Hindawi
Complexity
Volume 2018, Article ID 6953836, 15 pages
https://doi.org/10.1155/2018/6953836

Research Article

Similarity Learning and Generalization with Limited Data: A Reservoir Computing Approach

Sanjukta Krishnagopal,[1] **Yiannis Aloimonos**,[2] **and Michelle Girvan**[1,3,4]

[1]*Department of Physics, University of Maryland, College Park, MD 20740, USA*
[2]*Department of Computer Science, University of Maryland, College Park, MD 20740, USA*
[3]*Santa Fe Institute, New Mexico 87501, USA*
[4]*London Mathematical Laboratory, London WC2N 6DF, UK*

Correspondence should be addressed to Sanjukta Krishnagopal; sanjukta@umd.edu

Received 15 April 2018; Revised 20 September 2018; Accepted 3 October 2018; Published 1 November 2018

Academic Editor: Danilo Comminiello

We investigate the ways in which a machine learning architecture known as Reservoir Computing learns concepts such as "similar" and "different" and other relationships between image pairs and generalizes these concepts to previously unseen classes of data. We present two Reservoir Computing architectures, which loosely resemble neural dynamics, and show that a Reservoir Computer (RC) trained to identify relationships between image pairs drawn from a subset of training classes *generalizes* the learned relationships to substantially different classes *unseen* during training. We demonstrate our results on the simple MNIST handwritten digit database as well as a database of depth maps of visual scenes in videos taken from a moving camera. We consider image pair relationships such as images from the same class; images from the same class with one image superposed with noise, rotated 90°, blurred, or scaled; images from different classes. We observe that the reservoir acts as a nonlinear filter projecting the input into a higher dimensional space in which the relationships are separable; i.e., the reservoir system state trajectories display different dynamical patterns that reflect the corresponding input pair relationships. Thus, as opposed to training in the entire high-dimensional reservoir space, the RC only needs to learns characteristic features of these dynamical patterns, allowing it to perform well with very few training examples compared with conventional machine learning feed-forward techniques such as deep learning. In generalization tasks, we observe that RCs perform significantly better than state-of-the-art, feed-forward, pair-based architectures such as convolutional and deep Siamese Neural Networks (SNNs). We also show that RCs can not only generalize relationships, but also generalize combinations of relationships, providing robust and effective image pair classification. Our work helps bridge the gap between explainable machine learning with small datasets and biologically inspired analogy-based learning, pointing to new directions in the investigation of learning processes.

1. Introduction

Different types of Artificial Neural Networks (ANNs) have been used in the areas of feature recognition and image classification. *Feed-forward* machine learning architectures such as convolutional neural networks (CNNs) [1], deep neural networks [2], and stacked autoencoders [3] and *recurrent* architectures such as Recurrent Neural Networks (RNNs) [4] and Long Short-Term Memories (LSTMs) [5] have been immensely successful for several tasks from speech recognition [6] to playing the game GO [2].

There have also been a number of rapid advances in other *recurrent* machine learning architectures such as Echo State Networks (ESNs) (originally proposed in the field of machine learning) [7] and Liquid State Machines (LSMs) (originally proposed in the field of computational neuroscience) [8], commonly falling under the term Reservoir Computing [9]. Compared with deep neural networks, Reservoir Computers (RCs) are a brain-inspired machine learning framework, and their inherent dynamics when trained on cognitive tasks have been shown to be useful in modeling local cortical dynamics in higher cognitive function [10].

The goal of this work is to demonstrate the unreasonable efficiency of Reservoir Computers (RCs) in learning the relationships between images with very little training data and consequently generalizing the learned relationships to classes of images not seen before. We recognize that other machine learning techniques such as deep learning [11] and CNNs have been proven to be extremely successful at image classification and have also been used for tasks involving learning concepts of similarity [12–14]; however, they generally require large training datasets and high computational resources. To our knowledge, similarity-based tasks have not been systematically investigated using RC architectures. However, RCs, because of their dynamical properties and simple training needs, may inherently be better suited for learning from a small training set and generalization of this learning [15]. While other recurrent architectures, like LSTMs and Gated Recurrent Units (GRUs), may also offer dynamical properties enabling generalization, due to their complex structure and training, they often require comparatively much larger datasets for training and hence are more computationally intensive.

RCs are dynamical systems that nonlinearly transform input data in a reproducible way in order to serve as a resource for information processing. They are appealing because of their dynamical properties as well as easy scalability, since only the output weights are trained, while the recurrent connections within the reservoir are fixed randomly. Applications of RCs include processing and prediction of many real world phenomena such as weather patterns, stock market fluctuations, self-driving cars, language interpretation, and robotic control, several of which are inherently nonlinear phenomenon. RCs are also appealing because of their biologically inspired underpinnings. Biological systems such as the visual cortex are known to have primarily (~70%) recurrent connections with less than 1 % of the connections being feed-forward [16]. RCs (or closely related architectures) provide insights into how biological brains can carry out accurate computations with an "inaccurate" and noisy physical substrate [17], for example, accurate timing of the way in which visual spatiotemporal information is super-imposed and processed in primary visual cortex [18]. Additionally, models of spontaneously active cortical circuits typically exhibit chaotic dynamics, as in RCs [19, 20].

In biological systems, a recurring method of learning is through analogies, using only a handful of examples [21]. For example, in [22], bees were trained to fly towards the image in an image pair that looked very similar to a previously displayed base image. On training bees to fly towards the visually similar image, the bees were presented with two scents, one very similar to and one different from a base scent. As a consequence of the visual training that induced preference to the very similar category, the bees flew towards the very similar scent. Recent work has also been done on the phenomenon of "peak shift", where animals not only respond to entrained stimuli, but respond even more strongly to similar ones that are farther away from nonrewarding stimuli [23]. In this way, biological systems have been found to translate learning of concepts of similarity across sensory inputs, suggesting that the brain has a common and fundamental mechanism that comprehends through analogies or through concepts of "similarity," allowing for generalization of the relationships to unseen classes of data. Compared with machine learning, humans learn much richer information using very few training examples. Moreover, humans learn more than how to do pattern or object recognition: they learn a concept, i.e., a model of the class that allows their acquired knowledge to be flexibly applied in new and unseen situations [24]. While many machine learning approaches can effectively classify images with human-like accuracy, these approaches often require large training datasets and consequently increasingly powerful GPUs.

Despite the fact that research in learning from very few images, e.g., one shot learning [25], etc., has gained momentum recently, integrating it with generalization of learning is a relatively unexplored area. One shot learning, which learns a class (e.g., sleeping cats) from one example, is distinctly different from the task of generalization to an entirely new class (e.g., recognizing sleeping dogs after having only been trained to recognize sleeping cats). In our framework, the RC not only requires very few training examples compared to techniques such as deep learning, but can also effectively use analogies to learn relationships, leading to easy generalization.

RCs are built on several prior successful approaches that emphasize the use of a dynamical system, e.g., with temporal reinforcement, for successful, neuroinspired learning. In the ground-breaking work of Hopfield in [26], the success of Recurrent Neural Networks (RNNs) depends on the existence of attractors. In training, the dynamical system of the RNN is left running until it ends up in one of its several attractors. Similarly, in [27], a unique conceptor is found for each input pattern in a driven RNN. However, training of RNNs is difficult due to training problems like exploding or vanishing gradient. RCs overcome this problem by training only the output weights. RCs offer a convenient solution to some the problems with RNNs, while offering many of the same advantages.

In this work, we explore two RC architectures that broadly resemble neural architecture (Section 2.1). We train the RCs on both the MNIST handwritten digit database (to demonstrate proof of concept) as well as depth maps of visual scenes from a moving camera, to study generalization of the learned relationships between pairs of images. The data and methods are outlined in Section 2. The methods include training the RC to identify relationships between image pairs drawn from a subset of handwritten digits (0–5) from the MNIST database and generalizing the learned relationships to images of handwritten digits (6–9) unseen during training. Additionally, using a database of depth maps of images taken from a moving camera, we train RCs to learn relationships such as "similar" (e.g., same scene, different camera perspectives) and "different" (different scenes) and investigate the system's ability to generalize its learning to visual scenes that are very different from those used in training. In Section 3.1, we present the performance of our RC architectures in generalization to unseen classes, showing

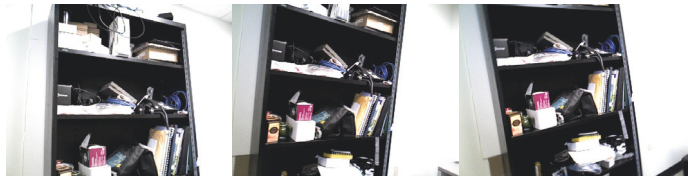


Figure 1: Examples of images taken from a moving camera from the same class. A pair of these would be classified under the category "similar".

successful generalization for both handwritten digits and depth maps.

We also compare, in Section 3.2, the RC performance for our generalization task to two pair-based, feed-forward approaches: a deep Siamese Neural Network (SNN) and a convolutional Siamese Neural Network (CSNN). Several recent studies have been very successful in using Siamese (pair-based) feed-forward networks for similarity-based tasks such as sketch-based image retrieval [28], gait recognition in humans [29], signature verification [30], verification and one/few shot learning on the Omniglot dataset [31, 32], etc. For our generalization task, we show that the reservoir performs significantly better than commonly used deep and convolutional Siamese Neural Networks, both for simpler MNIST images as well as for depth maps, highlighting the utility of the RC approach for generalization to unseen data classes using limited training data. We also show, in Section 3.3, that the reservoir is able to recognize not only the individual relationships it has been trained on but also combinations of them.

