise to a relatively small tristable region. This cusp bifurcation originates from a buckling of the FP_3 branch where a new branch of stable fixed points emerges, labeled as FP_8 in Fig. 18.

A limit cycle is born at the right end point of FP_3 for $\rho \approx 0.181$. In Fig. 18, we plot the evolution of the maximum and minimum x_3-values of this period-1 limit cycle which we label as LC_2. Close to this cusp bifurcation we see for $\rho = 0.18$ that LC_2 is contained within a Hopf-bubble as it exists between two supercritical Hopf

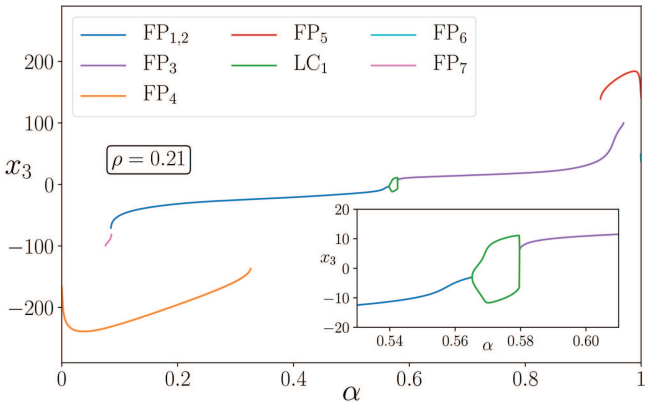


FIG. 17. Tracking untrained attractors for $\rho = 0.21$: Max and min values of limit cycle, LC_1, born at the right and left end points of the $FP_{1,2}$ and FP_3 branches. Also seen are the FP_4, FP_5, FP_6, and FP_7 branches of stable fixed points.

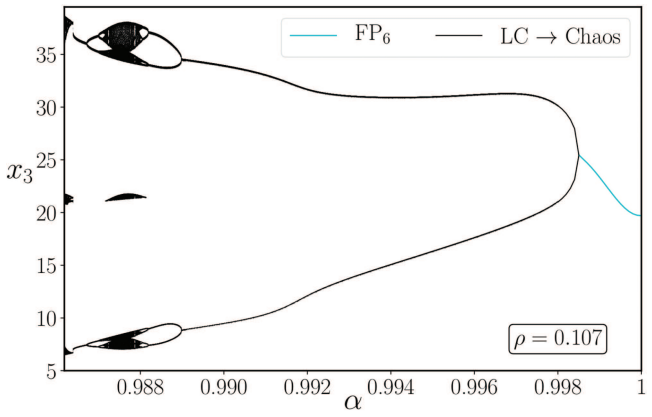


FIG. 18. Behavior of the untrained attractors in the predicted x_3 direction for large α and small ρ.

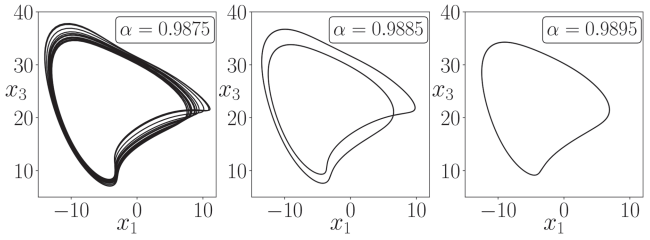


FIG. 19. ($\rho = 0.107$): Evolution of FP_6 as it transitions to a limit cycle and then to a chaotic attractor through an infinite sequence of period-doubling bifurcations. Also shown are snapshots of this limit cycle along its route to chaos.

bifurcations. However, as ρ is decreased further, the amplitude of oscillation of LC_2 increases and the bubble bursts. We also find that LC_2 undergoes period-doubling (PD) bifurcations, the first of which was found nearby $(\alpha, \rho) = (0.959, 0.166)$. These PD bifurcations attribute to the significant reduction in α-values for which we are able to track LC_2. Also shown in Fig. 18 are cases where LC_2 subsequently encounters another PD bifurcation giving rise to a period-4 limit cycle.

