Coherency in Representational Space

For Spiking Reservoir Computers

-

Ryan O'Loughlin, Marleen Schippers, Jelmer Borst, Guillaume Pourcel, Thomas Tiotto, Steve Abreu
An Informal Progress Report on a Master's Thesis from The University of Groningen

***Abstract*—As promise of the next hardware revolution looms, attending to unconventional computing paradigms becomes more important than ever. Recurrent spiking neural networks (RSNNs) are highly suitable for neuromorphic hardware and come ready with an actionable reservoir computing scheme known as the Liquid State Machine (LSM). An LSM employs an RSNN "reservoir" to process input such that meaningful features can be learned with a simple linear readout map trained on reservoir states. LSMs have garnered success in a number of domains, such as speech, image, and even video recognition. However, the hyper-parameter space for reservoir design is prohibitively large and all of these above mentioned successes make use of specialized LSMs, often with significant design-and-tuning overhead. Therefore, a pertinent task for the advancement of LSM literature is the development of an increasingly generic playbook for good reservoir design. To this end, we here survey LSMs in their many forms, both prominent and novel, with particular attention to learning rules and topologies. We characterize reservoir quality as a balance between diverse computational efficacy and coherent dynamics in representational state space. In doing so, we aim to distill generically favorable reservoir attributes and therefore better equip the state-of-the-art for LSMs that may exhibit the sort of robust versatility we find in general intelligence.**

I. Introduction - Liquid State Machines

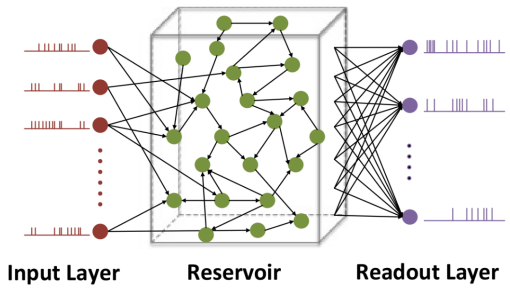

Fig. 1. All three layers of typical LSM, diagram borrowed from []

Complexity abounds in natural and artificial systems. One need only consider cellular automata to realize the explosive tendency toward complexity in dynamical systems. Reservoir computing (RC) manages to exploit highly complex non-linear dynamics without the need to explicitly understand or control them. The two principle brands of RC are *Echo State Networks* [2] and *Liquid State Machines* [1]. Both employ recurrent neural networks–the prior with sigmoidal activation functions and the latter with neural spikes– as reservoirs. While the mathematical elegance of continuity is lacking in the spiking approach, certain hardware opportunities may be gained [3]–[8]. Therefore, it is a productive task to better illuminate generically good properties of these spiking reservoirs and we will thus here focus our attention here on this latter brand, which takes the form of Fig. 1.

In brief, arbitrary input may be fed to the system in the form of neural spikes, which are then processed by the reservoir as it is perturbed from its resting state. These perturbations carry information that is salient in different (and hopefully better) ways than what was obvious in the original input and therefore a mapping of these "liquid states" may be learned from the reservoir to some new characterization, such as classification or prediction. Supervised learning only occurs in the readout layer via simple regression. This deceptively simple paradigm carries surprising computational force, achieving state-of-the art performance on canonical machine learning tasks such as speech recognition [9]–[12], image classification [**?**], [12], and even *video event recognition/prediction* [13], [14]. However, all of the above mentioned papers employ different reservoir schemes and readout mechanisms. The combinatoric explosion of possible architectures (when considering the nuances of learning rules, topologies, hyperparameters, and hierarchies) is gigantic. Even the modest survey conducted presented here has permuted through thousands of configurations. The question then becomes, how to best select for 'good' LSM architectures? We here attempt to distill this inquiry to *how to best select for good liquid state reservoirs?*

For reservoir analysis, this survey considers a number of design parameters, including sparsity, excitatory/inhibitory behaviors, topologies, and learning rules. These latter two are given great attention, and novel versions are explored. To asses overall LSM efficacy, performance on the Heidelberg speech recognition dataset [15] is evaluated in conjunction with state-space dynamics. State-space is analyzed in terms of its representational quality, using principle component analysis (PCA) and Euclidean distance measures. Quality here is defined in-part by the original metrics of [1]–*the separation property* (expansion of input distances) and *the approximation property* (suitability for linear readout mapping)–the prior of which is expanded upon in a novel proposed metric we here call *coherent separation*.