In order to explain the success of the reservoir in generalization, we look for recurring dynamical patterns the reservoir system state trajectories in Section 3.4. We find that the reservoir state trajectories in response to different types of input pairs effectively cluster, with different clusters corresponding to different relationships between the pair of input images. The reservoir can then be thought of as a nonlinear filter whose goal is to map the input into a high-enough dimensional space that the important features become nearly linearly separable. In addition, the dynamical properties of the reservoir allow for temporally encoded "memory". We speculate that this combination of effective nonlinear filtering and temporally encoded memory allows for generalization of the learned relationships to classes of image pairs seen and unseen by the reservoir using a small number of training image pairs.

2. Data and Methods

We use two datasets for our study: (1) the handwritten digit MNIST database that consists of 70000 images, each 28×28 pixels in size, of handwritten digits 0-9; and (2) depth maps from a moving camera from 6 different visual scenes recorded indoors in an office setting (refer data availability for access to dataset). Each visual scene has depth maps from at least 300 images, each compressed to 100×100 pixels in size, recorded as the camera is moved within a small distance (~30cm) and rotated within a small angle (~30°). A sample of three RBG images from one of the 6 classes is shown in Figure 1.

In our framework, images are always considered in pairs (image 1 and image 2). We study five relationships, noise, rotated, zoomed, blurred, and different. We are interested in exploring relationships between images through concepts of "similarity" and "difference". The relationships we consider are a natural extension of these concepts. Examples of the image pair relationships applied to the MNIST dataset are shown in Figure 2. We create the image pairs as follows.

Two different images from the same class, i.e., of the same digit, are taken directly from the MNIST database for cases 1–4. There may be significant variation between these images in spite of them being from the same class.

(1) Noise: one of the images in the pair (image 1) remains untransformed, whereas the other (image 2) is transformed by superimposing random noise with peak value given by 20 % of the peak value of image 1 (Figure 2(a)).

(2) Rotated: image 2 is 90◦ rotated (Figure 2(b)).

(3) Zoomed: image 2 is zoomed with a magnification of 2 (Figure 2(c)).

(4) Blurred: image 2 is blurred (Figure 2(d)) by convolving every pixel of the image by a 6 × 6 convolution matrix with all values 1/36.

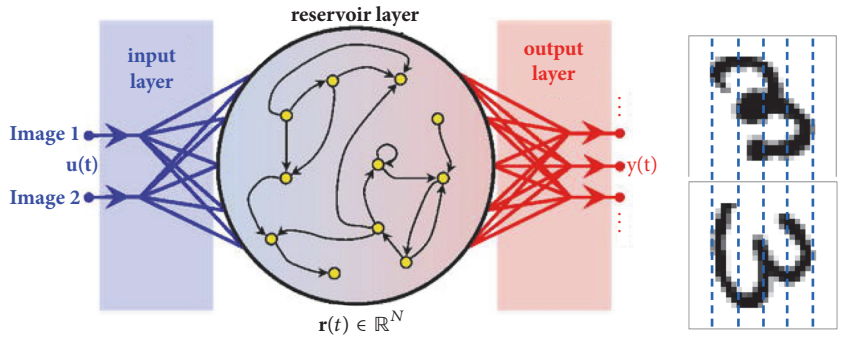
(5) Different: two different images from different classes (Figure 2(e)).

All pairs are characterized by the relationship between the image pair. For instance, we call a pair rotated if we start from two different handwritten images of the same digit and rotate the second image 90° with respect to the first. Since two different handwritten images of the same digit are used, the relationship between the image pair involves an initial nonlinear transformation in addition to the applied transformation.

2.1. Network Architecture. In this work we use the Echo State Network (ESN) class of RCs for training and classification. Our RCs are neural networks with two layers: a hidden layer of recurrently interconnected nonlinear nodes, driven both by inputs as well as through feedback from other nodes in the reservoir layer and an output or readout layer. Only the output weights of the reservoir are trained. RCs have been found to replicate attractors in dynamical systems [33, 34]. It works particularly well for analyzing time series input data due to its short-term memory [15] and high-dimensional encoding of the input [35, 36]. The input images are hence converted into a "time series" by feeding the reservoir a

(a) (b) (c) (d) (e)

Figure 2: Pairs of images that are representative of the transformations classified into five labels: (a) very similar, (b) rotated by 90°, (c) zoomed, (d) blurred, and (e) different.



(a) (b)

Figure 3: (a) Reservoir architecture with input state of the two images at time t denoted by $\vec{u}(t)$, reservoir state vector at a single time by $\vec{r}(t)$, and output state by $\vec{y}$. (b) shows one image pair from the rotated 90° category of the MNIST dataset split vertically and fed into the reservoir in columns of 1 pixel width, shown to be larger here for ease of visualization.

column of the input image at each time point (as in [37]). The method of "temporalization" of the input (row-wise, column-wise, etc.) simply changes the input representation and does not affect the analysis. While there is limited understanding of the actual processes through which the brain processes analogies, we explore two models that are inspired by cortical processing of relationships between inputs. There has been some evidence [38] of integrated processing, particularly in the visual cortex. To mimic an integrated processing system more closely, we study the single reservoir architecture (Figure 3(a)). However, there is also some evidence that analogy processing involves two steps: (1) the brain generated individual mental representations of the different inputs and (2) brain mapping based on structural similarity, or relationship, between them [39]. We create the dual reservoir architecture (Figure 4) in an attempt to mimic this parallel processing of signals followed by mapping based on the differences between the processed signal in the cortex. Since there is not a consensus in the neuroscience community about the details of cortical processing, we present both the single and dual reservoir architecture here.

2.1.1. Single Reservoir Architecture

Input Layer. As discussed above, in order to exploit the memory properties of RCs, the input is converted to a time series. We vertically concatenate the image pair to form the combined image. We then input the combined image, through the input weights matrix W^{in}, column by column (shown in Figure 3(b) for the MNIST database) into the reservoir, i.e., with the time axis to run across the columns of the combined image. While this "temporalization" may seem artificial, there's a unique reproducible reservoir state trajectory (the sequence of reservoir states) corresponding to each image, causing the results to be independent of order of temporalization, as long as all images are temporalized the same way. W^{in} is randomly chosen and scaled such that the inputs to the reservoir are between 0 and 1.

Reservoir Layer. The reservoir can be thought of as a dynamical system characterized by a reservoir state vector $\vec{r}(t)$ which describes the state of the reservoir nodes as a function of time t. $\vec{r}(t)$ is given by

$$\vec{r}(t+1) = \tanh\left(W^{\text{in}} \cdot \vec{u}(t)\, W^{\text{res}} \cdot \vec{r}(t) + b\right) \quad (1)$$

The input weights matrix $W^{\text{in}} \in \mathbb{R}^{N_R \times N_u}$, where N_R is number of nodes in the reservoir and N_u is the dimension of the input vector $\vec{u}(t)$; here N_u is the number of rows of the concatenated image. The activity of the reservoir at time t is given by $\vec{r}(t)$, of size N_R. The recurrent connection weights $W^{\text{res}} \in \mathbb{R}^{N_R \times N_R}$ are set randomly between -1 and 1. b is a scalar bias. We use hyperbolic tangent as the nonlinear activation function. We rescale W^{res} to have a spectral radius γ (maximal absolute eigenvalue) of 0.5, but we observe no conclusive correlation or robust pattern between performance and this choice as seen in Figure 10. Our choice of spectral radius is in part influenced by the analysis of the effect of spectral radius on performance presented in [40]. The reservoir is a dynamical system that transforms the low dimensional input into a much higher dimensional reservoir space and is not affected by W^{in} and W^{res} being sparse, making it computationally faster. Matrix sparsity is 0.9

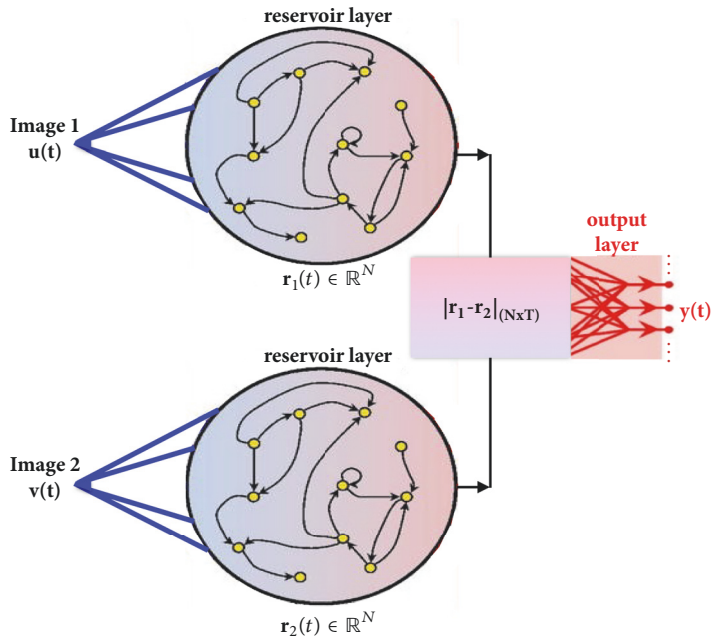


FIGURE 4: Dual reservoir architecture with input state of the two images at time t denoted by $\overrightarrow{u}(t)$ and $\overrightarrow{v}(t)$, reservoir state vectors by $\overrightarrow{r_{1,2}}(t)$, and output state by $\overrightarrow{y}$.