The existence of FP_8 is relatively short; as ρ is decreased, its end points are drawn closer together until this branch becomes but a single point-like attractor at $(\alpha, \rho) \approx (0.98, 0.137)$ and we are no longer able to track. This event and that mentioned earlier regarding FP_7 is evidence that further flavors of cusp-like bifurcations take place in the prediction state space. A codimension-3 bifurcation analysis (involving either σ, β, or γ) could, for example, unveil a swallowtail bifurcation point (the point at which two cusp branches collide). For reading on more cusp-like bifurcations, see Kuznetsov[35] or Guckenheimer and Holmes.[36] Alternatively, if swallowtail bifurcation points were found to take place in lower dimensional representations of Eq. (6) (for $N = 2, 3, \ldots$), then their existence could be generalized to the higher dimensional picture. The benefit of reducing the dimension would allow for the use of numerical continuation software such as AUTO[37] in Eq. (6). Moreover, this approach facilitates the study of unstable attractors and in identifying further dynamical features inherent to Eq. (6). We leave this for future work.

As shown in Fig. 18, we find another branch of fixed points which we label as FP_9. For $\rho = 0.135$, we see here that at the right end point of FP_9, a limit cycle labeled as LC_4 is born. Increasing ρ results in the loss of LC_4 as the end points of FP_9 are being drawn closer together where eventually at $(\alpha, \rho) = (0.989, 0.153)$, we are no longer able to track.

Throughout Fig. 18, we plot the evolution of the previously mentioned branch, FP_6. As ρ is decreased, a period-1 limit cycle, labeled as LC_3, is born from the left end point of FP_6. The brown region in Fig. 16 depicts the coexistence of LC_3, LC_4, and FP_5. However, as we follow the evolution of the local maxima and minima of LC_3, there are certain points in the (α, ρ)-plane where it undergoes a PD bifurcation. Furthermore, there are times at which one PD bifurcation leads to another and in turn triggers a period-doubling cascade (PDC), ultimately resulting in chaotic behavior. We find two relatively small distinct regions in Fig. 16 (colored in black and cyan) where PD bifurcations of LC_3 lead to PDCs. An example of this particular sequence of PD bifurcations is shown in Fig. 19 when setting $\rho = 0.107$.

Starting from $\alpha = 1$, we see in Fig. 19 how FP_6 evolves as α is decreased. We plot the local maxima and minima of LC_3 sprouting from the left end point of FP_6 for $\alpha = 0.9986$ and continue to track as α is reduced further. Here, we see the first of these PD bifurcations occurring at $\alpha \approx 0.989$ where there are now two distinct local maxima and minima of LC_3. This particular behavior is depicted below the bifurcation diagram in Fig. 19 with snapshots of LC_3 as it travels along its route to a PDC in the (x_1, x_3) prediction state space. Here, we see how LC_3 transitions from period-1 at $\alpha = 0.9895$ to period-2 at $\alpha = 0.9885$. Decreasing α further to 0.9875, we see that there has been an infinite number of PD bifurcations giving rise to a chaotic attractor. Periodic behavior briefly resumes for decreasing α further before entering another PDC as a second bout of chaos begins at $\alpha = 0.986\,18$ and ends at $0.986\,38$.

Despite that multifunctionality is not achieved in the relatively small regions in which we find these PDCs, it is a remarkable result as it shows that when attempting to train the RC to promote a coexistence of two desired chaotic attractors, it is also possible for another chaotic attractor to exist in the background. Moreover, for a different choice of the other reservoir parameters, these PDCs could play a larger role in the prediction state space and further influence the RC ability to reconstruct a given attractor. Throughout our results, we have kept the topology of the RC matrices, $\mathbf{M}$ and $\mathbf{W}_{in}$ fixed. Given a different initialization of these matrices, we expect that quantitative changes in these bifurcation figures would occur, but that the main characteristics would remain.

