
Hybrid Backpropagation Parallel Reservoir Networks

Matthew S. Evanusa *
 Department of Computer Science
 University of Maryland
 College Park, MD
 evanusa@cs.umd.edu

Snehash Shrestha
 Department of Computer Science
 University of Maryland
 College Park, MD

Michelle Girvan
 Department of Physics
 University of Maryland
 College Park, MD

Cornelia Fermüller
 Institute for Adv. Computer Studies
 University of Maryland
 College Park, MD

Yiannis Aloimonos
 Department of Computer Science
 University of Maryland
 College Park, MD

Abstract

In many real-world applications, fully-differentiable RNNs such as LSTMs and GRUs have been widely deployed to solve time series learning tasks. These networks train via Backpropagation Through Time, which can work well in practice but involves a biologically unrealistic unrolling of the network in time for gradient updates, are computationally expensive, and can be hard to tune. A second paradigm, Reservoir Computing, keeps the recurrent weight matrix fixed and random. Here, we propose a novel hybrid network, which we call Hybrid Backpropagation Parallel Echo State Network (HBP-ESN) which combines the effectiveness of learning random temporal features of reservoirs with the readout power of a deep neural network with batch normalization. We demonstrate that our new network outperforms LSTMs and GRUs, including multi-layer "deep" versions of these networks, on two complex real-world multi-dimensional time series datasets: gesture recognition using skeleton keypoints from ChaLearn, and the DEAP dataset for emotion recognition from EEG measurements. We show also that the inclusion of a novel meta-ring structure, which we call HBP-ESN M-Ring, achieves similar performance to one large reservoir while decreasing the memory required by an order of magnitude. We thus offer this new hybrid reservoir deep learning paradigm as a new alternative direction for RNN learning of temporal or sequential data.

1 Introduction and Motivations

Temporal learning is still a hard task that currently requires massive network architectures to achieve satisfactory results. In order to process temporal information with neural networks, the network needs to keep a memory of its past activity. This memory can be accomplished by either feeding the entire sequence into a feed-forward network, as in the style of transformer-attention architectures [50], or by feeding back the activity from the previous state into the neuron at the next time step. The main

*Corresponding author. Code available at: <https://github.com/Symbiomancer/HBP-ESN>

drawback of Transformer networks is that they require the entire sequence be passed in at once, and this prevents online learning. The domains of deep learning and reservoir computing need not be mutually exclusive, and years of research into optimizing deep networks can benefit the reservoir paradigm. Here, we focus on a novel RNN architecture and compare it to other state-of-the-art RNN architectures that take in input one item at a time, rather than as the entire sequence.

1.1 Engineering Motivation

A fundamental assumption in all learning tasks is that the 'real' world from which we sample is a dynamical system – each state of the latent variables of the outside world can be described by a differential equation given the last state. The feed-forward approach to learning temporal sequences amounts to a sampling of the sequence at regular intervals, taking advantage of Takens' theorem [22], which says that a dynamical system can be approximated using a sufficient number of samples of the system in a feed-forward network. If one chooses to learn temporal sequences with a feed forward network in this manner, however, the model size must be drastically increased to account for each temporal step, and arbitrary-length sequences cannot be used. The feed forward network must act as an unrolled recurrent network.

A recurrent neural network compacts a deep feed-forward network into a more efficient representation [49]. However, the issue then becomes how to learn reusable recurrent weights for an arbitrary sequence. This work proposes a novel method that avoids gradient-based updates in the recurrent layers. Here we argue that reservoir-type recurrent networks offer an efficient manner of temporal processing *without* having to unroll the network, and the complications that come with learning on the unrolled network.

1.2 Biological Motivation

In addition to power consumption and metabolic limitations, it has been argued that biological brains also 'compact' the unrolled network into a recurrent one [49]. Beyond just merely having recurrent connections, there is a biological argument that human brain regions actually behave as a reservoir [13], and that reservoir computing can offer a method to study cortical dynamics [44]. Furthermore, [42] argued that deep learning is lacking concepts inherent to the prefrontal cortex, and one of them is the existence of recurrent connections.

It is important to note that we do not argue that the current architecture is fully biologically plausible, because it employs backpropagation. However, backpropagation here is used only as a readout mechanism (see section 2), and in the future it could be replaced with any suitable supervised learning mechanism that can learn non-linear classifications, including any biologically-realistic ones; this does not effect the reservoir mechanism.

Our main philosophy is that engineering and biological needs are not mutually exclusive [49] and can inform each other in the generation of new learning algorithms, especially when the end goal of A.I. is a learning system that acts and behaves like human intelligence [27].

