Reservoir computing for spatiotemporal signal classification without trained output weights

Ashley Prater

Abstract

Reservoir computing is a recently introduced machine learning paradigm that has been shown to be well-suited for the processing of spatiotemporal data. Rather than training the network node connections and weights via backpropagation in traditional recurrent neural networks, reservoirs instead have fixed connections and weights among the ‘hidden layer’ nodes, and traditionally only the weights to the output layer of neurons are trained using linear regression. We claim that for signal classification tasks, one may forgo the weight training step entirely and instead use a simple supervised clustering method. The proposed method is analyzed theoretically and explored through numerical experiments on real-world data. The examples demonstrate that the proposed clustering method outperforms the traditional trained output weight approach in terms of speed, accuracy, and sensitivity to reservoir parameters.

Introduction

Reservoir computing is a recently developed bio-inspired machine learning paradigm for the processing of spatiotemporal data [1, 2]. In the language of neural networks, a reservoir is the collection of hidden layer nodes with nonlinear recurrent dynamics, where the nodes are sparsely connected with fixed weights that are not trained to fit specific data. Because the weights are fixed, using a reservoir requires only a simple initialization step, as opposed to more traditional recurrent neural networks whose weights and connections must be learned in a tedious backpropagation training step [3]. Strengths of the reservoir design include the ease of initialization, along with a reservoir’s ability to promptly adapt to new data and applications.

Reservoirs, like all recurrent neural networks, are based on the premise that the state of the reservoir at a particular time should depend on the current value of the input signal, along with recent inputs and reservoir states. To be an effective method for computation, a reservoir should map input data into a sufficiently high-dimensional space. It is desirable for a reservoir to operate ‘at the edge of chaos’ [4], so dissimilar inputs are sufficiently separated in the reservoir node states, yet inputs with only small perturbation-like differences do not stray too far apart. Reservoir dynamics demonstrate long short-term memory [5], so any individual point-wise errors in a signal will not corrupt the entire reservoir response.

Two reservoir variants that have emerged in literature are echo state networks (ESNs) and time-delay reservoirs (TDRs). An ESN uses randomly, yet sparsely, connected nodes with randomly assigned fixed weights [1, 2, 6]. A TDR uses a cyclic topology, where each node is connected to exactly two other nodes with fixed, non-random weights [7, 8, 9].

The output layer of both ESN and TDR-type reservoirs almost always use linear output weights, trained on a labeled dataset using least squares or ridge regression[1, 10, 11]. Although effective, this method is computationally expensive to apply to test data, since it requires many matrix-vector multiplications and comparisons. The cost is so high that it is difficult to use this method in real-time to classify incoming signals.

In this research, a simple supervised clustering method is proposed for use in classification tasks using ESNs and TDRs. The method used is based upon comparing the reservoir response of a signal against the principal components of the reservoir states for classes of labeled training data. The clustering method is shown to have lower computational complexity than using trained output weights to classify new input
signals. We present a rigorous analysis of the clustering method, including two theorems characterizing the upper bound of the difference in reservoir responses for two input signals, with the upper bound in terms of the input signals, the reservoir type, and the user-generated parameters. Moreover, we explore the difference in performance of the two methods through numerical simulations performed using a real-world dataset for both ESNs and TDRs for various reservoir parameters. In every simulation, the clustering approach outperforms the trained output weights in terms of both accuracy and CPU time required to classify test signals.

The following notation is used in this work. For a collection of signals \( \{u\} \), the \( j^{th} \) element in the collection is denoted by \( u^{(j)} \). Training sets are partitioned into \( K \) classes. Let \( \mathcal{C}_k \) be the collection of indices of signals in the \( k^{th} \) class. That is, \( u^{(j)} \) is in the \( k^{th} \) class iff \( j \in \mathcal{C}_k \). For a vector \( v \), the \( \ell_2 \) norm is given by \( \|v\|_2 = (\sum_j v_j^2)^{1/2} \), and the \( \ell_1 \) seminorm is given by \( |v|_1 = \sum_j |v_j| \). For a matrix \( A \), \( \rho(A) \) is the spectral radius, i.e. the largest absolute value of an eigenvalue of \( A \). We use \( \mathcal{O}(\cdot) \) with the standard ‘big O’ meaning, that \( f(x) = \mathcal{O}(g(x)) \) if there exists \( M > 0 \) and \( x_0 \in \mathbb{R} \) such that \( |f(x)| \leq M|g(x)| \) for all \( x \geq x_0 \).