However, because inputs, reservoirs, and readout maps are inextricably entangled in the LSM paradigm, we must first preface the role of each before digging into the reservoir dynamics.

II. Inputs

The diversity of potential input for liquid state machines is arbitrarily large. Considering that the entire phenomenal experience of a human being is encoded into spikes, we can be sure it is a robust medium for input of information. Because our survey here is concerned with reservoir analysis and not the more engineering-orientated task of impressive machine learning results, and because of computational resource limitations (i.e. working on a laptop) two simple input classification tasks are selected–one abstract and the other real-world.

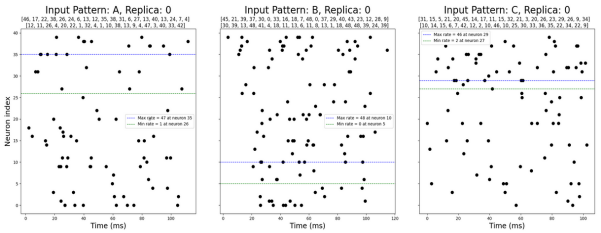

Fig. 2. Example of a figure caption.

Poisson pattern classification is adapted from [1] as a task of arbitrary difficulty. Some number of input channels are each assigned a random number, defining the rate of that channel's Poisson Spike trains, which are simply spiking events that occur in accordance to a Poisson distribution. The task can be made more or less challenging by the number of channels, and spatial grouping (similar spiking rates for nearby channels). Replicas (for training and testing) can be generated by adding Gaussian noise to the original examples.

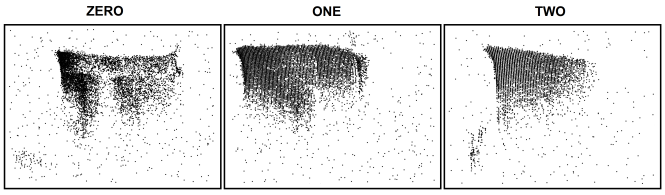

Fig. 3. Cochlear implant recordings of spoken digits "zero", "one", and "two" from [].

To be sure our abstract results carry over to the real world, we further test on the Heidelberg Spoken Digit Dataset [**?**] where spoken digits are recorded via a cochlear implant in a spiking form similar to how the brain might receive audio information. Of course, any sound could in theory be preprocessed to adopt a spiking form [**?**], such as done in [11].

III. Reservoirs

Before delving into the variety of available reservoir design decisions, it is worth mentioning the two key properties defined in original LSM paper [1], which still stand as useful metrics today and one of which the present research attempts to expand upon (refine?).

- ***The Separation Property*** defines the degree to which differences in input are expanded in representational space. This is to say that characteristic input features should ideally become more salient in liquid state responses. This is accomplished two-fold by changing (usually increasing) the dimensionality of the input, and then propagating it through the non-linear dynamics of the reservoir. Both of these mechanisms should in theory "magnify" input features. A good reservoir ought to always expand differences in input examples, and moreover expand greater differences more [1].
- ***The Approximation Property*** refers to the resolution with which liquid states can be interpreted. Remember that the classification for reservoir computing is often accomplished by no more than a linear readout map trained on liquid states. Therefore, it is essential that these liquid states be "legible."

The authors here are concerned more with the separation property than the approximation property. In fact, a generous readout map is be designed to give the liquid states the benefit of the doubt in terms of performance, so that better attention may be leveraged onto the internal representations of the reservoirs in terms of their ideal performance. This is discussed in more detail in section –.

How to design a generically good reservoir in terms of these two properties is a well studied, and yet still very open, question. Among the vast arsenal of reservoir design decisions, it is perhaps most straightforward to speak in terms of reservoir *structure* and reservoir *rules*. Here we speak of *topologies* and *learning* respectively.

