Expert Systems with Applications 39 (2012) 1597–1606

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Topological constraints and robustness in liquid state machines

Hananel Hazan*, Larry M. Manevitz

Department of Computer Science, University of Haifa, Mount Carmel, Haifa 31905, Israel

ARTICLE INFO

Keywords:
Liquid State Machine
Reservoir computing
Small world topology
Robustness
Machine learning

ABSTRACT

The Liquid State Machine (LSM) is a method of computing with temporal neurons, which can be used amongst other things for classifying intrinsically temporal data directly unlike standard artificial neural networks. It has also been put forward as a natural model of certain kinds of brain functions. There are two results in this paper: (1) We show that the Liquid State Machines as normally defined cannot serve as a natural model for brain function. This is because they are very vulnerable to failures in parts of the model. This result is in contrast to work by Maass et al. which showed that these models are robust to noise in the input data. (2) We show that specifying certain kinds of topological constraints (such as "small world assumption"), which have been claimed are reasonably plausible biologically, can restore robustness in this sense to LSMs.

1. Introduction

Processing in artificial neurons typically is a-temporal. This is because the underlying basic neuronal model, that of Pitts and McCulloch (1943) is a-temporal by nature. As a result, most applications of artificial neural networks are related in one way or another to static pattern recognition. On the other hand, it has long been recognized in the brain science community that the McCullough–Pitts paradigm is inadequate. Various models of differing complexity have been promulgated to explain the temporal capabilities (amongst other things) of natural neurons and neuronal networks.

However, during the last decade, computational scientists have begun to pay attention to this issue from the neurocomputation perspective as well, e.g. Fern and Sojakka (n.d.), Jaeger (2001a, 2001b, 2002), Lukosevicius and Jaeger (2009) and Maass, Natschläger, and Markram (2002a, 2002b, 2002d), and investigations as to the computational capabilities of various models are being investigated.

One such model, the Liquid State Machine (LSM) (see Fig. 1) (Maass et al., 2002a), has had substantial success recently. The Liquid State Machine is a somewhat different paradigm of computation. It assumes that information is stored, not in "attractors" as is usually assumed in recurrent neural networks, but in the activity pattern of all the neurons which feed-back in a sufficiently recurrent and inter-connected network. This information can then be recognized by any sufficiently strong classifier such as an Adaline (Widrow & Hoff, 1960), Back-Propagation, SVM[1] or Tempotron (Gutig & Sompolinsky, 2006). (The name "liquid state" comes from the idea that the history of, e.g. timings of rocks thrown into a pond of water, is completely contained in the wave structure.) Moreover, the "persistence of the trace" (or as Maass put it, the "fading memory" (Lukosevicius & Jaeger, 2009)) allows one to recognize at a temporal distance the signal that was sent to the liquid; and sequence and timing effects of inputs.

The Liquid State Machine is a recurrent neural network. In its usual format (Lukosevicius & Jaeger, 2009; Maass et al., 2002a), each neuron is a biologically inspired artificial neuron such as an "integrate and fire" (LIF) neuron or an "Izhikevich" style neuron (Izhikevich, 2003). The connections between neurons define the dynamical process, and the recurrence connections define what we call the "topology" in this paper. The properties of the artificial neurons, together with these recurrences, results in any sequence of history input being transformed into a spatio-temporal pattern activation of the liquid. The nomenclature comes from the fact that one can intuitively look at the network as if it was a "liquid" such as a pond of water, the stimuli are rocks thrown into the water, and the ripples on the pond are the spatio-temporal pattern.

In the context of LSM the "detectors" are classifier systems that receive as input a state (or in large systems a sample of the elements of the liquid) and are trained to recognize patterns that evolve from a given class of inputs. Thus a detector could be a SVM or an Adaline (Widrow & Hoff, 1960), perceptron (Pitts & McCulloch, 1943), or three level back propagation neural networks, etc.

* Corresponding author. Tel.: +972 4 8288337; fax: +972 4 8288181.
E-mail addresses: hhazan01@cs.haifa.ac.il (H. Hazan), manevitz@cs.haifa.ac.il (L.M. Manevitz).