Reservoir Computing Models for Classification

Suppose \( u(t) \) is an input signal at time \( t \), possibly after the application of a multiplexing mask. The values of the reservoir nodes at time \( t \) are called the reservoir states and are denoted by the vector \( x(t) \). The \( n^{th} \) entry of this vector, \( x_n(t) \) denotes the state of the \( n^{th} \) reservoir node at time \( t \). The dynamics of the ESN and TDR architectures are described by the following models:

\[
\text{ESN: } x(t) = f(W_{\text{in}}u(t) + W_{\text{res}}x(t-1)). \tag{1}
\]

\[
\text{TDR: } x_n(t) = \begin{cases} f(\alpha u(t) + \beta x_{N-1}(t-1)), & \text{if } n = 0, \\ x_{N-1}(t-1), & \text{if } n \in \{1, 2, \ldots, N-1\}. \end{cases} \tag{2}
\]

For ease of notation, suppose each reservoir type has \( N \) nodes. In the ESN topology, the vector \( W_{\text{in}} \in \mathbb{R}^N \) weights the input signal feeding into the nodes, while the matrix \( W_{\text{res}} \in \mathbb{R}^{N \times N} \) determines the fixed connections and weights among the nodes. That is, node \( m \) feeds into node \( n \) weighted by the value \( W_{n,m} \) in the ESN model. The TDR has \( N-1 \) virtual nodes and one physical node, labeled as \( x_0 \). The parameter \( \alpha \) is the input gain, and \( \beta \) is the attenuation value. Notice in the TDR the node values are simply passed along the reservoir unchanged except at the physical node \( x_0 \). Models of the ESN and TDR reservoirs are shown in Figure 1.

The function \( f \) in (1) and (2) is a nonlinear activation function. Typical choices for \( f \) include sinusoidal, logistic, sigmoidal, and piecewise linear functions.

![Figure 1](image)

**Figure 1:** Representations of two architectural variants of reservoirs, the echo state network (left) and time delay reservoir (right).

Output Weights Trained using Regularized Least Squares

Classically, signals are classified in a reservoir computer using a linear combination of the reservoir states, determined by a collection of trained output weights [1, 10, 11, 12]. To train the output weights, input signals
of all classes of the training set are processed, storing all of the reservoir states in a matrix $X_{\text{train}} \in \mathbb{R}^{TS \times N}$, where $T$ is the number of times the reservoir state is collected for each input signal, $S$ is the total number of signals in the training set, and $N$ is the number of nodes in the reservoir. The output weights are chosen to map the reservoir states onto given target functions. Typically a unique target vector $y_k \in \mathbb{R}^T$ is chosen for each class $k$. All of the individual target vectors are combined into a block diagonal target matrix $Y_{\text{target}}$, with diagonal blocks arranged to correspond to the class of the input signals. The most popular method to compute the output weights $W_{\text{out}}$ is to use regularized least squares. That is, $W_{\text{out}}$ is found that minimizes the equation

$$||Y_{\text{target}} - X_{\text{train}} W_{\text{out}}||^2_2 + \lambda ||W_{\text{out}}||^2_2.$$  

The regularization constant helps to minimizes overfitting of the data. If $\lambda$ is very close to zero, then the weights will fix $X_{\text{train}}$ to the targets $Y_{\text{target}}$ very well, but the coefficients in $W_{\text{out}}$ may be large and cause sensitivity to noise. Instead, a moderately small value of $\lambda$ is chosen so the data fits well while dampening the weights. Since a single target vector may be insufficient to describe an entire class of input signals, it is determined to belong to the $k^{th}$ class if

$$\left\| \tilde{y}^{(j)} - X_{\text{train}} W_{\text{out}} \right\|_2^2,$$  

is minimized for $j \in C_k$.

For large training sets, it may be very time consuming to compute the output weights. However, the training step is computed offline, and once finished the weights can be stored in memory. Unfortunately, assigning a single unclassified input signal to a class using these stored weights is also expensive with a computational complexity of $O(NTS)$. For moderately sized reservoirs and signals, this method may be prohibitively expensive for real-time classification. As shown in the following section, one can reduce the computational complexity by forgoing the trained weights entirely and performing classification using a supervised clustering method on the reservoir states.

### Classification via Clustering with Principal Components

The underlying idea for the training method (3) is that similar inputs to the reservoir produce similar outputs, even after the non-linear high-dimensional processing is applied. Under this assumption, it is feasible that one could classify data using a clustering method. Therefore, we propose the following method using the principal components of reservoir responses to perform classification.