A. Topologies

The connective architecture of a network defines its topology. There are a number of common methods for generating certain types of connectivity, and within each of these methods there exist a high number of unique possible networks on the order of $2^{n(n-1)/2}$ []. Luckily, in reservoirs, random connectivity already performs well [], but there is still much research to be done in terms of optimizing reservoir dynamics as a function of topology. Our survey here is by no means exhaustive, but nonetheless samples what may be the most notable and common archetypes of network structure, which may help to illuminate in what direction among infinite graphs should attention be focused.

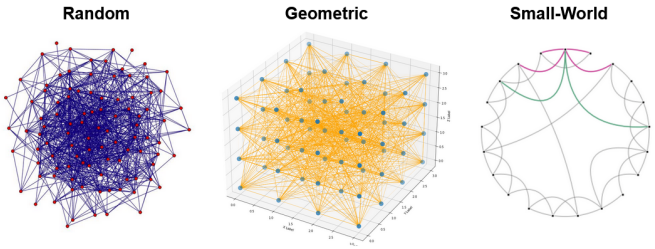

Fig. 4. Example of a figure caption.

1) Random Topology: Random graphs (Fig. 8) essentially sample from all possible graphs with a uniform distribution

defined by $p^m(1-p)^{N-m}$ where $N = \binom{n}{2}$. Here they are implemented such that any two nodes (neurons in our case) have some probability p of being connected. This p value therefore determines the internal sparsity of a reservoir simply by virtue of it defining the the frequency with which connections are made. This flavor or random graph generation is that of [**?**].

In terms of using random graphs to determine synaptic connectivity for reservoirs, it should noted this approach results has no particular adherence to spatial aspects of input. This is to say that the spatial-temporal property become simply temporal properties in a random reservoir. Adjacency of features in input channels are dispersed randomly across the graph and do not remain grouped in their original order.

2) Geometric Topology: Unlike random graphs, geometric topologies maintain a certain amount of spatio-topic coherence to the input. This is accomplished by first generating a geometric structure of nodes in euclidean space, say a cube or the canonical $15 \times 3 \times 3$ neo-cortical column, and creating edges between nodes on the basis of an exponential decay as a function of distance, such that the probability of any to neurons being connected is defined by

$$P(a,b) = C \cdot \exp^{-(\frac{D(a,b)}{\lambda})^2}$$

where C is simply a connectivity coefficient, $D(a,b)$ the euclidean distance between a and b, and λ the connectivity parameter defining the typical distance and frequency of connected neurons. Such is the topological approach in LSMs founding paper [1].

It is worth noting, that the author has found geometric topologies to be *the* go to choice for LSM literature. Ostensibly, this is because of the biological plausibility of this sort of structure in the cortical column []. However, it is surprising that much of literature continues to employ this method without explicit justification despite research directives often not being that of biological plausibility, but rather of machine learning performance. It is therefore here suggested that alternate network generation methods ought to be more rigorously considered. Moreover, it is clear that this spatio-topic method constrains information flow locally in terms of neuron location. For certain types of information processing, it is imaginable that one would want some ratio of information (here we talk of spikes) to reach across the network at length, thereby distributing essential features across a greater network area.

3) Small-World Topology: Whereas random graphs implement no spatial consideration and geometric graphs bias toward local spatial properties, *small-world topologies* (Fig. 8 right), here considered in the flavor of the revised Watts-Strogatz model [], provide a network generation method by which the prevalence of local and long distance connections may be moderated with the parameter β via

$$p_s = \frac{k}{k + \beta(N-1-k)}$$
$$p_l = \beta \cdot p_s$$

for a ring-lattice graph, when $\beta = 0$, k is the number of neighbors a node is connected symmetrically on either side. As β increases, we see these local connections defined by p_s untether to make longer-distance (random) connections defined by p_l, until eventually, at $\beta = 1$ we have an Erdős-Rényi random graph very much akin to our aforementioned Gilbert random graph.

It is important to here realize that between the completely locally connected graph at $\beta = 0$ and the random graph at $\beta = 1$, there exists a spectrum of melanges, which may result in *scale-free* network properties where statistical outcomes at local scales may be as well observable at global scales . In some cases, this may be advantageous for the representational abilities of an LSM reservoir. Significant analysis on this subject remains to be done.