1.3 Recurrent Neural Networks

1.3.1 Types of Recurrent Networks

Allowing the network to learn not just static features, but also temporal information, increases the learning capacity of the network [21, 51]. Within the recurrent neural network (RNN) umbrella, we have two paradigms: a) fully differentiable recurrent networks, such as LSTMs [19], and b) Reservoir Computing (RC). We will not discuss here another paradigm, Spiking Neural Networks (SNNs) [18], which are by original design not a purely recurrent architecture, but can be made one (see below for LSMs), which muddies the classification. In fully differentiable RNNs, the learning mechanism required needs to handle backpropagating errors through the recurrent loops, and so algorithms such as unrolling backpropagation (BPTT) are currently the most widely used [53], although newer alternatives to BPTT do exist [3]. In reservoir computing, the recurrent connections are kept random and fixed, and only the readout of the reservoir is trained, i.e., the states of the reservoir are mapped to targets using a learning algorithm; this removes the need to backpropagate the error through the recurrent layers. Reservoirs can come in one of two "flavors": rate encoding real-valued reservoirs, Echo State Networks (ESN) [22], and spike or event-encoding Liquid State Machines (LSM) [29].

1.3.2 Learning with Random RNNs

One can view the reservoir as acting as a temporal kernel [47], expanding out a temporal sequence into a high dimensional random space that makes it more separable (linearly or otherwise). This temporal kernel works well even though the weights are kept random, which is similar to theories of the brain that are built off of unsupervised organizations of neuronal groups [12]. Once the reservoir expands out the data, it is traditional to use methods such as ridge or lasso regression to learn to map reservoir states to targets. This work replaces such a readout with a deep-network backpropagation mechanism. Reservoir computing has been shown to train faster and use less training data than their fully-differentiable counterparts in some domains [34]. For real world tasks, ESNs have been successfully used for a variety of tasks ranging from direct robotic control [37], electric load forecasting [4], wireless communication [23], image classification [48], robot navigation [1], reinforcement learning [8], and recently action recognition using preprocessed inputs [46].

In introducing ESNs, Jaeger [22] wrote that a potential limiting factor of using ridge regression for ESNs is that it is only a linearly separable readout, and that this might be limiting the learning capabilities of the system. This motivates an investigation into whether using readout mechanisms that can learn non-linear mappings can improve learning. In this work, we will show that using a readout capable of learning non-linear mappings in conjunction with a parallel structure greatly improves classification and regression accuracy.

One way to view reservoir computers is through the physics paradigm, where the network is learning a dynamical attractor state (cite Herbert 2004). From a neuroscience perspective, however, the reservoir can be seen through the lens of the "assembly readout mechanism" paradigm [6], where the reservoir is producing dynamical features, and the readout mechanism is interpreting them.

1.4 Contributions

This work is the first work, to the best of the authors' knowledge, that combines a novel shared-weight parallel ring reservoir scheme, a backpropagated readout mechanism, and modern normalization techniques, and compares it with a state of the art fully differentiable stacked LSTM and GRU, on two real-world learning tasks. It is also the first work to perform rigorous testing on the efficacy of the backpropagated mechanism versus traditional ridge regression mechanisms. Our hope is to bring together advances in reservoir computing from the physics community, as well as advances from the deep learning community in machine learning, and combine them to greater effect.

Our contributions to the current state of the art are:

- Use of a parallel reservoir mechanism in conjunction with a backpropagated deep feedforward network readout, what we call a Hybrid Backpropagation Parallel Echo State Network (HBP-ESN) and a variant with a shared interconnection weight matrix (HBP-ESN M-Ring)
- Testing of the HBP-ESN and HBP-ESN M-Ring on two challenging and different real-world datasets, a gesture recognition classification task and EEG signal emotion regression task
- Analysis and use of modern normalization techniques such as weight decay and batch normalization
- Experimentally demonstrate that the network can significantly outperform an LSTM and GRU

2 HBP-ESN Architecture

2.1 Reservoir Architecture

The most standard way to implement a reservoir computer is to attach a set (vector) of recurrent, randomly weighted neurons that remain fixed and unlearned, to a readout mechanism such as ridge regression, or sometimes, stochastic gradient descent [cite]. Here, we use the nonlinear leaky updates from [48]. At each timestep t , input is fed into the reservoir through a weight matrix W_{in} , and the reservoir and output vectors are updated according to the following equations:

$$x_t = (1 - \alpha)x_{t-1} + \alpha f(u_t W_{in} + x_{t-1} W_{rec}) \quad (1)$$

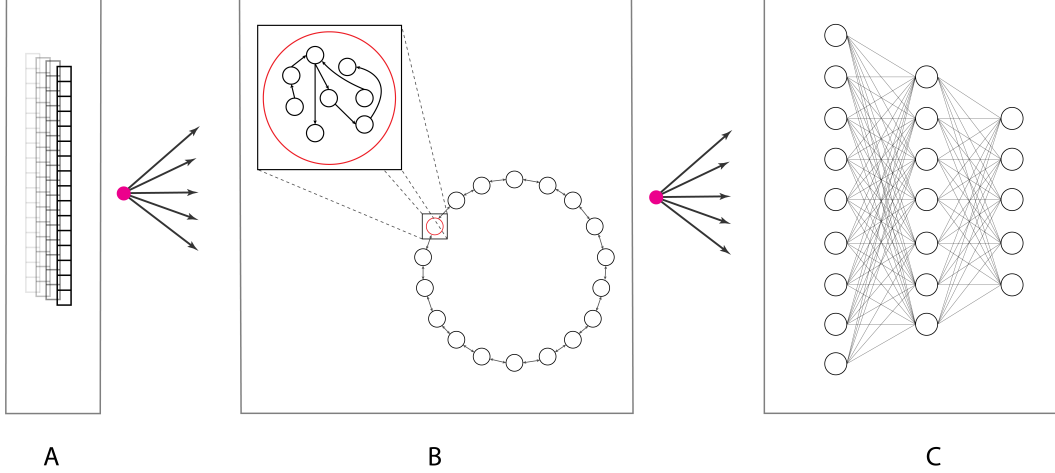


Figure 1: The HBP-ESN M-Ring architecture: (A) the input to the reservoir, with shading corresponding to time steps; (B) the parallel ring reservoir; (C) the backpropagated readout. The magenta dots and fan-out arrows between A, B, and C represent fully connected matrices. The ring in B is straightened and then flattened before being passed into C. Each sub-reservoir, shown zoomed in, is connected to its left and right neighbor in the shared weights structure, forming a ring of sub-reservoirs. In different experiments, some elements are swapped; shown here is the full contribution with the shared parallel cross-talk weights. HBP-ESN without M-Ring lacks the ring links. In the single reservoir experiment, part B consists solely of a large version of the zoomed-in sub reservoir. For the feed-forward only experiment, part B is removed entirely.

$$y_t = x_t W_{out}, \quad (2)$$

where y_t is the output vector at time t , x_t is the N -dimensional vector of the states of each reservoir neuron at time t , u_t is the input vector at time t , α is the leak rate, $f(\cdot)$ is a saturating non-linear activation function (here hyperbolic tangent), and W_{in} , W_{rec} , W_{out} are the input-to-reservoir, reservoir-to-reservoir, and reservoir-to-output weight matrices, respectively. If the input vector has length A , and the output vector has length B , then the dimensions of W_{in} , W_{rec} , W_{out} are $N \times A$, $N \times N$, $B \times N$ respectively. The ordering of the weight matrix multiplication is reversed based on whether one is using row-major or column-major weight ordering. From the updates, one can see that the reservoir keeps a memory trace of its past activity through the recurrent weights W_{in} , which is a function of the leak rate α as well as the spectral radius of W_{in} [cite]. In order to make sure inputs vanish over time, it is necessary to ensure the *echo state property* of the network, which says that the spectral radius, or largest singular value of the recurrent matrix, is less than one. Although this property has been re-analyzed [55], for simplicity we will use the original recipe for calculating the desired spectral radius as in [22]. For simplicity of the model and ease of replication, we also do not hyper-tune the reservoir weights using any optimization techniques such as genetic algorithms [14, 15] or particle swarm optimization [2, 10], although those can readily be applied to this work to decrease the variance of the reservoir trials.

The paradigm of reservoir computing simply prescribes for three elements: a read-in of the input, a reservoir, and a read-out that learns the states of the reservoir. It is the hope, but not guaranteed, that the reservoir will act as a perfect temporal kernel, expanding the input into a higher dimensional space that is *linearly* separable by the readout mechanism. However, often this may in fact not be the case for all possible inputs. One way to show this is to compare the performance of a linear learning method versus a non-linear method, such as a deep neural network, which is what we attempt to do here. We want to tease apart the three components into their own modules; for the purposes of this current work, we swap out the readout module.

2.2 Parallel Reservoir Structure

2.2.1 Motivations

As in [36], we implement a parallel reservoir structure. This structure serves a number of purposes. Biologically, the brain is partitioned into multiple sub-regions, which aids in its functions, for example see [24, 43]. On the other hand, there exist networks with massive cross-connectionism such as in Hopfield networks [30], which have the issue of what is known as "catastrophic interference" [17], wherein learning new information causes old information to be lost. Of course, we are not learning the weights of the recurrent network and thus cross-contamination is less of an issue, however we take this simply as motivation. By partitioning the networks, we prevent information from over-saturating other regions and enforce some sub-structural components. This catastrophic forgetting occurs not just in Hopfield networks, but in feed-forward MLP networks as well. There is empirical evidence that partitioning a network can alleviate this catastrophic forgetting [33]. A biologically-inspired reservoir network was shown to improve its capabilities by partitioning activity using gating [39].

Second, by separating into smaller sub-reservoirs, our hope is that we can decrease the exponential increase of the size of one large reservoir. As our experiments show and as in [36], multiple small reservoirs perform well in the face of an increase of the size of the input. Third, multiple reservoirs are better adept at handling data that is naturally already partitioned. In our experimental data, the vector is already partitioned into (x,y) components, with each two indexes comprising one (x,y) position. This concept can be extended to any case where the data naturally partitions itself, such as regions of an image. Due to the shared connectivity matrix described in section 2.2.3, the sub-reservoirs are able to access information about other regions as well.