Overall, these results indicate that regardless of the particular attractor one is attempting to reconstruct, there will always be some untrained attractor present in the prediction state space. The long-term characterization of the RC prediction of a given attractor in Figs. 9(a) and 9(b) combined with the classification of the untrained attractors in Fig. 16 provides us with a clearer picture of the prediction state space for a given α and ρ. When put together, we are able to see the regions in which multifunctionality was achieved, the

manner in which the prediction fails, and the particular untrained attractor it can tend toward. Moreover, one could argue that attractor reconstruction may fail if the basin of attraction of the desired attractor interferes with one of these untrained attractors.

V. CONCLUSION

In this paper, we have demonstrated that a Reservoir Computer (RC) can be trained to exhibit multifunctionality whereby, for a given initial condition, the climate of more than one chaotic attractor can be successfully reconstructed in its prediction state space.

In order to train a RC to express multifunctionality, we introduce the "blending technique" as a means to combine and weight data from two different sources. We test the flexibility of this technique by training the RC to reconstruct a coexistence of chaotic attractors from a system that already exhibits multistability, two parameter settings of a system, and two different systems entirely.

However, in order to achieve the desired outcome, there is a crucial dependence on certain reservoir and training parameters, in particular, the "blending" parameter, α, and the spectral radius of the reservoir's internal connection matrix, ρ. When varying these parameters, we find an abundance of nontrivial transitions between multifunctionality and modes of failure as there is a competition between attractors in the prediction state space of the RC for a given α and ρ. For example, when attempting to reconstruct the climate of a given chaotic attractor, there are times where the prediction can switch to the other. In this case, the RC is able to reconstruct the climate of only one chaotic attractor. This behavior comes as a consequence of training a RC to reconstruct the climate of more than one attractor. Furthermore, we see that in the event of failure, the predicted trajectory on a given chaotic attractor can tend toward a limit cycle or a fixed point.

On closer inspection, we find that even when the RC is successfully trained to express multifunctionality, there are additional attractors existing within the prediction state space that were not involved in the training, and we call these the "untrained attractors." Moreover, it is shown that these untrained attractors have dynamics of their own and play a significant role in the event that the desired attractor cannot be successfully reconstructed.

By tracking the evolution of these untrained attractors with respect to α and ρ, we have identified a number of "behind-the-scenes" bifurcations. In particular, when tracking the evolution of FP_6, we see in Fig. 19 how this branch of stable fixed points transitions to the limit cycle LC_3, which subsequently undergoes a series of period-doubling bifurcations to the point of provoking a period-doubling cascade inevitably leading toward the creation of a chaotic attractor. Therefore, while we try to train the RC to reconstruct a coexistence of two specific chaotic attractors in its prediction state space, there is another chaotic attractor created in a *sub rosa* fashion.

Much like our investigation of the untrained attractors in the RC prediction state space, there is also evidence of similarly occurring events in the brain, which influence and disrupt its normal behavior. It is understood that neurological disorders such as Parkinson's disease or epilepsy occur as a result of certain active regions of the brain deviating from its normal state of operation and then becoming trapped within some undesirable behavior beyond which it may not be able to resume its normal function. The effort is often made to model and control these events in order to counteract the effects of these illnesses.[38–40]

The ability to combine attractors from different sources to then coexist in the same state space broadens the current set of RC applications. This demonstrates that instead of having to change parameters in a system for it to exhibit different behavior, the various desired modes of operation can coexist in the state space of a multifunctional RC where an appropriate controller could in theory be designed to switch between attractors.[34] This draws further parallels to biological neural networks, where such a control mechanism is comparable to neuromodulators,[41] a hierarchical system of neurons that some believe to be involved in instigating the switching of activity patterns in multifunctional neural networks. Additionally, we have illustrated that these input sources are not limited to one system. We show this in Fig. 8(b) where the RC was successfully trained to permit the coexistence of the Lorenz butterfly attractor, $\mathcal{L}$, and the chaotic attractor $\mathcal{A}_2$ generated from Eq. (8). Naturally, the question emerges as to the amount of attractors that can be successfully trained to coexist in the RC prediction state space, and we leave this for future work.