B. Learning Rules

The nature by which information may propagate through a reservoir structure (or any structure for that matter) must be defined by some set of rules. Typically, spiking neural networks run on *leaky integrate and fire* (LIF) neurons, and sometimes incorporate plasticity rules that may evoke temporary or permanent change in the synapses or even the neurons themselves.

1) Leaky Integrate and Fire Neurons: The name LIF does a fine job in self-description, because leaking, integrating, and firing is exactly what such a neuron does. At any given moment, a neuron may have some membrane potential $v(t)$ which is constantly "leaking" according to

$$\frac{dv}{dt} = \frac{-v}{\tau_{mem}} + \sum_i \delta(t - t_i)$$

where τ is the membrane time constant that defines the rate of decrease for this membrane potential and the Dirac term simply describes a presynaptic spiking event that will increase the value of v by an approximately discrete jump in proportion to the weighting coefficient of that synapse (see Fig. **??**). Therefore, spikes coming from upstream may increase the membrane potential of a neuron faster than it is leaking, and if they manage to instantiate an increase up to some threshold, the neuron in question spikes itself, sending spiking events moderated by unique synaptic weights to all of its own downstream neurons.

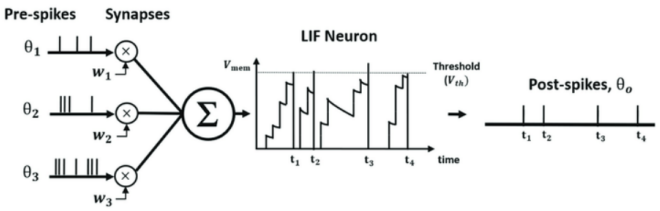

Fig. 5. Example of a figure caption.

This "rule" of information processing is already sufficient (in synchrony with a complex structure) for complex behavior with rich representational abilities. Nonetheless, generally for

LSM literature, this is only the building block for more complicated neural coordination through what are commonly known as *learning rules*.

2) Short Term Change – STSP: So far we have encountered a number of neuron and network parameters, many of which may be adapted over time to the benefit of unsupervised *learning*. By learning, we here mean that the network is changing in response to input, which may better equip itself to more meaningfully process input in the future. This learning via self-change can be a temporary or permanent affair. In the case of *Short-Term Synaptic Plasticity* (STSP) [**?**], the synaptic parameters of calcium buildup u and resource availability x define a pair of coupled differential equations

$$\frac{dx}{dt} = \frac{1-x}{\tau_D} - ux\sum_i \delta(t-t_i)$$
$$\frac{du}{dt} = \frac{U-u}{\tau_F} + U(1-u)\sum_i \delta(t-t_i)$$

that comprise a short term change in synaptic (and therefore neuron) behaviour, whereby recently active synapses are more likely to become active again as calcium increases up until a certain point where resources have sufficiently depleted to override this facilitation. See Fig. 6 for a visualization of this relationship. Also realize that of course due to their own time constants, changes in these synaptic variables will always decay back to baseline if no firing occurs. Importantly, τ_F is greater and thus calcium buildup decays more slowly than the fraction of resources required to fire, resulting in overall facilitation.

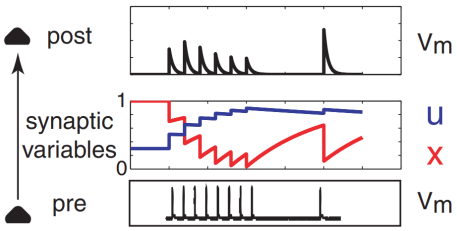

Fig. 6. The relationship of required firing resources x and calcium buildup u, borrowed from [**?**]

Notably, [**?**] found that through these temporary synaptic update mechanisms, 'activity-silent' information can be stored in this ephemeral network of facilitory connections such that a network response associated to a given input may be re-elicited with a random noise stimulus–so great is the priming effect of calcium build-up.

It is also worth considering that while the original LSM paper [1] uses similar synaptic parameters as STSP, they are implemented under the biologically derived [**?**] paradigm and are mostly depressing. STSP of the Mongillo brand has received little to no attention from the LSM literature thus far.