(90% of the entries are randomly chosen to be zero) unless otherwise stated.

Output Layer. In the single reservoir architecture, for one combined input image, the reservoir state trajectory, X, is formed by concatenating the $N_R \times 1$ reservoir state vectors (the state of all reservoir nodes) at every timestep $\overrightarrow{r}(t)$ as follows:

$$X = \left[\overrightarrow{r}(0) \;\; \overrightarrow{r}(t=1) \;\; \ldots \;\; \overrightarrow{r}(t=T)\right], \tag{2}$$

where X is an augmented matrix of size $N_R \times c$ and c is the number of columns in the image (number of time steps, T, through which the entire image is input). For the single reservoir case, X is the same as the reservoir *system* state trajectory, denoted by $\widetilde{X}$ for both single and dual reservoir architectures). $\widetilde{X}$ is the matrix obtained by processing the input through the reservoir architecture that is then used to generate the output weights.

The output/readout layer representation (Y_i) for a very similar pair is $(1,0,0,0,0)$, rotated pair is $(0,1,0,0,0)$, zoomed pair is $(0,0,1,0,0)$, blurred pair is $(0,0,0,1,0)$, and different pair is $(0,0,0,0,1)$. The output weights convert the reservoir system state trajectories $\widetilde{X}_k$ into the reservoir output y_i (whose values are reservoir predicted probabilities of each category). Ridge regression (see Appendix A) is then used to train the output weights of the reservoir. While testing, a fractional probability is allotted to each output label, and the image pair is classified into the label with the highest probability.

2.1.2. Dual Reservoir Architecture

Input Layer. In order to exploit the memory properties of RCs, for the dual reservoir architecture, the input is again converted to a time series. However unlike for the single reservoir architecture, we input each image (image 1 and image 2) column by column into two identical reservoirs, allowing the time axis to run across the columns of the image.

Reservoir Layer. The reservoir state vectors for the two reservoirs (corresponding to image 1 and image 2), $\overrightarrow{r_1}(t)$ and $\overrightarrow{r_2}(t)$, are given by

$$\begin{aligned}\overrightarrow{r_1}(t+1) &= \tanh\left(W^{\text{in}} \cdot \overrightarrow{u}(t) + W^{\text{res}} \cdot \overrightarrow{r_1}(t) + b\right)\\ \overrightarrow{r_2}(t+1) &= \tanh\left(W^{\text{in}} \cdot \overrightarrow{v}(t) + W^{\text{res}} \cdot \overrightarrow{r_2}(t) + b\right)\end{aligned} \tag{3}$$

where $\overrightarrow{u}(t)$, $\overrightarrow{v}(t)$ are inputs from image 1 and 2 respectively. The properties of the internal dynamics of the reservoir are identical and the same as the single reservoir. W^{res} for both reservoirs are identical and randomly chosen as outlined in the single reservoir case.

Output Layer. Contrary to the single reservoir case, here we have two distinct reservoirs. The reservoir state trajectory X_k of one individual reservoir for one image k is then formed by concatenating the reservoir state vector as in (2). However, for the dual reservoir, we obtain two individual reservoir state trajectories, whose difference forms the reservoir system state trajectory $\widetilde{X}_k$, that is used in the determination of the output weights.

The k^{th} reservoir system state trajectory is given by $\widetilde{X}_k = |X_{k1} - X_{k2}|$, where X_{k1}, X_{k2} are the reservoir state trajectories corresponding to the images 1 and 2 respectively, for the kth input image pair. The readout layer representations for different relationships are the same as that in the single reservoir case. Ridge regression (refer Appendix A) is then used to train the output weights of the reservoir.

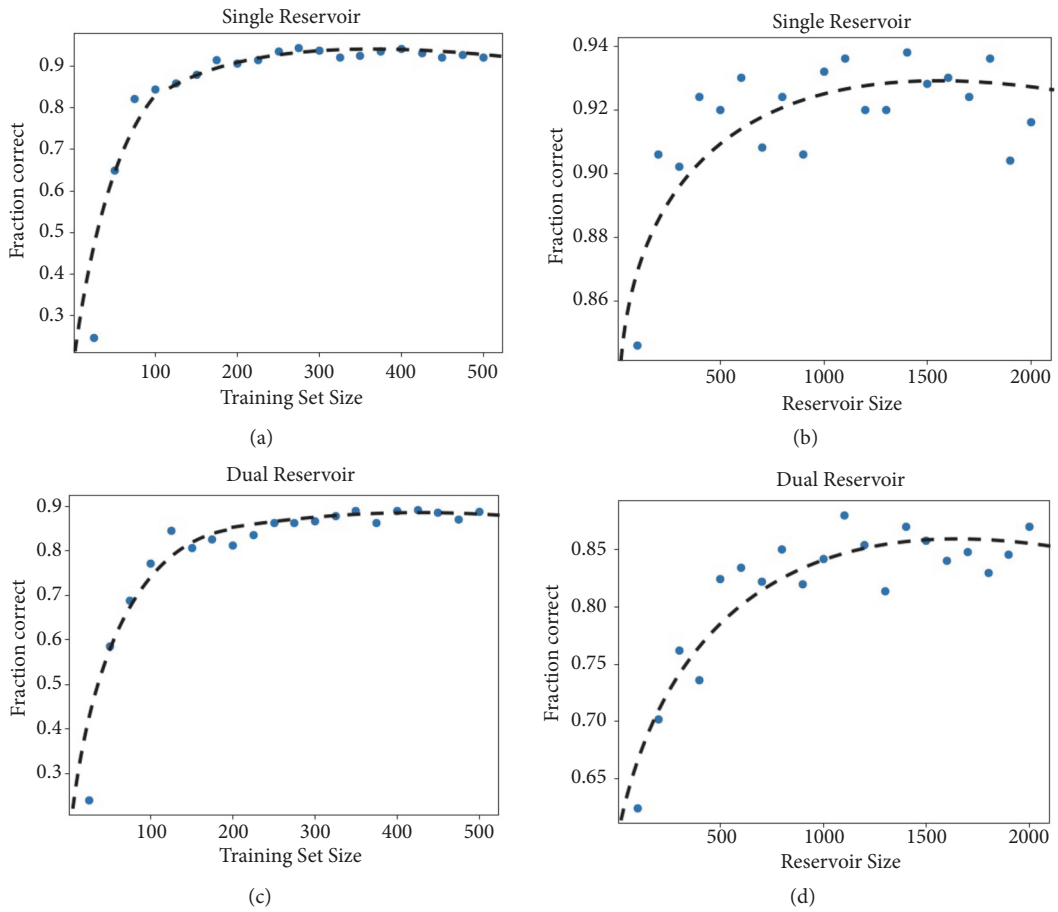


Figure 5: Fraction of image pairs correctly classified versus training set size (a and c) and reservoir size (b and d). Single reservoir results show in (a and b). Dual reservoir results shown in (c and d). Dashed curves denote fourth order polynomial best fit. Reservoir size=1000 nodes for (a and c); training size=250 pairs for (b and d). Spectral radius $\gamma = 0.5$, sparsity = 0.9, and testing size= 500 pairs.

3. Results

3.1. Generalization to Untrained Image Classes. In this section we discuss the performance of the single and dual reservoir in the task of generalization of learned relationships. We present the results obtained on the MNIST dataset as proof of concept. The systems were trained on the five relationships, noise added, 90° rotation, blur, zoom, and different (i.e., no relationship), on image pairs of handwritten digits 0-5. Then they were tested on identifying the same relationships (in equal measure) between image pairs of handwritten digits 6-9 (digits they have never seen before). We use fraction correct (1- error rate) as a metric of performance.

In Figures 5(a) and 5(c), we see that the reservoir performance increases rapidly with training set size and plateaus at around 200 training pairs. A training set size of ~250 image pairs gives a reasonable trade-off between performance and computational efficiency. This is significantly lower than the training set sizes typically used in machine learning. Hence, our system achieves an important goal for many biomimetic architectures, i.e., the ability to train with relatively few training examples. Figures 5(b) and 5(d) show that for a constant training data size (250 pairs) the performances increase as expected with reservoir size up to ~750 nodes after which it saturates. The overall performance of the single reservoir appears to be better than that of the dual reservoir for a given reservoir size.

Further, we examine the reservoir performance as a function of the spectral radius γ in Figure 10; we observe a significant spread in performance values across γ; however we see no definitive pattern or conclusive correlation between the spectral radius and performance for the range investigated. While we notice a better performance for $\gamma = 0.1$ in the single reservoir, this is neither consistent across the single and dual reservoir architectures, nor the boost in performance robust across all small γ values. For reference, reservoir activity, single node activity, and output weights are shown in the Appendix B.