2.2.2 Implementation

Equation 1 becomes, for the parallelized scheme without the cross-talk matrix:

$$x_{t,r} = (1 - \alpha)x_{t-1,r} + \alpha f(u_t W_{in,r} + x_{t-1,r} W_{rec,r}) \quad (3)$$

and for the shared cross talk matrix:

$$\delta_{ring} = \beta(f(x_{t-1,r-1} W_{shared}) + f(x_{t-1,r+1} W_{shared})) \quad (4)$$

$$x_{t,r} = (1 - \alpha)x_{t-1,r} + \alpha f(u_t W_{in,r} + x_{t-1,r} W_{rec,r}) + \delta_{ring} \quad (5)$$

Equation 2 becomes, for both cases:

$$y_t = [x_{t,1}; x_{t,2}; \dots; x_{t,R}] W_{out} \quad (6)$$

for all sub-reservoirs r from 1 to R , where R is the number of sub-reservoirs, and ';' is the concatenation operator in eq. 4. All sub-reservoirs are stored in an R dimensional array, referred to henceforth as the sub-reservoir array. For equation 4, reservoir indexes -1 and +1 reflect the left and right index in the sub-reservoir array. The sub-reservoir connections are wrapped around, forming a ring structure. β is a parameter chosen to reflect how strongly to weight the cross-talk connections. W_{shared} is the shared connectivity matrix used for all sub-reservoirs, which saves on memory usage. We use the same alpha for each sub-reservoir, although this can be parameterized as well.

A significant benefit of the parallel reservoir structure is a dramatic decrease in the memory usage of the weight matrix, which alleviates the N^2 increase in the size of the weight matrix for the recurrent weights. As an example, for a large single-reservoir of 3200 neurons, we would have 3200^2 recurrent weights, on the order of 10^7 . However, for the same number of total neurons organized into 8 sub-reservoirs with 400 neurons each, this only amounts to 4×400^2 , or on the order of 10^6 , a full order of magnitude smaller. This would allow for large scaling of network sizes. The use of a shared cross-talk matrix, W_{shared} , further cuts down on memory usage: this matrix needs to be allocated only once, and has constant memory with respect to the number of sub-reservoirs.

2.2.3 Shared Cross-Talk Matrix and the Reservoir Ring

In order to allow each sub-reservoir to exchange information, we introduce a novel meta-ring structure (M-Ring), wherein each sub-reservoir uses a shared weight matrix to connect to its left and right neighbors (Fig 1). Sub-reservoirs at the end wrap around, creating a ring structure, which resembles the internal ring structure of simple RC networks [40].

2.3 Deep Network Readout

A centerpiece of our architecture is that unlike "traditional" RCs, the readout is done using a deep network using backpropagation [41], which does credit assignment through the layers of the network via the chain rule. While this does increase the computational complexity versus ridge regression, we argue that it is minimal; the backpropagation still stops before the recurrent layer, and does not need to be propagated backwards through time. The multiple parallel reservoirs are concatenated at the end, and fed in as a single layer to the input layer of the deep network.

2.4 Related Work

There have been attempts to replace the standard ridge regression in echo state networks with alternatives [[37]]. There also have been efforts to apply CNN techniques to the reservoir readout [[28]]. Recent work has attempted to create hybrid models that integrate feed-forward networks into the reservoir structure [56].

Other earlier works have attempted to attach a multi-layered backpropagation learning structure to an ESNs [5, 54], however neither of these works used batch normalization or a parallel scheme. One of the roadblocks hampering the use of reservoirs for real-world tasks is that the performance of the reservoir diminishes when the input dimension begins to increase: thus most test datasets used for reservoir computing, even very recently, involve very low-dimensional data. Recently, it was shown in [36, 38] that this issue can be alleviated by introducing a parallel reservoir schema, where the work is divided up among multiple reservoirs in parallel. Our work extends this architecture for use with non-linear readouts and real-world data. The idea of using reservoirs in parallel can be seen as an extension and fusion of the ideas of deep feed-forward multilayer perceptrons, where each neuron is replaced by a reservoir. ESNs have also very recently been applied to EEG learning tasks [25].

3 Experiments

For consistency and reproducibility, we provide for all experiments, for each network type the results as the standard deviation and mean from multiple runs. For all experiments, we test against the current state-of-the-art recurrent neural network gradient descent networks. The two gradient-descent RNNs that we focus on are LSTMs [19] and GRUs [9].

To further stress test the comparisons, we also compare against deep (stacked) versions of LSTMs and GRUs with three recurrent stacked layers. We performed an exhaustive hyper-parameter search for the LSTM and GRUs to find the best parameters in each experiment.