1 SVM = support vector machine.

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.06.052

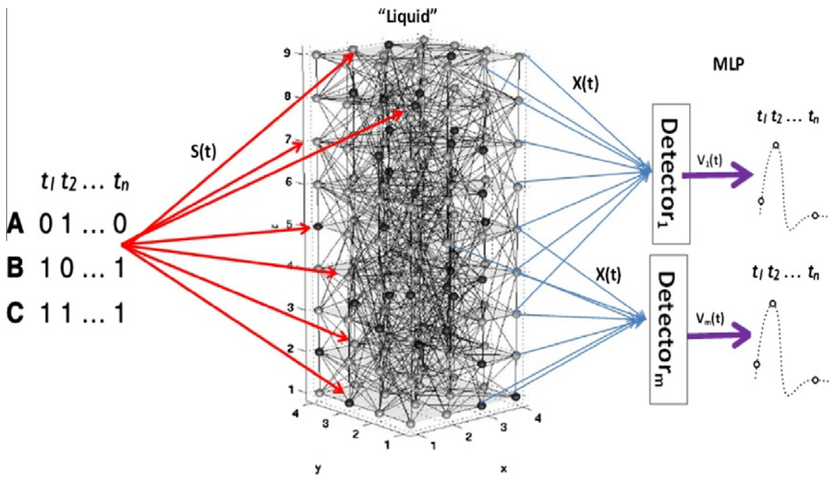

Fig. 1. Liquid State Machine framework.

The term detector is standard in the LSM community and date back to Maass et al. (Jaeger, 2001a; Lukosevicius & Jaeger, 2009; Maass, 2002; Maass & Markram, 2004; Maass et al., 2002b) the idea is that the "detectors" are testing whether the information for classification resides in the liquid; and thus are not required to be biological. In this way, it is theoretically possible for the detectors to recognize any spatio-temporal signal that has been fed into the liquid; and thus the system could be used for, e.g. speech recognition, or vision, etc.

This is an exciting idea and, e.g. Maass and his colleagues have published a series of papers on it. Amongst other things, they have recently shown that once a detector has been sufficiently trained at any time frame, it is resilient to noise in the input data and thus it can be used successfully for generalization (Bassett & Bullmore, 2006; Fern & Sojakka, n.d.; Maass et al., 2002b).

Furthermore, there is a claim that this abstraction is faithful to the potential capabilities of the natural neurons and thus is explanatory to some extent from the viewpoint of computational brain science. Note that one of the underlying assumptions is that the detector works without memory; that is the detector should be able to classify based on instantaneous static information; i.e. by sampling the liquid at a specific time. That this is theoretically possible is the result of looking at the dynamical system of the liquid and noting that it is sufficient to cause the divergence of the two classes in the space of activation.

Note that the detector systems (e.g. a back propagation neural network, a perceptron or a support vector machine (SVM)) are not required to have any biological plausibility; either in their design or in their training mechanism, since the model does not try to account for the way the information is used in nature. Despite this, since natural neurons exist in a biological and hence noisy environment, for these models to be successful in this domain, they must be robust to various kinds of noise. As mentioned above, Maass et al. (Lukosevicius & Jaeger, 2009; Maass, Legenstein, & Markram, 2002; Maass et al., 2002b; Maass & Markram, 2004) addressed one dimension of this problem by showing that the systems are in fact robust to noise in the input. Thus small random shifts in a temporal input pattern will not affect the LSM's ability to recognize the pattern. From a machine learning perspective, this means that the model is capable of generalization.

However, there is another component to robustness; that of the components of the system itself.

In this paper we report on experiments performed with various kinds of "damage" to the LSM and unfortunately have shown that the LSM with any of the above detectors is not resistant, in the sense that small damages to the LSM neurons reduce the trained classifiers dramatically, even to essentially random values (Hazan & Manevitz, 2010; Manevitz & Hazan, 2010).

Seeking to correct this problem, we experimented with different architectures of the liquid. The essential need of the LSM is that there should be sufficient recurrent connections so that on the one hand, the network maintains the information in a signal, while on the other hand it separates different signals. The models typically used are random connections; or those random with a bias towards "nearby" connections. Our experiments with these topologies show that the network is very sensitive to damage because the recurrent nature of the system causes substantial feedback.

Taking this as a clue, we tried networks with "hub" or "small world" (Albert & Barabási, 2000; Barabási, 2000; Barabási & Albert, 1999) architecture. This architecture has been claimed (Achard, Salvador, Whitcher, Suckling, & Bullmore, 2006; Bassett & Bullmore, 2006; Varshney, Chen, Paniagua, Hall, & Chklovskii, 2011) to be "biologically feasible".

The intuition was that the hub topology, on the one hand, integrates information from many locations and so is resilient to damage in some of them; and on the other hand, since such hubs follow a power rule distribution, they are rare enough that damage usually does not affect them directly. This intuition was in fact borne out by our experiments.