3.2. Comparison with Siamese Neural Networks. The topic of generalized learning has, to the best of our knowledge, not been satisfyingly addressed using a dynamical-systems-based machine learning approach that renders itself to easy analysis. To assess the effectiveness of our approach, we compare the performance of RCs with variants of a Siamese Neural Network (SNN), a successful pair-based machine learning technique (SNN architecture illustrated in Figure 6). Specifically, we compare the single and dual reservoir model to three

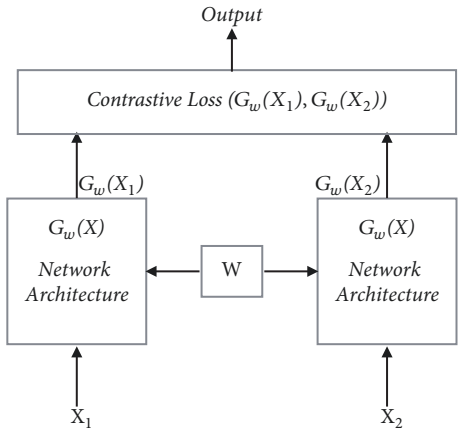


Figure 6: Siamese Network Architecture. Two inputs $X1$ and $X2$ are fed into two identical networks. $G_w(X)$ is the network transformation of the input X. W is the shared weights between the two heads of the Siamese architecture.

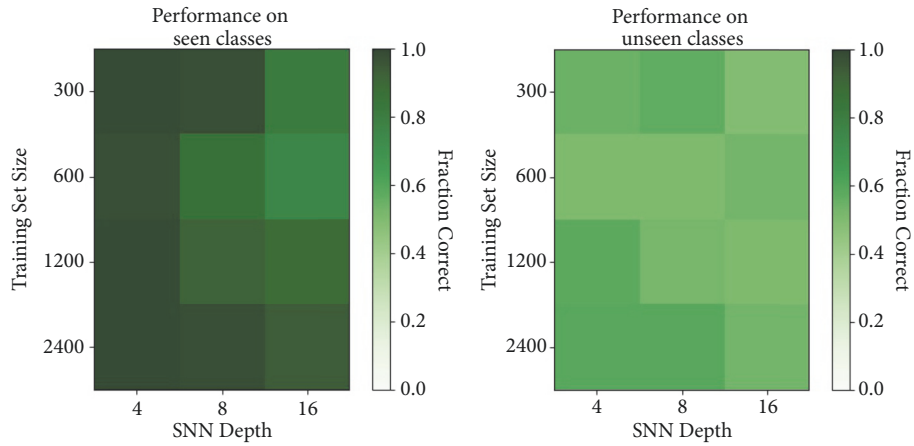
other architectures: a base SNN multilayer perceptron with 4 fully connected layers of 128 neurons each, a deep SNN multilayer perceptron with 8 fully connected layers of 128 neurons each, and a convolutional SNN (convolutional layer with 32 filters, 3×3 kernel, and a rectified linear nonlinearity, followed by 4 fully connected layers with 128, 64,32, and 2 neurons each). We compared performance for two binary classification tasks (Figure 7(c)): (1) learning the 90° rotation operator on MNIST image pairs; (2) learning to detect depth maps that come from the same visual scene class for the dataset of depth maps from a moving camera.

All SNN architectures were trained using contrastive loss (following [41]) and the optimizer Adadelta with a self-adjusting learning rate. The objective of our SNN is not classification but differentiation. Hence the contrastive loss function that pulls neighbors together and pushes nonneighbors away is a natural choice compared to classification loss functions such as cross entropy. The single and dual reservoirs have 1000 nodes with $\gamma = 0.5$ and sparsity 0.9. Training is done for a 100 (40) epochs on the base and deep SNN multilayer perceptrons and 40 (20) epochs on the convSNN for MNIST (visual scenes) data, respectively, and once on the reservoirs on 500 image pairs.

While we present a select few SNN architectures here (and selected choices of parameters), we tried several other SNN architectures including VGG16-SNN and deep convSNN and found their performance to be comparable to the representative SNN performances we have shown. We also show SNN multilayer perceptron performance on varying depth (number of layers) and varying training data size (varied in the lower range compared to traditional deep network training sizes for comparison with the RCs and to motivate the question of biological plausibility) while testing on seen (trained) classes and unseen (test) classes (Figures 7(a) and 7(b) respectively) and find that while the network performs fairly well on the trained classes, it performs consistently poorly on the unseen classes. The loss and accuracy plots for the SNN architectures for both tasks are in Appendix C.

3.2.1. Generalized Learning of the Rotation Operator on the MNIST Dataset. We train the reservoir on a simple binary classification task, i.e., classify image pairs from the MNIST dataset as having the relationship "rotated" or not. Our training set consists of rotated and not rotated images of digits 0-5. Figure 7(c) shows the fraction of correct classification by the RCs and the SNNs on the training classes (seen, digits 0-5) and testing classes (unseen, digits 6-9), as rotated or not rotated. We observe that while the performance of all the networks is comparable on training set digits (digits 0-5), all the SNN architectures seem to have a near-random percent correct for untrained digits (6-9). Performance did not improve on increasing the depth of the base SNN (Figures 7(a) and 7(b)). The reservoir performance remains equally good over trained digits (0-5) and untrained digits (6-9), indicative of learning of the underlying relationship in the pairs and not the individual digits themselves. From observations in Section 3.4, we speculate that the generalization ability of the reservoir may be attributed to the convergence of parts of the reservoir system state trajectories for all rotated image pairs. The dynamical properties of the reservoir create temporal patterns that enable memory. These properties may make learning on small datasets easier by requiring the RC to learn only some features of the dynamical patterns instead of the whole reservoir space. By contrast, the feed-forward SNN is not a dynamical system that enables temporally encoded memory, and training occurs explicitly on the images as opposed to the classes of relationships, which may be a possible cause for poorer performance while generalizing. For comparison, we present performance of a fully connected SNN upon varying depth and training data size in Figures 7(a) and 7(b).

3.2.2. Generalizing Similarities in Depth Perception from a Moving Camera. Identifying similarities in scenes and properties of scenes such as depth, style, etc. from a moving camera is an important problem in the field of computer vision [42, 43]. We are interested in studying how the reservoir could learn and generalize relationships between images of visual scenes from a moving camera, frames of which may be nonlinearly transformed with respect to each other. To demonstrate the practicality of our method, we implement it on depth maps from 6 different visual scenes recorded indoors in an office setting. Each visual scene has depth maps from 300 images, recorded as the camera is moved within a small distance (~30cm) and rotated within a small angle (~30°). We then train the networks to identify pairs of depth maps as very similar (same visual scene) or different (different visual scenes), learning to capture small spatial and rotational invariance. Training is done on 500 images each from the first three visual scenes. We study whether the systems are able to generalize, i.e., identify relationships between depth maps from the other three visual scenes. Figure 7(c) shows the reservoir performs significantly better on untrained scenes than the SNN, which classifies randomly. Both systems have a comparable and very high performance on the trained scenes. Thus, the reservoir is able to identify frames with similar depth maps from scenes it has not seen before. This has potential



(a) (b)

Fraction correct for different network architectures

	Single Reservoir	Dual Reservoir	Base SNN	Deep SNN	ConvSNN
Digits 6-9 (unseen classes)	**0.91**	**0.89**	0.54	0.51	0.53
Digits 0-5 (seen classes)	0.94	0.91	0.93	0.91	0.89
Scenes 4-6 (unseen classes)	**0.85**	**0.95**	0.53	0.51	0.78
Scenes 0-3 (seen classes)	0.87	0.95	0.89	0.89	0.93

(c)

Figure 7: SNN perceptron performance on trained (seen) classes (a) and test (unseen) classes (b) of MNIST data as a function of training dataset size and SNN perceptron depth. (c) Classification accuracy (fraction correct) of the single and dual reservoir, base SNN, deep SNN, and convSNN on seen (trained) classes and unseen (test) classes, on (1) identifying rotation transformation in MNIST images; and (2) identifying similar visual scenes from a moving camera. Training size: 500 images; testing size: 500 images.

applications in scene or object recognition using a moving camera.

3.3. Combining Relationships. In this section we train the reservoir independently on the five relationships as in previous sections. However our test images have a linear combination of multiple relationships applied on them simultaneously (e.g., rotated as well as blurred). We then study the ability of the reservoir to recognize all the separate relationships applied to the test input pair.

Several tests on subsets/combinations of relationships were performed; however we only present a few demonstrative cases here. Training is done on the five individual relationships (noise, rotated, blurred, zoomed, and different) for digits 0-5. Here we present testing on a combination of 3 relationships (90° rotation, zoom and blur), combination of 2 relationships (90° rotation and blur) as well as solo 90° rotation for digits 6-9. For testing image pairs with n relationships applied simultaneously, we consider the reservoir to have classified correctly if the n highest label probabilities predicted by the reservoir during testing correspond to the n applied relationships. In Figure 8 we observe that both the single and dual reservoirs perform very well (in terms of percent correct) at identifying combined relationships in images that they have never seen before. The single reservoir, on average, performs slightly better than the dual reservoir. While there may be some inherent biases (ex. in Figure 8(f), the dual reservoir shows a bias towards the zoomed category), in spite of the biases, the reservoirs are able to not only identify and separate linear combinations of these relationships, but also generalize this knowledge to previously unseen classes. We speculate that this ability to generalize combinations of multiple relationships is a result of overlap of reservoir system state trajectory clusters that correspond to the separate relationships. The cases shown in Figure 8 are representative of the higher end of the range of accuracies obtained with other combinations (not presented here) as well.