3) Learning Through Long Term Change – STDP: Of course for many learning tasks, it is not hard to imagine that it would be useful to elicit some permanent change in the network and in fact this has already been shown to be a useful mechanism in LSMs [**?**], most often taking the form of *Spike Timing Dependent Plasticity* (STDP), which is best summarized by the meme "Neurons that fire together wire together" -Donald Hebb. As seen in Fig. **??**, the more closely in time two neurons fire, the greater the change in the synaptic weight. If they fire in the correct order (pre-then-post), they are assumed to be correlated and the synaptic change is that of facilitation through an increase in their shared synaptic weight. Should they fire in the wrong order, the weight is decreased. These updates take the simple form of

$$\Delta w_+ = F_+(w)\cdot \exp^{\frac{\Delta t}{\tau_+}}$$
$$\Delta w_- = F_-(w)\cdot \exp^{\frac{\Delta t}{\tau_-}}$$

Where $F(\cdot)$ may be additive (the generic choice for reservoirs) or multiplicative with respect to the existing weight value.

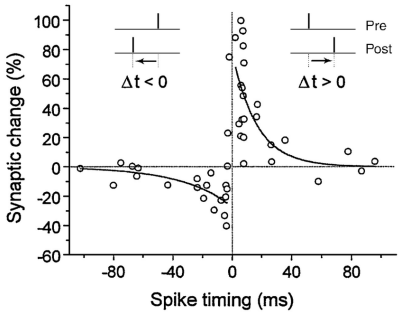

Fig. 7. Example of a figure caption.

This canonical form of Hebbian learning is biologically plausible and effective in a number of spiking neural network domains–an elegant implementation of unsupervised local learning. Note, that there are some additional choices of what neighborhood of neurons to consider for updates, such as nearest neighbor or even global (all pairing), the latter of which is employed in the present study on the basis of its higher performance for reservoirs of relevant size found in [11].

4) Hybrid: Finally, equipped with an abundance of synaptic rules, we may finally get creative and propose the novel (for LSMs) combination of STDP with STSP, thereby promoting both short and long term responsiveness to varying input. Surely, this duet merits investigation. With proper tuning, somewhere in hyper-parameter space there may exist a synergy of operation for both short and long-term unsupervised learning, and it would of course be called *Long Short-Term Plasticity* (LSTP).

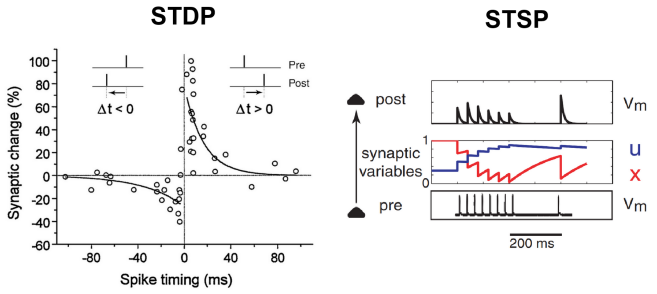

Fig. 8. Example of a figure caption.

IV. Readout Maps

As aforementioned, the present research is concerned with better understanding reservoir dynamics in order to characterize reservoir quality. That being said, a generous readout map is designed, such that we can assume a sufficient amount of information is being consumed from the liquid states to approach some pseudo-ideal classification performance. This is in contrast to the many clever spiking readout maps found in the literature, such as [10] that are native to fully spiking hardware and therefore more energy efficient and much faster. By instead opting for an external logistic regression model, however, we may reduced our bias toward reservoirs that may simply be better suited for some lean readout map in particular and can therefore better assess fundamental reservoir qualities.

In order to implement an external readout map using the classical machine learning logistic regression algorithm, spikes must first be translated into numerical data. This is achieved simply by one-hot encoding spikes in 1ms time windows for a given liquid state across all neurons. If a neuron i spikes in some window $[t, t+1ms]$ than its neuron index is assigned the value of one at the time window index ($matrix[i,t] = 1$) and and if multiple spikes occur for that neuron in that window, the value is simply incremented by that amount (therefore the data is not strictly one-hot-encoded, but rather binned).

Now equipped with some prototypical machine learning data, an even further advantage is given to the system by partitioning the data into groups of ten windows and *then* feeding them into a generic logistic regression algorithm with the appropriate labels. Once weights are learned for these "chunks," future unseen chunks may be classified appropriately.