For each input $u^{(j)}$ in the $k^{th}$ class of the training set, compute the vector $v^{(j)}$, whose entries are the norms of the reservoir node states:

$$v^{(j)}(t) = \left\| x^{(j)}(t) \right\|_2,$$  

where $x^{(j)}(t) \in \mathbb{R}^N$ are the values of the reservoir nodes at time $t$ for input $u^{(j)}$.

Let $u$ be an unclassified input vector with reservoir node states $x$ and output vector $v$ computed as in (4). Then $u$ is determined to be in class $k$ if $v$ is best described by the first few principal components of the collection $D_k = \{v^{(j)} : j \in C_k\}$. More specifically, for each class $k$ compute the matrix $U_k \in \mathbb{R}^{T \times R}$ whose columns are the first $R$ principal components of $D_k$. Say $u$ is in class $k$ if

$$k = \arg\min_{\ell \in \{1, 2, \ldots, K\}} \left\| (I - U_k U_k^*) v \right\|_2^2.$$  

The method in Equation (5) requires $O(KT^2)$ multiplications to assign a single input vector of length $T$ to one of $K$ classes. Despite the quadratic factor of $T$, this method tends to be significantly less expensive in practice than training the output weights as in Equation (3) since, in general, $KT \ll NS$. 

3
Analysis of Reservoir Behavior

The clustering method (5) will be more accurate if small variations in the input signals lead to bounded differences in reservoir states, while large discrepancies in inputs are mapped farther apart. To confidently use (5) we must characterize reservoir behavior for similar inputs.

Several studies of reservoir performance based on the type of reservoir architecture, chosen parameters, as well as the characteristics of the input data have been performed, with evidence that some combinations of the aforementioned factors can seriously degrade performance [2, 7, 10, 13]. However, the metrics used in the reservoir computing literature tend to be experimentally investigated. To explore how well the reservoir response separates classes, the separation ratio [6, 14], point-wise separation [2, 15], and class separation [16] have been used. These all measure how well a reservoir can separate inputs from distinct classes, by having distances between disparate classes large while keeping similar inputs close. Similarly, to measure how effectively a reservoir can process a particular dataset, researchers use the universal approximation property [2] kernel quality [16, 17, 18], reservoir capacity [19], and the Echo State Property [1]. These measures and properties concern the representation of inputs within the reservoir response and the reconstructability of an input signal from reservoir states. For robustness to noise, generalization rank [18] or the Lyapunov coefficient [14, 16, 17, 20, 21, 22] are considered.

Although the reservoir dynamics (1) and (2) have simple descriptions, rigorous treatment of their behavior have proven difficult, with few results so far. In Proposition 3 of [1], the distance between two reservoir states at a given time is bounded in terms of the reservoir states at the previous timestep and the spectral radius of the reservoir weights. Although mathematically proven, this Proposition covers only randomly connected ESNs incrementing one timestep with activation functions of the form $f$ and $\delta$. Finally, let $[\cdot]_n$ denote a vector whose entries run over the range of the variable $n$.

Theorem 1. Suppose the reservoir node states are determined using the ESN dynamics from Equation (1). If $\rho(W_{\text{res}})$ is the spectral radius of $W_{\text{res}}$, then the distance between the reservoir nodes at time $t$ corresponding to two input signals $u(i)$ and $u(j)$ satisfies

$$\varepsilon_{i,j,t} \leq L\bar{\varepsilon}_{i,j} ||W_{\text{in}}|| \frac{1 - (L\rho(W_{\text{res}}))^{t+1}}{1 - L\rho(W_{\text{res}})}.$$  

Proof. By Equation (1) and the Lipschitz continuity of $f$,

$$\varepsilon_{i,j,t} = \left\| x^{(i)}(t) - x^{(j)}(t) \right\|$$

$$= \left\| f \left( W_{\text{in}} u^{(i)}(t) + W_{\text{res}} x^{(i)}(t-1) \right) - f \left( W_{\text{in}} u^{(j)}(t) + W_{\text{res}} x^{(j)}(t-1) \right) \right\|$$

$$\leq L \left\| W_{\text{in}} [u^{(i)}(t) - u^{(j)}(t)] + W_{\text{res}} [x^{(i)}(t-1) - x^{(j)}(t-1)] \right\|$$

$$\leq L \left\| W_{\text{in}} \delta_{i,j,t} + L\rho(W_{\text{res}}) \varepsilon_{i,j,t-1} \right\|.$$  