3.4. Clustering Reservoir Space. Here we present a study of the features reservoir system state trajectories that may be important for generalization. In order to generalize, for a given relationship between the input image pair, there must be a corresponding relationship between the reservoir activity, dependent only on the relationship between the input images and not on the specific features of the input

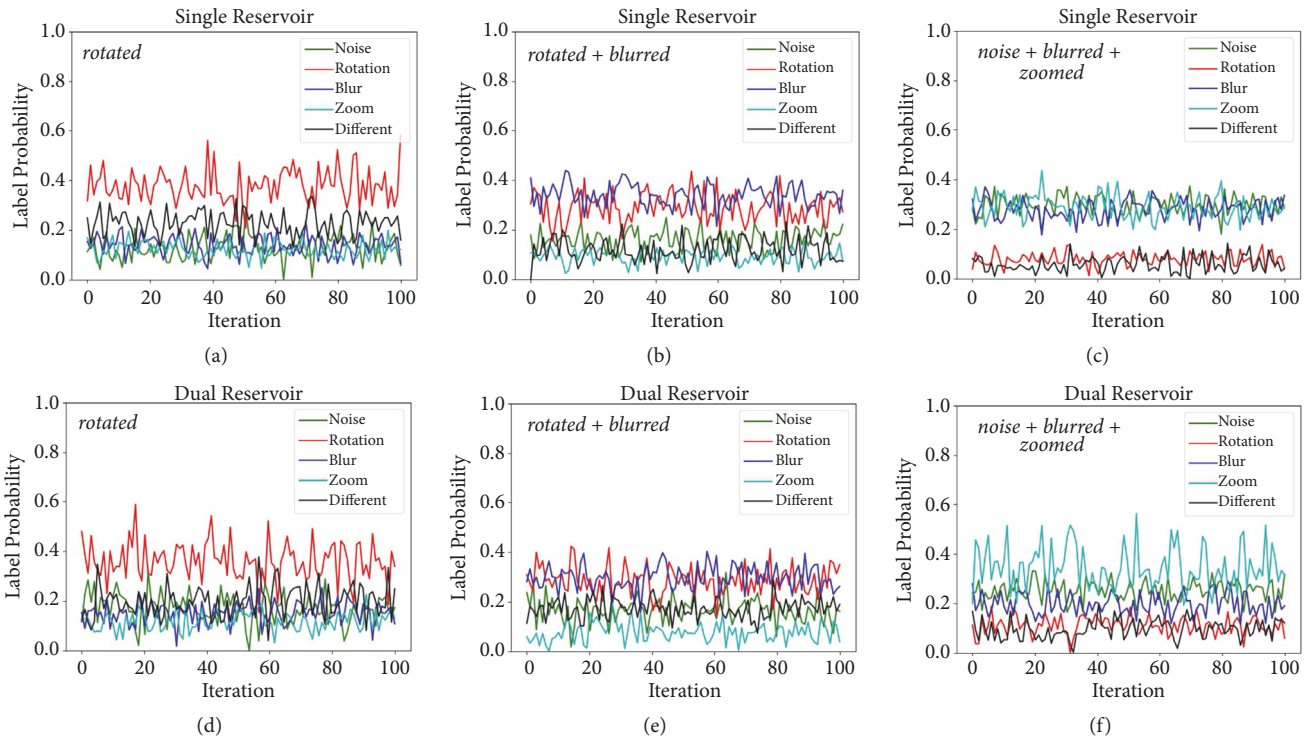


FIGURE 8: Graphing probabilities of the reservoir output (y_k) versus the image pair number k, for image pairs that are rotated (a, d) 2 combination: rotated and blurred (b, e); 3 combination: noise, blurred, and zoomed (c, f), for single and dual reservoir, respectively. The fractions correct, where classification is considered to be correct if the n predicted maximum probability labels are the n transformations applied to the test image pair (shown on top left of each panel), are 0.97, 0.97, 1.0, 0.93, 0.84, and 0.93 for (a, b, c, d, e, f), respectively. γ=0.5; reservoir size=1000. Training digits: 0-5; testing digits: 6-9. Training size: 250 pairs.

images themselves. As discussed earlier, the reservoir serves as a nonlinear filter, whose goal for classification problems is to map the input into a high-dimensional space where the different relationships become linearly separable. In addition, the dynamical properties of the reservoir allow it to encode memory (because the reservoir state at time t depends on its state at time $t-1$). In this way, the reservoir's dynamical activity pattern in response to the input can highlight important features/relationships within the temporalized input. In this section we illustrate that reservoir system state trajectories corresponding to a relationship do indeed cluster/become separable in reservoir space, allowing for generalization.

In Figure 9, we plot a representation of 500 reservoir system state trajectories for each relationship (using different input digits; equally sampled) for (a) the single reservoir and (b) the dual reservoir. We show here the five standard relationships for MNIST, noise, rotate, blur, zoom, and different, as well as one combined relationship, blur+rotate. A single reservoir system state trajectory has a very high dimensionality ($N_R \times T$). We are interesting in viewing this high-dimensional data in a reduced dimensional space. Hence, we use the following dimensionality reduction techniques: first, we use Principal Component Analysis (PCA) to extract the 100 largest principal components (PCs) of each reservoir system state trajectory. We then use the t-Distributed Stochastic Neighbor Embedding (t-SNE) technique [44] on the extracted PCs for further dimensionality reduction. t-SNE, being particularly well suited for the visualization of high-dimensional datasets, has been used very successfully in recent years along with PCA.

We visualize the reservoir system state trajectories in a two-dimensional space and find that relationships between images cluster. We observe from Figure 9 that the separation of relationships is more prominent for the dual reservoir (Figure 9(b)) compared to the single reservoir (Figure 9(a)). This may be attributed to the architecture of the dual reservoir, which takes the difference between the individual image trajectories, thus more directly encoding the classification features, i.e., the features of the *differences* between the image pair (blur, scale, rotation feature, etc.), unlike the single reservoir. However, we note that despite the fact that we see better separation of of clusters for the dual reservoir, the single reservoir slightly outperforms it on the MNIST data (see Figure 7(b)). One possible reason for this is that Figure 9 only shows a two-dimensional representation of the clusters and perhaps the single reservoir shows a better separation than the dual reservoir in higher dimensions. Another possible reason is that the reservoir system state trajectories do not take into account the training, which, in addition to the clustering of reservoir trajectories, is a key component of the reservoir's performance. There may be some features of the reservoir system state trajectories from the single reservoir architecture that are not captured in Figure 9 yet allow for more effective training.

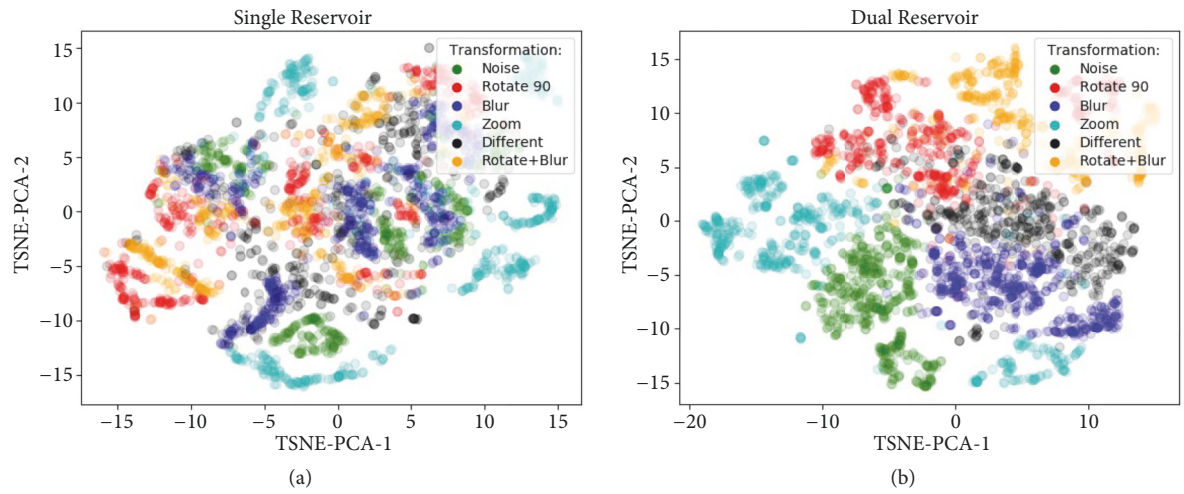


FIGURE 9: 500 reservoir system state trajectories for each relationship in the reduced dimensional space spanned by the two largest components obtained using t-SNE on the 100 largest principal components of the reservoir system state trajectory for (a) single reservoir and (b) dual reservoir. Input images: digits 0-9 of MNIST dataset. N_R: 1000. t-SNE iterations: 300. perplexity: 40.