The results of this paper emanate from employing the dyadic "two-way street" approach of framing neurological features in the context of dynamical systems. Moreover, this work indicates that if other hypotheses or known facets of the brain can be articulated in this manner, then there lies the potential to portray it artificially.

ACKNOWLEDGMENTS

This work was funded by the Irish Research Council Enterprise Partnership Scheme (Grant No. EPSPG/2017/301). A.F. would like to thank Sebastian Wieczorek for introducing him to Reservoir Computing and Paul O'Keeffe and Christopher O'Connor for their helpful conversations.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] P. A. Getting, "Emerging principles governing the operation of neural networks," Annu. Rev. Neurosci. **12**, 185–204 (1989).
[2] P. S. Dickinson, "Interactions among neural networks for behavior," Curr. Opin. Neurobiol. **5**, 792–798 (1995).
[3] E. Marder and R. L. Calabrese, "Principles of rhythmic motor pattern generation," Physiol. Rev. **76**, 687–717 (1996).
[4] K. L. Briggman and W. B. Kristan, "Imaging dedicated and multifunctional neural circuits generating distinct behaviors," J. Neurosci. **26**, 10925–10933 (2006).
[5] S. Lieske, M. Thoby-Brisson, P. Telgkamp, and J. Ramirez, "Reconfiguration of the neural network controlling multiple breathing patterns: Eupnea, sighs and gasps," Nat. Neurosci. **3**, 600 (2000).
[6] K. L. Briggman and W. Kristan, Jr., "Multifunctional pattern-generating circuits," Annu. Rev. Neurosci. **31**, 271–294 (2008).
[7] A. N. Pisarchik and U. Feudel, "Control of multistability," Phys. Rep. **540**, 167–218 (2014).
[8] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks—With an erratum note," German National Research Center for Information Technology, GMD Technical Report, Vol. 148, 2001.