As ablation studies for the ring and parallel architectures, we include three variants of the HBP-ESN: 1) HBP-ESN, the parallel architecture without ring interconnections, 2) HBP-ESN M-Ring, the same but with the meta-ring interconnections as shown in Fig. 1, and 3) HBP-ESN Single, a non-parallel variant with one large reservoir. We also include for both experiments a fully connected deep ANN (FCNN) as a baseline to demonstrate the necessity for a recurrent network for these tasks. We split all data 80/20 for train/test.

For our first experiment, we run a label classification on the ChaLearn [52] dataset, which extracts 8 upper body skeleton keypoints using OpenPose [7] for the Helicopter Aircraft Marshaling [16]. We train our networks to learn classify these gestures into the 9 labels as shown in Figure 3.

For both experiments, we found that batch norm performed better on the HBP-ESN in the fully-connected classification layers on these two datasets than for the LSTM or GRU. Batch normalization was applied on the LSTM and GRU recurrent layers for those networks, but not the fully-connected readout layers, and batch norm was applied in the fully-connected readout layers of the HBP-ESNs, as these were the best performing network configurations for all networks.

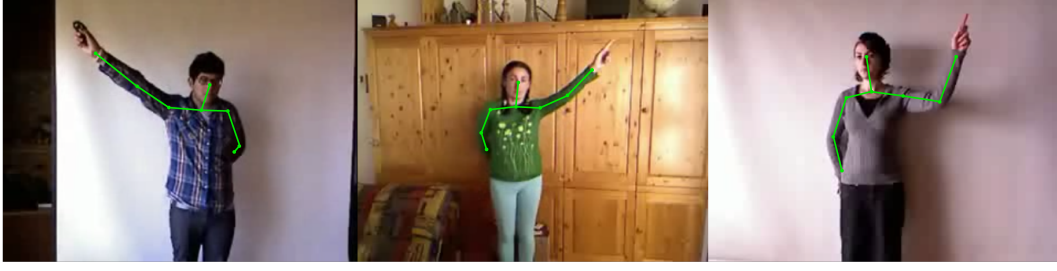


Figure 2: ChaLearn Dataset: Helicopter Airmarshal Subset with 8 upper body skeleton keypoints acquired from OpenPose containing 1792 samples - shown is label 3 (Take off) in Figure 3. These examples demonstrate variations due to camera angles, body types, individual styles, hands used, etc. that make the gestures difficult to learn.

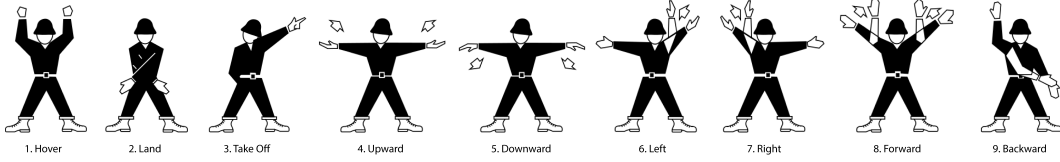


Figure 3: Helicopter Airmarshal labels, a total of 9 classes, present in the ChaLearn dataset

We can see from Table 1 that all variants of the HBP-ESN outperform the LSTM and GRU networks. The sub-reservoir network was able to achieve comparable and even superior accuracy to the full reservoir, even though it uses an order of magnitude smaller weight matrices (Table 1). We see that the M-Ring outperforms the vanilla parallel architecture as well. We found that for a problem as small as 16-dimensions, the backpropagation deep readout was able to make up for any shortcomings in separability issues as described in [36]. However, the size of the reservoir made the training time much longer, and increasing the size of the reservoir further was not feasible in memory with one reservoir. The parallel reservoir structure, however, performed as well as the single-reservoir with backpropagation. All parallel versions used 8 sub-reservoirs. The best HBP-ESN results came from sub-reservoirs of size 300 units, and from the M-Ring, size 400 units. For all gesture HBP-ESN experiments, for the reservoir a spectral radius was set to .1 according to the method of [22], α was 0.05. For the backpropagation readout, a learning rate of 0.001 was used, weight decay was 0.001, and optimized with SGD with momentum 0.01 and ReLU activations. Batch size 64 was found to be the optimal, given the small dataset size. For the reservoir, because it keeps a memory of past activity, only every 5th input frame was used to save on computation costs. β was set to 0.005 for both experiments.

Classification Methods	acc (%) $\pm$ sd
FCNN Baseline	70.714 $\pm$ 2.519
LSTM 1-Layer	74.551 $\pm$ 2.542
LSTM Deep	73.770 $\pm$ 3.293
GRU 1-Layer	73.902 $\pm$ 3.282
GRU Deep	78.415 $\pm$ 2.364
HBP-ESN Single	84.007 $\pm$ 0.853
HBP-ESN	83.763 $\pm$ 0.994
HBP-ESN M-Ring	84.059 $\pm$ 1.090

Table 1: Airmarshal Gesture Classification Results. Deep LSTM and GRU networks used 3 stacked recurrent layers. FCNN is the deep feed-forward ANN baseline. HBP-ESN is the parallel scheme without interconnections; HBP-ESN Single is one large reservoir, and HBP-ESN M-Ring is the parallel structure with ring-connected components as shown in Fig. 1. Each number is the mean over 20 runs, with std. deviation posted as $\pm$.