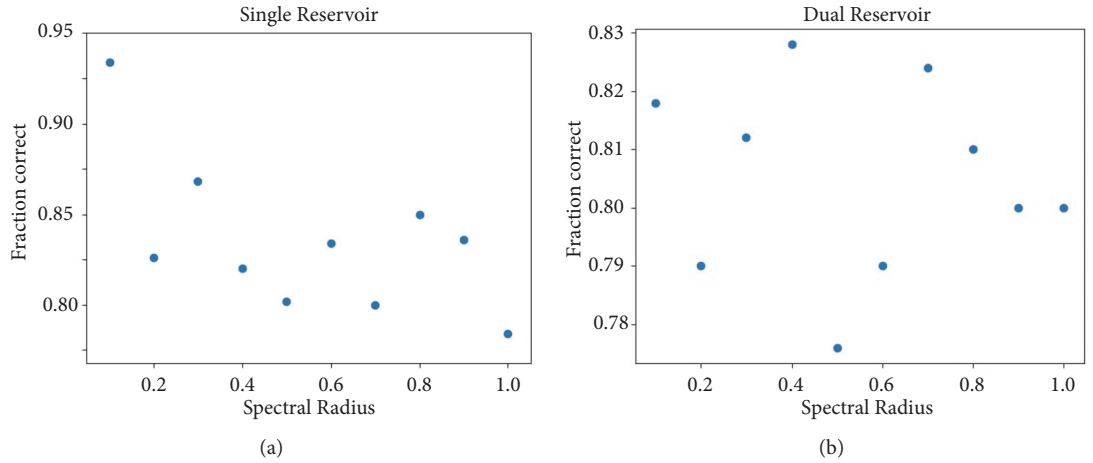


FIGURE 10: Fraction correct as a function of spectral radius for (a) single reservoir; (b) dual reservoir. N_R=1000; training size=250 pairs; testing size=500 pairs; $\gamma = 0.5$; sparsity = 0.9.

We speculate that the separation of the system trajectories in reservoir space is important for generalizing with small datasets when using a linear training procedure like ours. Here, we have demonstrated that the reservoir does indeed function as an effective nonlinear filter that acts upon the image pairs and separates them in high-dimensional reservoir space into clusters characterized by the relationships between the two input images.

4. Conclusion

In this paper we have used Reservoir Computers (RCs) for image classification problems that involve generalization of relationships learned between image pairs using limited training data. While image classification has been studied extensively before, here we present a biologically inspired recurrent network approach that not only generalizes learning, but also allows us to build an interpretation of the results. We present our results on the simple handwritten digits database, as well as on a video dataset of depth maps from a moving camera, useful in identification of similar scenes from different camera perspectives. We observe that the reservoir system state trajectories obtained from input image pairs with the same relationship cluster in reservoir space. This can be interpreted as the reservoir trajectory exhibiting dynamical patterns corresponding to image pair relationships. Because the reservoir system state trajectories separate in the high-dimensional reservoir space according to the input pair relationships, a linear method of training such as ridge regression is effective. The separability of the clusters allows for training to converge relatively quickly and with limited training data. By reducing dimensionality from the reservoir space to the space mapped by the clusters, we obtain a well-generalizing reservoir using only a small training dataset, whereas contemporary methods such as deep learning require much larger datasets. Although we see

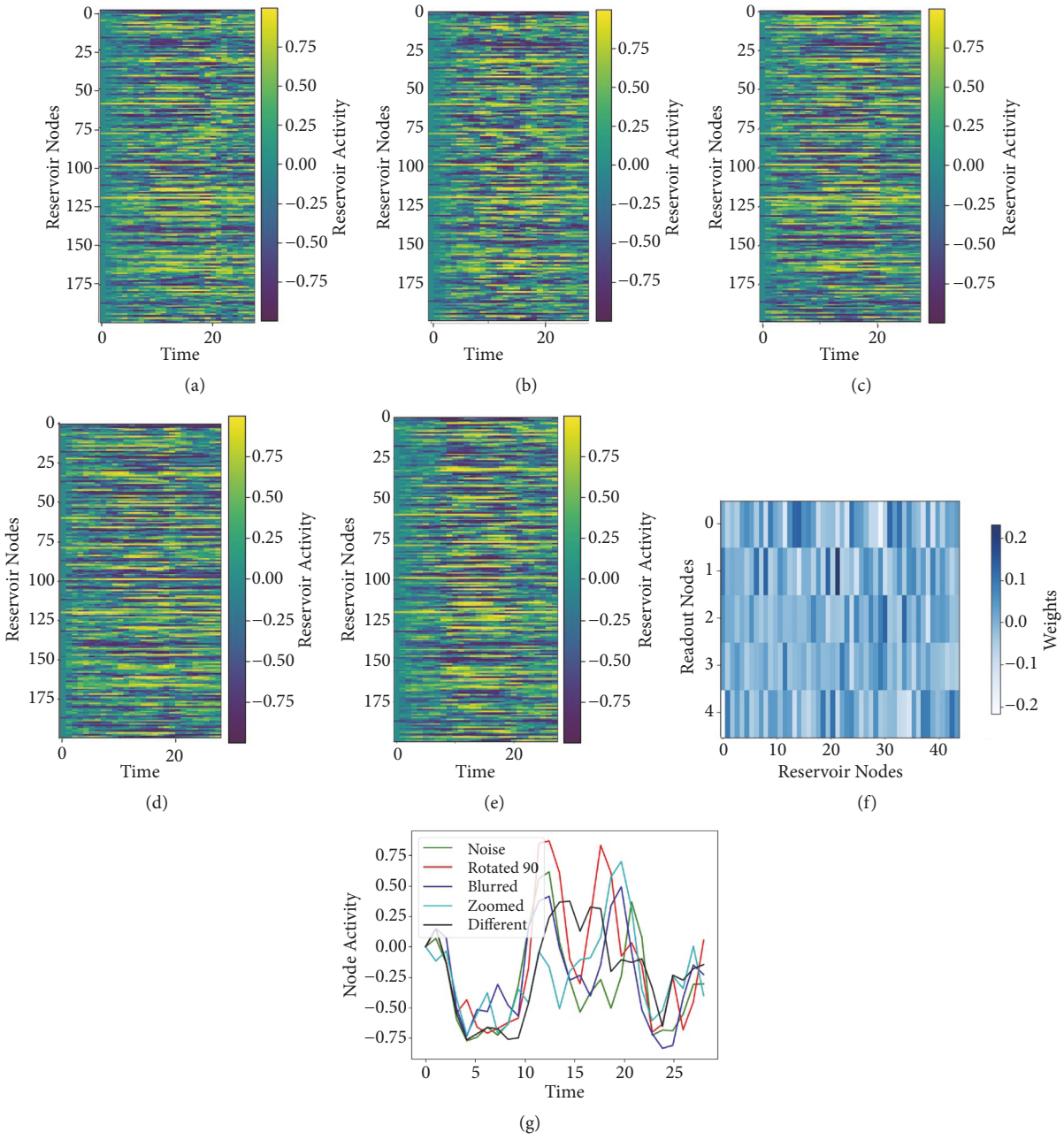


FIGURE 11: Reservoir activity for the single reservoir architecture. (a), (b), (c), (d), and (e) show the differential reservoir activity of 200 nodes over 28 timesteps for input relationships noise, rotated, zoomed, blurred, and different, respectively. (f) shows the output weight matrix(W^{out}) for 50 reservoir nodes. (g) shows activity of a random node for all output labels over 28 timesteps. N_R: 1000; $\gamma = 0.5$; sparsity= 0.9.

strong performance with a sparse reservoir and few training images in our proof-of-concept study, we suspect that, for more complex input images, a more powerful (and possibly more sophisticated) architecture would be required to match performance.

We find that the RCs perform significantly better than deep/convolutional SNNs for the task of generalization. From a computation perspective, the RC has the added advantage of speed since only the output weights are trained and the reservoir is sparsely connected. Our system is biologically inspired in two ways. First, the learning mimics biological learning through comparisons and analogies. Second, the internal dynamics of the reservoir are known to broadly resemble neural cortex activity. We conclude that although state-of-the-art machine learning techniques such as SNNs (for pairwise input) work exceedingly well for traditional image classification, they do not work as well for generalization of learning, for which RCs significantly outperform them in our study, perhaps due, in part, to their dynamical "memory" properties that lead to distinctive dynamical patterns in the reservoir state trajectories. While more complex architectures such as LSTMs may also have much greater success in