[9]W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," Neural Comput. **14**, 2531–2560 (2002).
[10]D. Verstraeten, B. Schrauwen, M. d'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," Neural Netw. **20**, 391–403 (2007).
[11]H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," Science **304**, 78–80 (2004).
[12]A. Jalalvand, G. Van Wallendael, and R. Van de Walle, "Real-time reservoir computing network-based systems for detection tasks on visual contents," in *2015 7th International Conference on Computational Intelligence, Communication Systems and Networks* (IEEE, 2015), pp. 146–151.
[13]P. Buteneers, D. Verstraeten, B. Van Nieuwenhuyse, D. Stroobandt, R. Raedt, K. Vonck, P. Boon, and B. Schrauwen, "Real-time detection of epileptic seizures in animal models using reservoir computing," Epilepsy Res. **103**, 124–134 (2013).
[14]Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, "Reservoir observers: Model-free inference of unmeasured variables in chaotic systems," Chaos **27**, 041102 (2017).
[15]J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, and E. Ott, "Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data," Chaos **27**, 121102 (2017).
[16]A. Banerjee, J. Pathak, R. Roy, J. G. Restrepo, and E. Ott, "Using machine learning to assess short term causal dependence and infer network links," Chaos **29**, 121104 (2019).
[17]Z. Lu, B. R. Hunt, and E. Ott, "Attractor reconstruction by machine learning," Chaos **28**, 061104 (2018).
[18]E. N. Lorenz, "Deterministic nonperiodic flow," J. Atmos. Sci. **20**, 130–141 (1963).
[19]L. Larger, M. C. Soriano, D. Brunner, L. Appeltant, J. M. Gutiérrez, L. Pesquera, C. R. Mirasso, and I. Fischer, "Photonic information processing beyond Turing: An optoelectronic implementation of reservoir computing," Opt. Express **20**, 3241–3249 (2012).
[20]K. Nakajima, H. Hauser, T. Li, and R. Pfeifer, "Information processing via physical soft body," Sci. Rep. **5**, 10487 (2015).
[21]G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, "Recent advances in physical reservoir computing: A review," Neural Netw. **115**, 100 (2019).
[22]M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," Comput. Sci. Rev. **3**, 127–149 (2009).
[23]L. A. Thiede and U. Parlitz, "Gradient based hyperparameter optimization in echo state networks," Neural Netw. **115**, 23–29 (2019).
[24]J. Yperman and T. Becker, "Bayesian optimization of hyper-parameters in reservoir computing," arXiv:1611.05193 (2016).
[25]S. Krishnagopal, M. Girvan, E. Ott, and B. R. Hunt, "Separation of chaotic signals by reservoir computing," Chaos **30**, 023123 (2020).
[26]D. M. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control," Neural Netw. **11**, 1317–1329 (1998).
[27]J. Tani, M. Ito, and Y. Sugita, "Self-organization of distributedly represented multiple behavior schemata in a mirror system: Reviews of robot experiments using RNNPB," Neural Netw. **17**, 1273–1289 (2004).
[28]M. I. Rabinovich, P. Varona, A. I. Selverston, and H. D. Abarbanel, "Dynamical principles in neuroscience," Rev. Mod. Phys. **78**, 1213 (2006).
[29]G. J. Mpitsos and C. S. Cohan, "Convergence in a distributed nervous system: Parallel processing and self-organization," J. Neurobiol. **17**, 517–545 (1986).
[30]I. R. Popescu and W. N. Frost, "Highly dissimilar behaviors mediated by a multifunctional network in the marine mollusk *Tritonia diomedea*," J. Neurosci. **22**, 1985–1993 (2002).
[31]H. Lu, "Chaotic attractors in delayed neural networks," Phys. Lett. A **298**, 109–116 (2002).
[32]B. Bao, H. Qian, J. Wang, Q. Xu, M. Chen, H. Wu, and Y. Yu, "Numerical analyses and experimental validations of coexisting multiple attractors in Hopfield neural network," Nonlinear Dyn. **90**, 2359–2369 (2017).
[33]Z.-H. Guan, Q. Lai, M. Chi, X.-M. Cheng, and F. Liu, "Analysis of a new three-dimensional system with multiple chaotic attractors," Nonlinear Dyn. **75**, 331–343 (2014).
[34]H. Richter, "Controlling chaotic systems with multiple strange attractors," Phys. Lett. A **300**, 182–188 (2002).
[35]Y. A. Kuznetsov, *Elements of Applied Bifurcation Theory* (Springer Science & Business Media, 2013), Vol. 112.
[36]J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields* (Springer Science & Business Media, 2013), Vol. 42.
[37]E. J. Doedel, T. F. Fairgrieve, B. Sandstede, A. R. Champneys, Y. A. Kuznetsov, and X. Wang, "Auto-07p: Continuation and bifurcation software for ordinary differential equations," Technical Report, 2007.
[38]P. Tass, M. Rosenblum, J. Weule, J. Kurths, A. Pikovsky, J. Volkmann, A. Schnitzler, and H.-J. Freund, "Detection of n: M phase locking from noisy data: Application to magnetoencephalography," Phys. Rev. Lett. **81**, 3291 (1998).
[39]M. Rosenblum and A. Pikovsky, "Delayed feedback control of collective synchrony: An approach to suppression of pathological brain rhythms," Phys. Rev. E **70**, 041904 (2004).
[40]P. A. Tass, C. Hauptmann, and O. V. Popovych, "Development of therapeutic brain stimulation techniques with methods from nonlinear dynamics and statistical physics," Int. J. Bifurcat. Chaos **16**, 1889–1911 (2006).
[41]R. M. Harris-Warrick and E. Marder, "Modulation of neural networks for behavior," Annu. Rev. Neurosci. **14**, 39–57 (1991).