Since $\varepsilon_{i,j,t-1} = 0$, it follows by induction that

$$\varepsilon_{i,j,t} \leq L \left\| W_{\text{in}} \sum_{r=0}^{t} (L\rho(W_{\text{res}}))^r \delta_{i,j,t-r} \right\| \leq L\bar{\varepsilon}_{i,j} \left\| W_{\text{in}} \right\| \frac{1 - (L\rho(W_{\text{res}}))^{t+1}}{1 - L\rho(W_{\text{res}})}.$$  

$\square$
Theorem 2. Suppose the reservoir node states are determined using the TDR dynamics from Equation (2). Then the distance between the reservoir nodes corresponding to two input signals $u^{(i)}$ and $u^{(j)}$ satisfies

$$\varepsilon_{i,j,t} \leq \alpha \tilde{d}_{i,j} L \sqrt{N} \frac{1 - (\beta L)^{\lfloor t/N \rfloor + 1}}{1 - \beta L}.$$ 

Proof. By Equation (2) and the Lipschitz continuity of $f$,

$$\varepsilon_{i,j,t} = \left\| x_n^{(i)}(t) - x_n^{(j)}(t) \right\| = \left\| x_0^{(i)}(t - n) - x_0^{(j)}(t - n) \right\| \leq \alpha L \left\| u^{(i)}(t - n) - u^{(j)}(t - n) \right\| + \beta L \left\| x^{(i)}_{N-1}(t - n) - x^{(j)}_{N-1}(t - n) \right\| \leq \alpha L \tilde{d}_{i,j} \sqrt{N} + \beta L \left\| x_n(t - N) \right\| = \alpha L \tilde{d}_{i,j} + \beta L \varepsilon_{i,j,t-N}.$$ 

Let $r \in \{0, 1, \ldots, N - 1\}$ be the remainder when $t$ is divided by $N$. By induction on the inequality above,

$$\varepsilon_{i,j,t} \leq (\beta L)^{\lfloor t/N \rfloor} \varepsilon_{i,j,r} + \alpha \tilde{d}_{i,j} L \sum_{k=0}^{\lfloor t/N \rfloor - 1} (\beta L)^k.$$ 

Since $r < N$, the $n^{th}$ reservoir node at time $r$ can be characterized by

$$x_n(r) = \begin{cases} f(\alpha u(r - n)), & \text{if } n \leq r \\ 0, & \text{if } n > r \end{cases},$$

yielding $\varepsilon_{i,j,r} \leq \alpha \tilde{d}_{i,j} L \sqrt{r + 1} \leq \alpha \tilde{d}_{i,j} L \sqrt{N}$. Therefore

$$\varepsilon_{i,j,t} \leq \alpha \tilde{d}_{i,j} L \sqrt{N} \sum_{k=0}^{\lfloor t/N \rfloor} (\beta L)^k = \alpha \tilde{d}_{i,j} L \sqrt{N} \frac{1 - (\beta L)^{\lfloor t/N \rfloor + 1}}{1 - \beta L}.$$ 

Theorems 1 and 2 show that for input signals with small pointwise discrepancies and well-chosen reservoir parameters, their associated reservoir state norms cluster well. However, the Theorems do not guarantee that very distinct inputs are mapped to dissimilar reservoir node state norms. For this, we turn to the separation ratio, introduced in [6] and further explored in [14]. For completeness, we include it here, modified for the classification scheme (5).

Let

$$M_k(t) := \frac{1}{|C_k|} \sum_{j \in C_k} \left\| x^{(j)}(t) \right\|,$$

the mean of the reservoir state norms at for all signals in the $k^{th}$ training class at time $t$. The inter-class distance $d$ and intra-class variance $w$ at time $t$ are

$$d(t) = \frac{1}{K^2} \sum_{k=1}^{K} \sum_{l=1}^{K} |M_k(t) - M_l(t)|,$$

and

$$w(t) = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{j \in C_k} \left| M_k(t) - \left\| x^{(j)}(t) \right\| \right|.$$

Then the separation ratio at time $t$ is defined as

$$\text{Sep}(t) = \frac{d(t)}{1 + w(t)}.$$ 

The larger Sep$(t)$ is, the better the separation among the classes at time $t$. 