V. Experimental Setup

All LSM simulations are coded from scratch using the *Brian Neural Simulator* [**?**]. Python packages are used for logistic regression (SciPy) and for small-world node pairing(git: Small-World). Everything else was hand-crafted.

In order to characterize meaningful representational reservoir dynamics, appropriate generic spatio-temporal tasks were curated, as described in section . Minimal training examples were shown to reservoirs of small neural populations, thus allowing reservoirs only limited resources and data, thereby standing to better gradiate performance. This approach had a secondary benefit of being more computationally cheap than large reservoirs on many hundreds of examples. Of course the latter (engineering) approach yields results of higher performance and there is indeed significant literature to this end. However, we are here not focused on recreating or improving upon the many applied experiments of LSM literature, but rather we are interested in learning about reservoirs themselves, as their own natural phenomena of dynamical systems. It would only be an added benefit should results in this way provide insights on performance-oriented approaches, or on natural reservoirs like (parts of) the brain. We are here simply trying to study liquid state reservoirs in and of themselves. Therefore, reservoirs of only 135 neurons receive a mere two training samples before being tested on single unseen test-sample to measure performance for both the poisson pattern and spoken digit classfication tasks. The prior trials last only 100ms and the latter 700ms.

With such a great number of possible hyperparameters and a clean slate with respect to ideas about representational dynamics, a number of hyperparameter sweeps were run both for the Poisson pattern classification and the Heidelberg spoken digit dataset. Due to typical computational constraints, may sub-sweeps were run separately, even when ideally they would combined into one combinatoric behemoth.

A. General Sweep

TABLE I
General Poisson Sweep

Hyperparameter	Values
Input Sparsity	0.1, 0.2, 0.3
Reservoir Sparsity	0.1, 0.2, 0.3
Learning	LIF, STDP, STSP, LSTP
Topology	Rand, Geo, SMW
Refractory	0ms, 3ms
Delay	0ms, 1.5ms
Random	0.1, 0.2, 0.3
Beta	0.0, 0.33, 0.66
Dimensions	$15 \times 15 \times 3$
Excite/Inhibit	True, False
Input feed	reset, continuous

B. Repeatability Sweep

Top performing reservoir configurations are rerun with new random weight initialization to asses the variance associated as consequence of these initial conditions. Of course the research would benefit from many repeated sweeps, but due to computational resource limitations, we limit ourselves to 10 reruns of the top ten performing configurations for each sweep.

**Repeated Heidelberg sweeps use different training and testing samples.

C. Weight Initialization and Excitatory/Inhibitory Sweep

A reduced version of the general sweeps is run for each tasks, with and without inhibitory/excitatory behaviour, again providing insight on the effects of initial conditions.

D. Control Sweeps

All task are repeated without the intermediate reservoir (but with the same readout map) to asses the information gained from reservoir dynamics.

E. Principle Compenent Analysis, Distance, and Clustering Measures

We preface here that in order to navigate the unseeably high dimension representational state space, a number of tactics are cautiously applied. Principle Component Analysis is a data compression technique that reduces dimensionality on the basis of preserving directions of maximum variance. This allows for the visualization of higher-dimensional spaces at a palpable and familar 3D scale, but it must be said that these are not necessarily lower-resolution versions of the full-dimensionality. Fundamental nuances of the data may be lost, we therefore use this technique here with tentative caution, drawing conclusions only about the relative dynamics–not inherent characteristics–of the full data. We are not alone in this approach however, and PCA for reservoir analysis can be found in [11], [**?**].

Euclidean distance measures between state vectors are as well used for assessment of dynamics, as in [**?**], and finally K-means Clustering is used as a metric for coherent separation of pattern representations.

VI. Results and Analysis

Due to the vast search space of possible LSMs and limited computation, the following results stand as a heuristically-advised sampling from this space and are by no means complete. However, the number of experiments presented should be sufficient to better our intuitions on premium reservoirs and their associated dynamics.

A. Raw Performance

In terms of classification performance, many reservoirs perform well and even achieve perfect classification with as little as two training examples per class. In the sparsity sweep, – out of – manage this feat. At typical high performance plot can be seen Fig. 9, where we can see after an initial wash-out period, during which time the reservoir is getting “acquainted” with its testing sample (moving from its random voltage and/or weight initializations), the reservoir quickly becomes certain which sample is which (in this case the spoken digits ”zero”, ”one”, and ”two”).