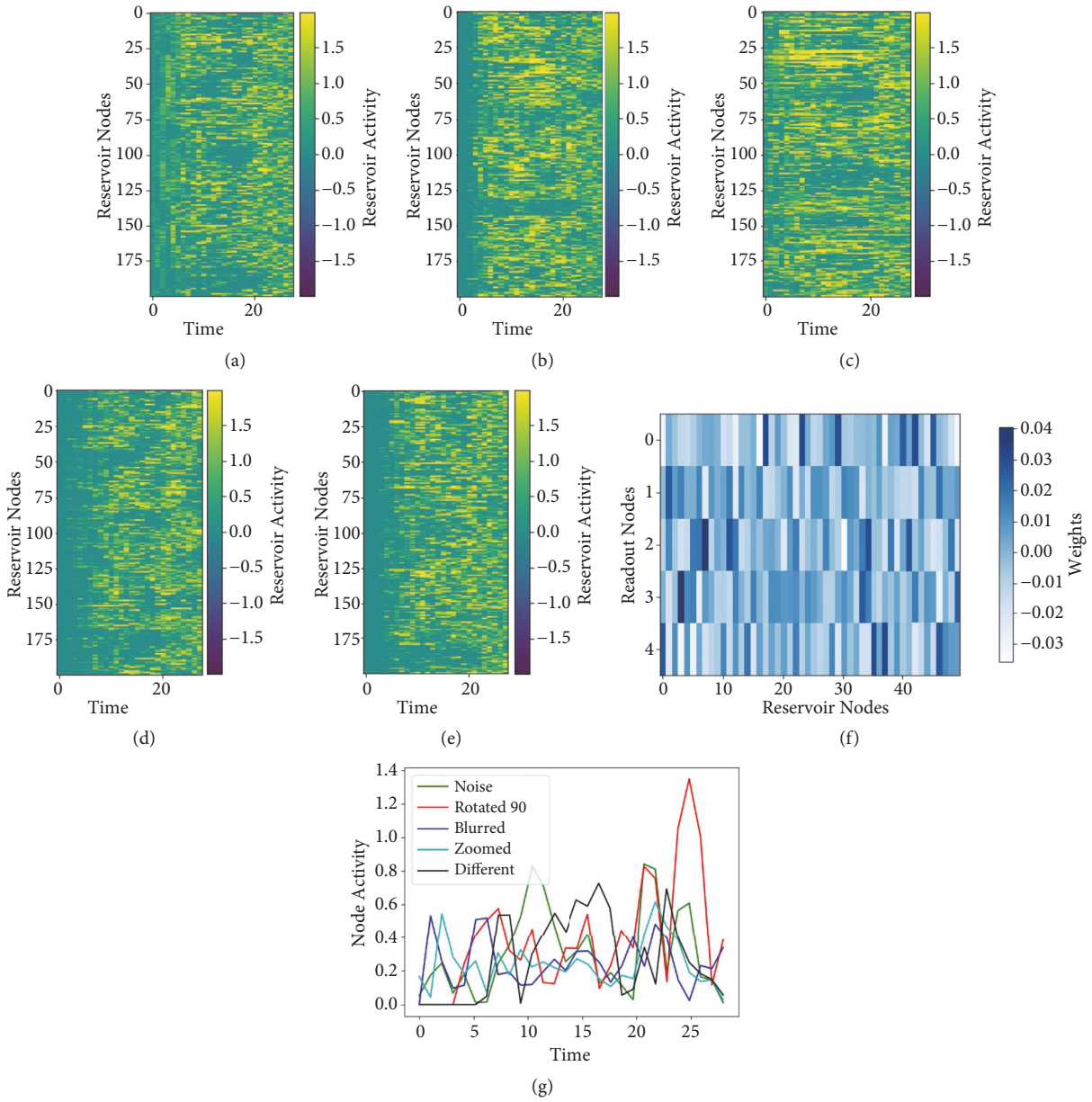


Figure 12: Reservoir activity for the dual reservoir architecture. (a), (b), (c), (d), and (e) show the differential reservoir activity of 200 nodes over 28 timesteps for input relationships noise, rotated, zoomed, blurred, and different respectively. (f) shows the output weight matrix(W^{out}) for 50 reservoir nodes. (g) shows activity of a random node for all output labels over 28 timesteps. N_R: 1000; $\gamma = 0.5$; sparsity= 0.9.

generalization than nonrecurrent architectures, they require much larger training data and more computational power. However, implementing the experiment on an LSTM network could be an interesting future direction, especially for more challenging generalization problems.

We see the strength of our work is lying not only in its demonstration of the utility of RCs for generalization using small datasets, but also in our ability to interpret this in terms of the clustering of the dynamics of the reservoir system state trajectory. This relates to new ideas in explainable Artificial Intelligence (AI), a topic that continues to receive traction. An interesting direction would be to explore different reservoir-like architectures that model the human brain better. Another promising direction would be to study synchronization patterns in the reservoir and their role in learning.

Appendices

A. Ridge Regression and Training

Only the output weight matrix W^{out} is optimized during training such that it minimizes the mean squared error $E(y, Y)$ between the output of the reservoir y and the target signal Y. The reservoir output is

$$y = W^{\text{out}}\delta X \tag{A.1}$$

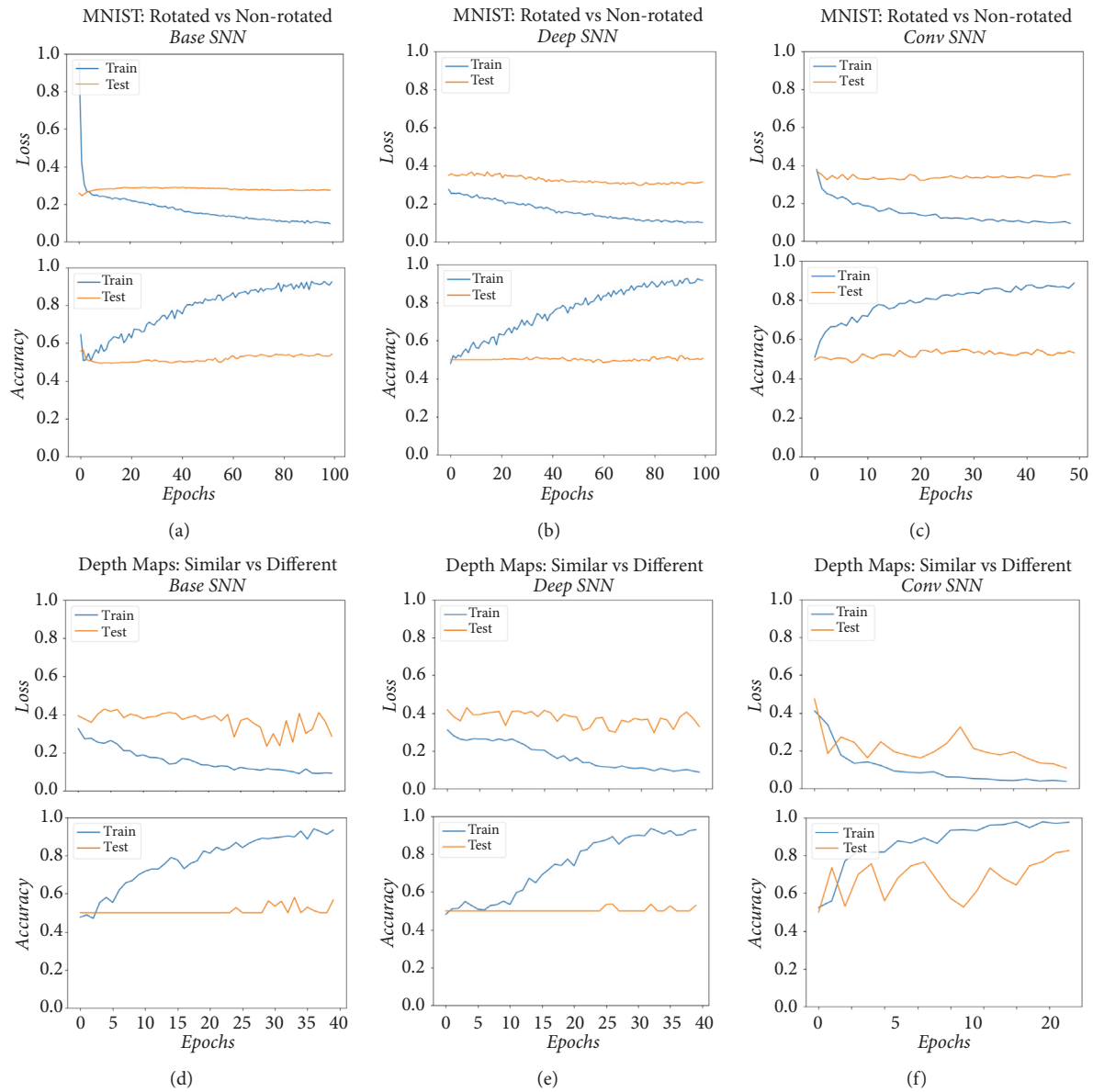


FIGURE 13: Plot of training loss and accuracy for (a and c) base Siamese network, (b and d) deep Siamese network, and (c and f) convolutional Siamese network.

$W^{\text{out}} \in \mathbb{R}^{N_y \times N_R}$ where N_y is the dimensionality of the readout layer.

δX or the concatenated reservoir system state trajectory is the matrix containing all reservoir system state trajectories during training phase, $\delta X = \left[\widetilde{X}_0 \;\; \widetilde{X}_1 \;\; \ldots \;\; \widetilde{X}_M\right]$ where M is the total number of training image pairs, input one after the other, and $Y = [Y_0 \;\; Y_1 \;\; \ldots \;\; Y_M]$ is the matrix containing the corresponding readout layer for all images. W^{out} is computed using Ridge Regression (or Tikhonov regularization) [45], which adds an additional small cost to least square error, thus making the system robust to overfitting and noise. Ridge regression calculates W^{out} by minimizing squared error $J(W^{\text{out}})$ while regularizing the norm of the weights as follows:

$$J\left(W^{\text{out}}\right) = \eta \left|W^{\text{out}}\right|^2 + \sum_i \left(\left(W^{\text{out}}\right)^T \delta X_i - Y_i\right)^2. \quad \text{(A.2)}$$

where δX is the concatenated reservoir system state trajectories over all training image pairs, Y contains the corresponding label representations, and the summation is over all training image pairs. Upon solving the stationary condition $\partial J / \partial W^{\text{out}} = 0$ is

$$W^{\text{out}} = \left(\delta X \delta X^T + \eta I\right)^{-1} \delta X Y. \quad \text{(A.3)}$$

where η is a regularization constant and I is the identity matrix.

B. Reservoir Dynamics and Performance

We present the performance of the single and dual reservoir as a function of spectral radius γ. γ is varied from 0 to 1 while looking for the optimal performance region where the

reservoir has memory or is in the "echo state" (edge of chaos) [46]; however we find no indicative pattern (Figure 10).