Numerical Example

Handwritten digits are classified using the clustering method proposed in Equation (5) and by linear readouts using weights trained as in Equation (3). The data used are from the United States Postal Service (USPS) database, obtained from [24]. A sample of these images is shown in Figure 2. Each image in the dataset is a $16 \times 16$ 8-bit grayscale image, reshaped as a 256 length column vector. The data are split in ten classes of 1100 images each, representing the digits 0 through 9. For each simulation presented below, 400 images in each class are randomly selected to form the training set, while the remaining 700 images are used as the test set. Although the nearby pixel behavior is not preserved in the $x$-direction by transforming each image into a column vector, the correlations are still present in the reservoir response due to the long short term memory property.

The classification accuracy and CPU time required to classify all images in the test dataset, using both (5) and (3), for both the ESN and TDR reservoir topologies for several parameter choices are presented below. We also give the separation metric (6), and verify that the inequalities in the conclusions of Theorems 1 and 2 are satisfied. The clustering approach in (5) is applied to the reservoir states, but the raw input signals could also be classified according to similarity with principal components, without processing the data in the reservoir. The classification accuracy of this approach is also investigated.

All numerical experiments are implemented in MATLAB R2015b on a workstation with an Intel i7 CPU (2.40 GHz) and 16GB RAM.

Experiment Setup

The reservoirs are set up with $N$ nodes, selecting $N$ from the set $\{25, 50, 100, 400\}$. Each image vector is multiplexed with a mask of length $N-1$, randomly taking values from the set $\{\pm 1\}$, creating input vectors $u$ of length $256(N-1)$. The parameter $\alpha$ varies over the set $\{0.1, 0.2, \ldots, 0.9\}$. The nonlinear activation function is chosen to be $f(x) = \sin(x)$ for all simulations.

When a reservoir with the ESN topology is used, the input weights are chosen as $W_{in} = [\alpha \ 0 \ \cdots \ 0]^T$, and the internal weight matrix $W_{res}$ is randomly filled to $20\%$ density and scaled so that $\rho(W_{res}) = \beta$. For both ESN and TDR topologies, we use $\alpha + \beta = 1$ in order to preserve the dynamical range of the data. The reservoir states are not sampled at every time step; instead we record the reservoir state values for $t = r(N-1) + 1$, with $r \in \mathbb{N}$.

For each simulation, the training dataset of size 400 is randomly chosen from the complete collection of images, and the remaining 700 images form the test dataset. However, the same selection is used for each pair $(N, \alpha)$, so the competing approaches can be fairly compared.

Results

The results of these simulations are presented in Figures 3-5.

Figure 3 displays the average value of the separation metric $\text{Sep}(t)$ from Equation (6) over all $t$ for each simulation. Notice that the separation quality of ESN-style reservoirs is modest, with a maximum value of 2.25. Moreover, the separation quality of ESNs depend on the parameter $\alpha$, but does not seem to depend...
on the reservoir size $N$. On the other hand, the separation quality for TDR-style reservoirs is much larger, and depends strongly on both $\alpha$ and $N$, with larger reservoirs separating the classes better.

Figure 4 displays the classification accuracy and the time required to classify all 7000 images in the test dataset for each reservoir type and combination of parameters. The clustering approach outperforms the linear output with trained weights in all simulations in terms of both accuracy and CPU time. The accuracy results are shown in the top two plots of Figure 4. For both reservoir types and classification strategies, the accuracy does not seem to depend on the reservoir size. The accuracy does depend on the parameter $\alpha$, especially for TDR-type reservoirs. The time results are shown in the bottom two plots of Figure 4. The analysis of the two classification methods determined that the computational complexity of using trained output weights does depend on the $N$, but the clustering approach does not, which agrees with the results shown. Overall, the clustering approach used in conjunction with either reservoir type is at least an order of magnitude faster than using trained output weights for classification. Additionally, the clustering approach with the reservoir outperforms clustering the raw input data without the reservoir. Over 100 trials, the average accuracy of the clustering method applied to the raw input data is 95.27%. This suggests that the clustering method is well-suited to this problem, but processing the data in a reservoir improves accuracy for most parameter choices since the reservoir preserves the spatial correlations well.