LSTM and GRU networks for 1-layer and deep versions had the best results for 256 units in the RNN layer(s) and 256 in the fully-connected 1-layer, and half of 256 in the subsequent layers for the deep version. The dropout was 0.5 in the input and 0.1 in the hidden layers both for RNN and fully-connected layers. As activation tanh was used for the RNN and ReLU for the fully-connected, no batch normalization, a learning rate of 0.001, the Adam optimizer with beta values of 0.9 and 0.999 and epsilon of 1e-07, and a batch size of 64 were used, which ran 20 times for all explorations.

For our second experiment, we run a regression learning on the DEAP EEG dataset [26] (Fig. 4). We use all 40 channels for training corresponding to the raw sensor recordings from 32 EEG electrodes and 8 body physiological sensors. The dataset has recordings from 32 participants, each with 40 trials, watching 1-minute video clips, where EEG and bio-sensor recordings were taken during the video watching session. Afterwards, participants were asked to rate emotions (valence, arousal, dominance, liking) on a 9-point scale. As in [35], we break up each 60-second trial into 1-second epochs, corresponding to 134-length vectors; each 1 second epoch is given the same label as the corresponding 60 second trial, giving 2400 samples per participant. Unlike most papers that run classification on high/low values, we run pure regression on the numerical label value and report results after running the network 5 times and showing the mean and std. dev. in Table 2. Lastly, shown here are the regression results for "arousal", corresponding to the second index of the label array. Other labels and combined regression results will be shown in supplementary materials. All input data was channel-wise normalized between -1 and 1. Thus at each of 134 time steps, a 40-element vector was fed into the networks corresponding to the 40 channels, and the network output was regressed to the arousal label. For all HBP-ESN runs for Experiment 2, 140 total reservoir neurons were used: 140 for the single reservoir, and 4 sub-reservoirs of size 40 were used for the parallel schemes. As Table 2 shows, the M-Ring variant outperformed all other networks in terms of both accuracy and std. deviation.

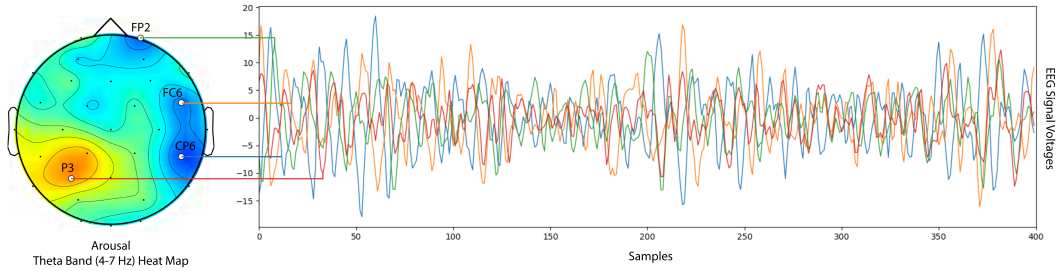


Figure 4: Left: A heatmap depicting the EEG nodes overlaid with the correlations in activity that correspond to the 'arousal' value. Red values indicate positive correlation and blue negative. For our purposes, we use all channels and let the network discern which channels are of importance.[26] Right: The raw EEG signal used for training, with lines showing the corresponding EEG node locations for example channels.

Regression Methods	mse $\pm$ sd
FCNN	2.402 $\pm$ 0.164
LSTM Deep	1.794 $\pm$ 0.307
GRU Deep	3.100 $\pm$ 0.792
HBP-ESN Single	1.320 $\pm$ 0.105
HBP-ESN	1.457 $\pm$ 0.119
HBP-ESN M-Ring	1.433 $\pm$ 0.122

Table 2: Results from the EEG emotion regression learning task for the 'arousal' label. Each network was ran individually on all 32 participants 5 separate times, and mean and std. deviation across all runs are posted.

At each layer of the deep learning readout, Batch Normalization [20] is applied, which regularizes each mini-batch. The batch size we use for our experiments is 64, found to be the best from empirical testing. Here we apply weight decay [45] as our main normalization technique, as it can empirically

outperform dropout [11], another common deep learning normalization technique. To save on memory usage, only every other reservoir state is fed into the reservoir, so the number of reservoir states is downsampled by two. For the LSTM and GRU network comparisons, dropout is used instead, which was found more effective for these networks.