Performance with Spectral Radius. Figure 10 shows fraction correct as a function of reservoir dynamics for (a) single and (b) dual reservoir.

Reservoir Dynamics. For completion, we plot the reservoir activity, i.e., averaged reservoir system state trajectory corresponding to our five relationships applied to the MNIST dataset, output weights, and single node activity. Figures 11 and 12 show plots of activity in the single reservoir and dual reservoir architecture, respectively. We see that the individual node (f) itself does not encode any decipherable information. However each output label (a, b, c, d, and e) has a slightly different signature in reservoir space.

C. Loss and Accuracy of SNNs

In Figure 13 we plot the training loss and accuracy for the base SNN (4 layers), deep SNN (8 layers), and convolutional SNN for the two tasks of identifying rotation operator in MNIST and identifying similar visual scenes from a moving camera. Since training data is small, losses converge fairly quickly over epochs. The optimizer Adadelta, which employs a variable learning rate, was used in training.

Data Availability

The visual scenes captured from a moving camera (images and depth maps) dataset used to support the findings of this study are available from the corresponding author upon request. The dataset has not been included in the article/supplementary material due to its large size. Code is available at https://github.com/a-jan-tusk/Reservoir-for-generalization.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

The authors would like to thank Professor Brian Hunt for his insightful discussions and helpful feedback. This research was supported in part by the University of Maryland's COMBINE (Computation and Mathematics for Biological Networks) program through NSF Award no. 1632976, the ONR under grant award N00014-17-1-2622, and a DoD contract under the Laboratory of Telecommunication Sciences Partnership with the University of Maryland.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, Mass, USA, 2016.

[3] B. Du, W. Xiong, J. Wu, L. Zhang, L. Zhang, and D. Tao, "Stacked Convolutional Denoising Auto-Encoders for Feature Representation," *IEEE Transactions on Cybernetics*, vol. 47, no. 4, pp. 1017–1027, 2017.

[4] Zachary Chase Lipton. A critical review of recurrent neural networks for sequence learning. CoRR, abs/1506.00019, 2015.

[5] W. Zhu, T. Yao, J. Ni, B. Wei, and Z. Lu, "Dependency-based Siamese long short-term memory network for learning sentence representations," *PLoS ONE*, vol. 13, no. 3, 2018.

[6] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2017*, pp. 4845–4849, USA, March 2017.

[7] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks. 148, 01 2001.

[8] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: a new framework for neural computation based on perturbations," *Neural Computation*, vol. 14, no. 11, pp. 2531–2560, 2002.

[9] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[10] P. Enel, E. Procyk, R. Quilodran, and P. F. Dominey, "Reservoir Computing Properties of Neural Dynamics in Prefrontal Cortex," *PLoS Computational Biology*, vol. 12, no. 6, 2016.

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[12] J. Mueller and A. Thyagarajan, "Siamese recurrent architectures for learning sentence similarity," *Association for the Advancement of Artificial Intelligence*, vol. 16, pp. 2786–2792, 2016.

[13] I. Melekhov, J. Kannala, and E. Rahtu, "Siamese network features for image matching," in *Proceedings of the 23rd International Conference on Pattern Recognition, ICPR 2016*, pp. 378–383, Mexico, December 2016.

[14] S. Bell and K. Bala, "Learning visual similarity for product design with convolutional neural networks," *ACM Transactions on Graphics*, vol. 34, no. 4, 2015.

[15] M. Inubushi and K. Yoshimura, "Reservoir Computing beyond Memory-Nonlinearity Trade-off," *Scientific Reports*, vol. 7, no. 1, 2017.

[16] N. M. da Costa and K. A. Martin, "The proportion of synapses formed by the axons of the lateral geniculate nucleus in layer 4 of area 17 of the cat," *Journal of Comparative Neurology*, vol. 516, no. 4, pp. 264–276, 2009.

[17] S. Haeusler and W. Maass, "A statistical analysis of information-processing properties of lamina-specific cortical microcircuit models," *Cerebral Cortex*, vol. 17, no. 1, pp. 149–162, 2007.

[18] W. Singer Danko Nikolic, S. Haeusler, and M. Wolfgang, "Temporal dynamics of information content carried by neurons in the primary visual cortex," in *the NIPS'06 Proceedings of the 19th International Conference on Neural Information Processing Systems*, 2006.

[19] C. Van Vreeswijk and H. Sompolinsky, "Chaos in neuronal networks with balanced excitatory and inhibitory activity," *Science*, vol. 274, no. 5293, pp. 1724–1726, 1996.

[20] R. Rasmussen, M. H. Jensen, and M. L. Heltberg, "Chaotic Dynamics Mediate Brain State Transitions, Driven by Changes in Extracellular Ion Concentrations," *Cell Systems*, vol. 5, no. 6, pp. 591–603.e4, 2017.

[21] P. J. Urcuioli, E. A. Wasserman, and T. R. Zentall, "Associative concept learning in animals: Issues and opportunities," *Journal of the Experimental Analysis of Behavior*, vol. 101, no. 1, pp. 165–170, 2014.

[22] M. Giurfa, S. Zhang, A. Jenett, R. Menzel, and M. V. Srinivasan, "The concepts of 'sameness' and 'difference' in an insect," *Nature*, vol. 410, no. 6831, pp. 930–933, 2001.

[23] S. C. Andrew, C. J. Perry, A. B. Barron, K. Berthon, V. Peralta, and K. Cheng, "Peak shift in honey bee olfactory learning," *Animal Cognition*, vol. 17, no. 5, pp. 1177–1186, 2014.

[24] M. Brenden Lake, D. Tomer Ullman, B. Joshua Tenenbaum, and J. Samuel Gershman, "Building machines that learn and think like people," *Behavioral and Brain Sciences*, vol. 40, 2017.

[25] Y. Duan, M. Andrychowicz, B. Stadie et al., "One-shot imitation learning," in *Proceedings of the In Advances in neural information processing systems*, pp. 1087–1098, 2017.

[26] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Acadamy of Sciences of the United States of America*, vol. 79, no. 8, pp. 2554–2558, 1982.

[27] H. Jaeger, Controlling recurrent neural networks by conceptors, 2014.

[28] Y. Qi, Y. Song, H. Zhang, and J. Liu, "Sketch-based image retrieval via Siamese convolutional neural network," in *Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP)*, pp. 2460–2464, Phoenix, AZ, USA, September 2016.

[29] C. Zhang, W. Liu, H. Ma, and H. Fu, "Siamese neural network based gait recognition for human identification," in *Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2832–2836, Shanghai, March 2016.

[30] A. Dutta, U. Pal, and J. Llados, "Compact correlated features for writer independent signature verification," in *Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 3422–3427, Cancun, December 2016.

[31] K. Gregory, Z. Richard, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," *In ICML Deep Learning Workshop*, vol. volume 2, 2015.

[32] A. Mehrotra and A. Dukkipati, in *Proceedings of the Computer Vision and Pattern Recognition*, vol. 2017, arXiv:1703.08033.

[33] 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: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 12, 121102, 9 pages, 2017.

[34] 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: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 4, 2017.

[35] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, "Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach," *Physical Review Letters*, vol. 120, no. 2, 2018.

[36] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, "Information processing capacity of dynamical systems," *Scientific Reports*, vol. 2, 2012.

[37] N. Schaetti, M. Salomon, and R. Couturier, "Echo State Networks-Based Reservoir Computing for MNIST Handwritten Digits Recognition," in *Proceedings of the 19th IEEE International Conference on Computational Science and Engineering, 14th IEEE International Conference on Embedded and Ubiquitous Computing and 15th International Symposium on Distributed Computing and Applications to Business, Engineering and Science, CSE-EUC-DCABES 2016*, pp. 484–491, France, August 2016.

[38] J. Bullier, "Integrated model of visual processing," *Brain Research Reviews*, vol. 36, no. 2-3, pp. 96–107, 2001.

[39] E. Volle, S. J. Gilbert, R. G. Benoit, and P. W. Burgess, "Specialization of the rostral prefrontal cortex for distinct analogy processes," *Cerebral Cortex*, vol. 20, no. 11, pp. 2647–2659, 2010.

[40] M. C. Ozturk, D. Xu, and J. C. Príncipe, "Analysis and Design of Echo State Networks," *Neural Computation*, vol. 19, no. 1, pp. 111–138, 2007.

[41] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*, pp. 1735–1742, June 2006.

[42] H. Rahmani, A. Mian, and M. Shah, "Learning a deep model for human action recognition from novel viewpoints," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1-1.

[43] A. Zeng, K.-T. Yu, S. Song et al., "Multi-view self-supervised deep learning for 6D pose estimation in the Amazon Picking Challenge," in *Proceedings of the 2017 IEEE International Conference on Robotics and Automation, ICRA 2017*, pp. 1386–1393, Singapore, June 2017.

[44] L van der Maaten and G. Hinton, "Viualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[45] F. Wyffels, B. Schrauwen, and D. Stroobandt, "Stable output feedback in reservoir computing using ridge regression," in *In Proceedings of the 18th International Conference on Artificial Neural Networks*, 2008.

[46] G. K. Venayagamoorthy and B. Shishir, "Effects of spectral radius and settling time in the performance of echo state networks," *Neural Networks*, vol. 22, no. 7, pp. 861–863, 2009.