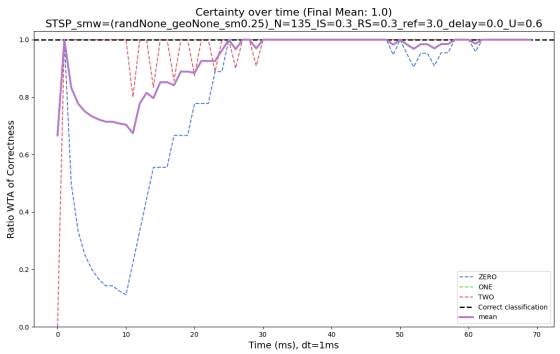

Fig. 9. A typical certainty plot for a high performing LSM. Here the dotted lines are certainty for each class and the filled line is their mean certainty. Note that certainty is defined by the number of correct guess so far in proportion to whatever is the most common prediction. A value of 1 indicates correct classification at that moment in time.

Through this ”sweep-name”, we can glean certain favorable combinations of learning rules with topologies, and in this case we see STSP with a random topology notably out-perform its competitors for different input and reservoir sparsities.

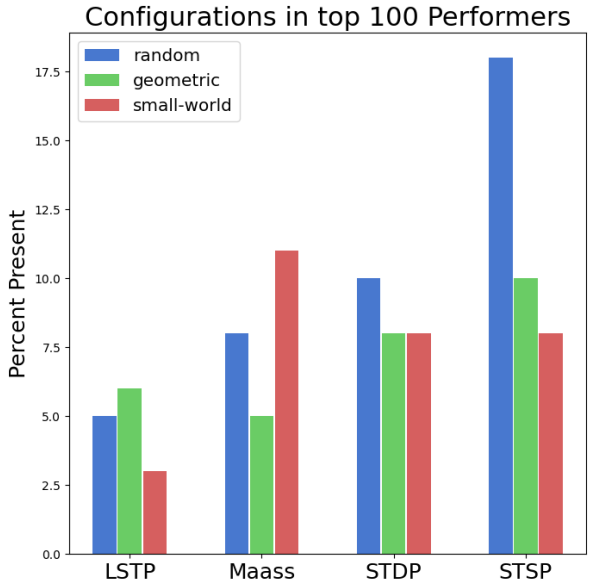

Fig. 10. Here we see a clear performance advantage for STSP in combination with a random topology.

B. Principle Component Analysis and Representational Spaces

Finally, we come to state space and find observe several consistent behaviour among top-performing reservoirs. Looking at the first three PCs for all samples, and plotting the path over time for each class centroid, we generally find chaotic scribbles. *However,* for high performing reservoirs, we find many perfect repetitious cycles as well as slowly evolving cycles (see Fig. 11). Visually, these are analogous to perfect limit cycles and chaotic attractors respectively.

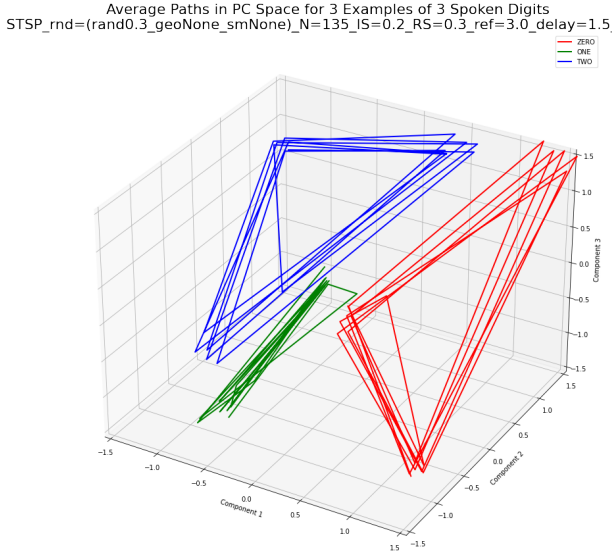

Fig. 11. Example of a figure caption.