4 Conclusion

We demonstrated the use of a backpropagation hybrid mechanism for parallel reservoir computing with a meta ring structure and its application on a real-world gesture recognition dataset. We show that it can be used as an alternative to state of the art recurrent neural networks, LSTMs and GRUs. Future work will expand upon and further integrate deep learning techniques into the reservoir learning framework. We offer this novel network as a new route to learning temporally changing or sequential data in a way that utilizes random recurrent matrices without the use of Backpropagation Through Time. We believe this can form the building blocks for future architectures that can modularize the reservoirs for hierarchical systems.

Acknowledgments and Disclosure of Funding

This work was supported by the University of Maryland COMBINE program NSF award DGE-1632976 and NSF grant BCS 1824198. The authors would like to also thank Vaishnavi Patil at the University of Maryland for thoughtful conversation and advice.

5 Broader Impacts

We believe that bio- and brain-inspired neural networks have the potential to both improve the state of the art of A.I., but also to give insights into how the brain operates. While much of the brain remains a mystery, even potentially non-biological advances in AI, which are inspired by neuroscience can have big impacts on how researchers orient their theories of the brain. For example, while we are not arguing for or against the plausibility of backpropagation, the idea of error reduction from backpropagation networks has led to new theories about the central role of error reduction in the brain [32]. Our hope is that by proposing biologically inspired networks, such as our hybrid network here that avoids mechanisms that are generally agreed upon as biologically implausible like BPTT [31], we can add further evidence for ways the brain can process temporal information. In addition, by testing on EEG datasets, we also aim to show that our work can be of broader use to the medical community in identifying brain patterns for a wide variety of uses. One example of a future direction we hope to take this work is for anomaly detection from brain signals to help identify early signs of epileptic seizures.

References

- [1] E. A. Antonelo and B. Schrauwen. On learning navigation behaviors for small mobile robots with reservoir computing architectures. *IEEE transactions on neural networks and learning systems*, 26(4):763–780, 2014.
- [2] S. Basterrech, E. Alba, and V. Snášel. An experimental analysis of the echo state network initialization using the particle swarm optimization. In *2014 Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC 2014)*, pages 214–219. IEEE, 2014.
- [3] G. Bellec, F. Scherr, D. Salaj, E. Hajek, R. Legenstein, and W. Maass. Biologically inspired alternatives to backpropagation through time for learning in recurrent neural networks.
- [4] F. M. Bianchi, E. De Santis, A. Rizzi, and A. Sadeghian. Short-term electric load forecasting using echo state networks and pca decomposition. *Ieee Access*, 3:1931–1943, 2015.
- [5] F. M. Bianchi, S. Scardapane, S. Løkse, and R. Jenssen. Bidirectional deep-readout echo state networks. *arXiv preprint arXiv:1711.06509*, 2017.
- [6] G. Buzsáki. Neural syntax: cell assemblies, synapsembles, and readers. *Neuron*, 68(3):362–385, 2010.

- [7] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.
- [8] H. Chang and K. Futagami. Convolutional reservoir computing for world models. *arXiv preprint arXiv:1907.08040*, 2019.
- [9] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [10] N. Chouikhi, B. Ammar, N. Rokbani, and A. M. Alimi. Pso-based analysis of echo state network parameters for time series forecasting. *Applied Soft Computing*, 55:211–225, 2017.
- [11] G. E. Dahl, T. N. Sainath, and G. E. Hinton. Improving deep neural networks for lvc sr using rectified linear units and dropout. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8609–8613. IEEE, 2013.
- [12] G. M. Edelman. Neural darwinism: selection and reentrant signaling in higher brain function. *Neuron*, 10(2):115–125, 1993.
- [13] P. Enel, E. Procyk, R. Quilodran, and P. F. Dominey. Reservoir computing properties of neural dynamics in prefrontal cortex. *PLoS computational biology*, 12(6), 2016.
- [14] A. A. Ferreira and T. B. Ludermir. Genetic algorithm for reservoir computing optimization. In *2009 International Joint Conference on Neural Networks*, pages 811–815. IEEE, 2009.
- [15] A. A. Ferreira and T. B. Ludermir. Comparing evolutionary methods for reservoir computing pre-training. In *The 2011 International Joint Conference on Neural Networks*, pages 283–290. IEEE, 2011.
- [16] W. Foundation. *Aircraft Marshalling*, 2019.
- [17] R. M. French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.
- [18] S. Ghosh-Dastidar and H. Adeli. Spiking neural networks. *International journal of neural systems*, 19(04):295–308, 2009.
- [19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [20] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
- [21] E. M. Izhikevich, J. A. Gally, and G. M. Edelman. Spike-timing dynamics of neuronal groups. *Cerebral cortex*, 14(8):933–944, 2004.
- [22] H. Jaeger. The “echo state” approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34):13, 2010.
- [23] H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *science*, 304(5667):78–80, 2004.
- [24] J. Jensen, A. J. Smith, M. Willeit, A. P. Crawley, D. J. Mikulis, I. Vitcu, and S. Kapur. Separate brain regions code for salience vs. valence during reward prediction in humans. *Human brain mapping*, 28(4):294–302, 2007.
- [25] D.-H. Jeong and J. Jeong. In-ear eeg based attention state classification using echo state network. *Brain Sciences*, 10(6):321, 2020.
- [26] S. Koelstra, C. Muhl, M. Soleymani, J.-S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, and I. Patras. Deap: A database for emotion analysis; using physiological signals. *IEEE transactions on affective computing*, 3(1):18–31, 2011.
- [27] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.
- [28] Q. Ma, E. Chen, Z. Lin, J. Yan, Z. Yu, and W. W. Ng. Convolutional multitimescale echo state network. *IEEE Transactions on Cybernetics*, 2019.
- [29] W. Maass. Liquid state machines: motivation, theory, and applications. In *Computability in context: computation and logic in the real world*, pages 275–296. World Scientific, 2011.