Figure 5 displays values of the inequalities (1) and (2) presented in Theorems 1 and 2, measuring the discrepancy of reservoir activations at time $t$ for similar inputs. The two input signals were randomly selected from the class of ‘3s’. The values shown in the Figure are

$$\varepsilon_{i,j,t}/\left( L\delta_{i,j}\|W_{\text{in}}\|^2 \frac{1 - (L\rho(W_{\text{res}}))^t + 1}{1 - L\rho(W_{\text{res}})} \right)$$

in the left image, and

$$\varepsilon_{i,j,t}/\left( \alpha\delta_{i,j}L\sqrt{N} \frac{1 - (\beta L)^{\lfloor t/N \rfloor} + 1}{1 - \beta L} \right)$$

in the right image, both plotted against $t$. The inequalities in the theorems are clearly satisfied, however the upper limits could be further refined in future research.

**Conclusion**

The numerical experiments demonstrate that the proposed clustering method (5) outperforms the standard trained linear output weights approach (3) for classification tasks in terms of time and accuracy, for both ESNs and TDRs, for all reservoir parameters investigated. The accuracy achieved by the clustering approach does not strongly depend on the reservoir type, size, or parameter selection, unlike the results achieved by trained output weights. Moreover, the clustering approach has the potential to be several orders of magnitude faster, allowing its use in real-time systems. The clustering method described in this paper is shown to be robust, and is a promising approach for use in reservoir computers for classification tasks.

![Figure 3: The average separation quality achieved by the reservoir for each simulation, for ESN-style reservoirs (left) and TDR-style reservoirs (right), for several values of $\alpha$ and $N$.](image-url)
Figure 4: A comparison of performance of the proposed clustering method and the traditional method of training output weights. All 7000 images of handwritten digits in the test dataset were classified using both approaches, for the ESN and the TDR reservoir topologies, for $\alpha = 0.1, \ldots, 0.9$ and reservoir sizes $N = 25, 50, 100$ and 400. The top two figures present the classification accuracy, and the bottom two figures present the CPU time required to perform the classification step for each parameter combination. The symbols ‘$\circ$’ and ‘$\ast$’ denote results for the clustering and the trained output weight approaches, respectively. The different lines within each class represent the different reservoir sizes.
Figure 5: Inequalities from Theorems 1 and 2 for randomly selected $i$ and $j$, plotted against values of $t$, with $N = 100$ and $\alpha = 0.5$. Here, the right hand side of Equations (1) and (2) are divided out, so the plotted values should be bounded above by 1.

References

[1] Jaeger, H. The “echo state” approach to analysing and training recurrent neural networks - with an Erratum note. Fraunhofer Institute for Autonomous Intelligent Systems, Technical report: GMD Report 148 (2011).

[2] Maass, W., Natschläger, H. & Markram, H. Real-time computing without stable states: a new framework for neural computational based on perturbations. *Neural Comput.* 14, 2531–2560 (2002).

[3] Werbos, P. Backpropagation through time: what it does and how to do it. *Proc. IEEE* 89, 1550–1560 (1990).

[4] Bertschinger, N. & Natschläger, H. Real-time computation at the edge of chaos in recurrent neural networks. *Neural Comput.* 6, 1413–1436 (2004).

[5] Jaeger, H. Long short-term memory in echo state networks: Details of a simulation study. *Jacobs University Bremen Technical Report* 27 (2012) Available at: http://minds.jacobs-university.de/sites/default/files/uploads/papers/2478_Jaeger12.pdf

[6] Goodman, E. & Ventura, D. Spatiotemporal pattern recognition via liquid state machines. *2006 International Joint Conference on Neural Networks (IJCNN)*, 3848-3853 (2006) doi: 10.1109/ijcnn.2006.246880

[7] Grigoryeva, L., Henriques, J., Larger, L. & Ortega, J. Optimal nonlinear information processing capacity in delay-based reservoir computers. *Sci. Rep.* 5, 12858; doi: 10.1038/srep12858 (2015)

[8] Ortín, S. et al. A unified framework for reservoir computing and extreme learning machines based on a single time-delayed neuron. *Sci. Rep.* 5, 10.1038/srep14945 (2015).

[9] Paquot, Y. et al. Optoelectronic reservoir computing. *Sci. Rep.* 2 10.1038/srep00287 (2012).

[10] Goudarzi, A., Banda, P., Lakin, M., Teuscher, C. & Stefanovic, D. A comparative study of reservoir computing for temporal signal processing. *University of New Mexico Technical Report* 11pp, arXiv:1401.2224 (2014).

[11] Lukoševičius, M. & Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comp. Sci. Rev.* 3, 127–149 (2009).
Author contributions statement

A. P. conceived and performed all work.

Additional information

The author declares no competing financial interests.

Acknowledgement of Disclaimer: Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the view of the United States Air Force.