VII. Discussion and Future Works

We thus conclude our survey of liquid state machines with lessons learned in one hand, and future questions in the other. Along the way we have found that many LSMs are capable of correctly classifying spiking speech patterns with incredibly small training sets and limited nerual resources. Further more, similar findings are produces for the abstract spatio-temporal poisson patterns and therefore, we expect these results to hold for many such tasks. Specifically, we have found the under-studied STSP to be the prominent performer, especially in tandem with random topologies. We speculate that the fast responsiveness of short term plasticity and the generic structure of random reservoirs pair well for generic tasks. Finally, we notice that behavior in state space is ostensibly correlated with performance and therefore may be used as an unsupervised reservoir assessment tool. In particular, slowly evolving limit cycles (“chaotic attractors”) appear to be the often favorable reservoir behavior.

With these results in mind, the work to do has only grown–namely on two fronts. The present research should be conducted in greater numbers with more computational power, more through validation, and on more task-types. Once these findings have been fortified with more experience, many ideas for the future come to mind. A learning algorithm trained on the hyper-parameters of many thousands of reservoirs across many hundreds of tasks may be used to optimize generalizability of LSMs. Moreover, using state space analysis, we can bias for useful dynamics, or even specifically enforce coherent separation with a learning algorithm on weight initialization. Through either of these methods, we may come to find a rich unsupervised state space that is semantically isomorphic to the world we live in.

References

[1] Maass, Wolfgang, Thomas Natschläger, and Henry Markram. ”Real-time computing without stable states: A new framework for neural computation based on perturbations.” Neural computation 14.11 (2002): 2531-2560.

[2] Jaeger, Herbert. ”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 (2001): 13.

[3] Schrauwen, Benjamin, et al. ”Compact hardware liquid state machines on FPGA for real-time speech recognition.” Neural networks 21.2-3 (2008): 511-523.

[4] Roy, Subhrajit, Amitava Banerjee, and Arindam Basu. ”Liquid state machine with dendritically enhanced readout for low-power, neuromorphic VLSI implementations.” IEEE transactions on biomedical circuits and systems 8.5 (2014): 681-695.

[5] Wang, Qian, Yingyezhe Jin, and Peng Li. ”General-purpose LSM learning processor architecture and theoretically guided design space exploration.” 2015 IEEE Biomedical Circuits and Systems Conference (BioCAS). IEEE, 2015.

[6] Misra, Janardan, and Indranil Saha. ”Artificial neural networks in hardware: A survey of two decades of progress.” Neurocomputing 74.1-3 (2010): 239-255.

[7] Seo, Jae-sun, et al. ”A 45nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons.” 2011 IEEE Custom Integrated Circuits Conference (CICC). IEEE, 2011.

[8] Merolla, Paul, et al. ”A digital neurosynaptic core using embedded crossbar memory with 45pJ per spike in 45nm.” 2011 IEEE custom integrated circuits conference (CICC). IEEE, 2011

[9] Verstraeten, David, et al. ”Isolated word recognition with the liquid state machine: a case study.” Information Processing Letters 95.6 (2005): 521-528.

[10] Zhang, Yong, et al. ”A digital liquid state machine with biologically inspired learning and its application to speech recognition.” IEEE transactions on neural networks and learning systems 26.11 (2015): 2635-2649.

[11] Jin, Yingyezhe, Yu Liu, and Peng Li. ”SSO-LSM: A sparse and self-organizing architecture for liquid state machine based neural processors.” 2016 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH). IEEE, 2016.

[12] Wijesinghe, Parami, et al. ”Analysis of liquid ensembles for enhancing the performance and accuracy of liquid state machines.” Frontiers in neuroscience 13 (2019): 504.

[13] Kaiser, Jacques, et al. ”Scaling up liquid state machines to predict over address events from dynamic vision sensors.” Bioinspiration biomimetics 12.5 (2017): 055001.

[14] Soures, Nicholas, and Dhireesha Kudithipudi. ”Deep liquid state machines with neural plasticity for video activity recognition.” Frontiers in neuroscience 13 (2019): 686.

[15] Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, Friedemann Zenke, December 6, 2019, ”Heidelberg Spiking Datasets”, IEEE Dataport, doi: https://dx.doi.org/10.21227/51gn-m114.

[16] LIF image

[17] STDP diagram

[18] LSM diagram