- [30] D. J. MacKay and D. J. Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.
- [31] L. Manneschi and E. Vasilaki. An alternative to backpropagation through time. *Nature Machine Intelligence*, 2(3):155–156, 2020.
- [32] A. H. Marblestone, G. Wayne, and K. P. Kording. Toward an integration of deep learning and neuroscience. *Frontiers in computational neuroscience*, 10:94, 2016.
- [33] N. Y. Masse, G. D. Grant, and D. J. Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115(44):E10467–E10475, 2018.
- [34] G. Neofotistos, M. Mattheakis, G. D. Barmparis, J. Hizanidis, G. P. Tsironis, and E. Kaxiras. Machine learning with observers predicts complex spatiotemporal behavior. *Frontiers in Physics*, 7:24, 2019.
- [35] C. Pan, C. Shi, H. Mu, J. Li, and X. Gao. Eeg-based emotion recognition using logistic regression with gaussian kernel and laplacian prior and investigation of critical frequency bands. *Applied Sciences*, 10(5):1619, 2020.
- [36] 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*, 120(2):024102, 2018.
- [37] A. Polydoros, L. Nalpantidis, and V. Krüger. Advantages and limitations of reservoir computing on model learning for robot control. In *IROS Workshop on Machine Learning in Planning and Control of Robot Motion, Hamburg, Germany*, 2015.
- [38] J. Qiao, F. Li, H. Han, and W. Li. Growing echo-state network with multiple subreservoirs. *IEEE transactions on neural networks and learning systems*, 28(2):391–404, 2016.
- [39] R. V. Rikhye, A. Gilra, and M. M. Halassa. Thalamic regulation of switching between cortical representations enables cognitive flexibility. *Nature neuroscience*, 21(12):1753–1763, 2018.
- [40] A. Rodan and P. Tino. Minimum complexity echo state network. *IEEE transactions on neural networks*, 22(1):131–144, 2010.
- [41] D. E. Rumelhart, R. Durbin, R. Golden, and Y. Chauvin. Backpropagation: The basic theory. *Backpropagation: Theory, architectures and applications*, pages 1–34, 1995.
- [42] J. Russin, R. C. O’Reilly, and Y. Bengio. Deep learning needs a prefrontal cortex.
- [43] B. Rypma and M. D’Esposito. The roles of prefrontal brain regions in components of working memory: effects of memory load and individual differences. *Proceedings of the National Academy of Sciences*, 96(11):6558–6563, 1999.
- [44] W. Singer. Cortical dynamics revisited. *Trends in cognitive sciences*, 17(12):616–626, 2013.
- [45] L. N. Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018.
- [46] N. Soures and D. Kudithipudi. Deep liquid state machines with neural plasticity for video activity recognition. *Frontiers in neuroscience*, 13:686, 2019.
- [47] P. Tino. Dynamical systems as temporal feature spaces. *Journal of Machine Learning Research*, 21(44):1–42, 2020.
- [48] Z. Tong and G. Tanaka. Reservoir computing with untrained convolutional neural networks for image recognition. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1289–1294. IEEE, 2018.
- [49] R. S. van Bergen and N. Kriegeskorte. Going in circles is the way forward: the role of recurrence in visual inference. *arXiv preprint arXiv:2003.12128*, 2020.
- [50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [51] F. Walter, F. Röhrbein, and A. Knoll. Computation by time. *Neural Processing Letters*, 44(1):103–124, 2016.

- [52] J. Wan, Y. Zhao, S. Zhou, I. Guyon, S. Escalera, and S. Z. Li. Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 56–64, 2016.
- [53] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.
- [54] A. Woodward and T. Ikegami. A reservoir computing approach to image classification using coupled echo state and back-propagation neural networks. In *International conference image and vision computing, Auckland, New Zealand*, pages 543–458, 2011.
- [55] I. B. Yildiz, H. Jaeger, and S. J. Kiebel. Re-visiting the echo state property. *Neural networks*, 35:1–9, 2012.
- [56] Z. Zhao, J. Qin, Z. He, H. Li, Y. Yang, and R. Zhang. Combining forward with recurrent neural networks for hourly air quality prediction in northwest of china. *Environmental Science and Pollution Research International*, 2020.