2. Materials and methods

We simulated the Liquid State Machine with 243 integrate and fire neurons (LIF) in the liquid following the exact set up of Maass and using the code available at the Maass laboratory software "A neural Circuit SIMulator".[2] To test variants of topology we re-implemented the code, available at our website.[3] The variants of the topologies implemented are described in the paper below as are the types of damages. Input to the liquid was at 30% of the neurons, the same input at all locations in a given time instances. The detectors of the basic networks were back propagation networks with three levels with 3 neurons in the hidden level and one output neuron. In most experiments, the input was given by the output of all non-input neurons of the liquid (i.e. 170 inputs to the detector). In some experiments (see section below) the inputs to the detector were given over 20 time instances and so the detector had 3400

[2] http://www.lsm.tugraz.at/csim/.
[3] http://www.cri.haifa.ac.il/neurocomputation.

inputs. The networks were tested with 20 random temporal binary sequences of length 45 chosen with uniform distribution. The experiments were repeated 500 times and statistics reported.

3. Theory/calculations

As discussed in the introduction, in a system, there are two sources of potential instability. First is the issue of small variants in the input. Systems need to balance the need of separation with generalization. That is, on the one hand, one may need to separate inputs with small variations into separate treatment, but on the other hand, small variants may need to be treated as "noise" or generalization of the trained system. For the LSM, as is typically presented in the literature, it is understood, e.g. from the work of Lukosevicius and Jaeger (2009) and Maass (2002)) that the LSM and its variants do this successfully in the case of spatio-temporal signals.

The second issue concerns that of the sensitivity of the system to small changes in the system itself, which we choose to call "damages" in this paper. This is very important if, as is the case for LSM, it is supposed to be explanatory for biological systems.

Our experiments therefore are based on simulating the LSM with temporal sequences and calculating how resistant they are to two main kinds of such damages. The damages chosen for investigation were: (1) at each time instance a certain percentage of neurons in the liquid would refuse to fire regardless of the internal charge in its state; (2) at each time instance a certain percentage of neurons would fire regardless of the internal charge, subject only to the limitation of the refractory period.

Since the basic results (see below) showed that the standard variants of LSM were not robust to these damages at various small levels, we considered topological differences in the connectivity of the LSM.

3.1. First experiments: LSMs are not robust

3.1.1. The experiments

To test the resistance of standard LSM to noise, we (i) downloaded the code of Maass et al. from his laboratory site[4] and then implemented two kinds of damage to the liquid and (ii) re-implemented the LSM code so that we could handle variants. These models use a kind of basic neuron that is of the "leaky integrate and fire" (LIF)[5] variety and in Maass' work, the neurons are connected randomly but with some biologically inspired parameters: 20% inhibitory and a connectivity constraint giving a preference to geometrically nearby neurons over more remote ones. (For precise details on these parameters, see: neural Circuit SIMulator4 and Maass and Markram (2002).) External stimuli to the network were always sent to 30% of the neurons, always chosen to be excitatory neurons. Initially, we experimented with two parameters: (i) the percentage of neurons damaged; (ii) the kinds of damages. The kinds were either transforming a neuron into a "dead" neuron; i.e. one that never fires or transforming a neuron into a "generator" neuron, i.e. one which fires as often as its refractory period allows it, regardless of its input. We did experiments with different kinds of detectors: Adaline (Widrow & Hoff, 1960), Back-Propagation, SVM and Tempotron (Gutig & Sompolinsky, 2006).

Classification of new data could then be done at any of the signal points. We ran experiments as follows: we randomly chose twenty temporal inputs; i.e. random sequences of 0s and 1s of length 45, corresponding to spike inputs over a period of time; and trained an LSM composed of 243 integrate and fire neurons as in the liquid (Maass & Markram, 2002) to recognize ten of these inputs and reject the other ten. Each choice of architecture was run 500 times varying the precise connections randomly. We tested the robustness of the recognition ability of the network with the following parameters:

– The neurons in the network were either leaky integrate and fire neurons (Maass, 2002) or Izhikevich (Izhikevich, 2003) style neurons.
– The average connectivity of the networks was maintained at about 20% chosen randomly in all cases although with different distributions.
– The damages were either "generators", i.e. the neurons issued a spike whenever their refractory period allowed it; or they were "dead" neurons that could not spike.
– The degree of damage was systematically checked at 0.1%, 0.5%, 1%, 5%, and 10% in randomly chosen neurons.

The results shown in tables throughout the paper are in percentages, over the (500) repeated tests. One hundred percent indicates that all the 20 vectors of one test, over 500 repetitions of the test were fully recognized correctly. Fifty percent indicates that only half the vectors over 500 times were recognized. (This corresponds to a chance baseline). The graphs presented below show the full distribution of all the tests and the results over all the kinds of damages and all varieties of topologies. As expected, they distribute as Gaussian, but note that the average success rate varies from a baseline of 10 successes (50%) for random guessing (see Fig. 2) to as high as almost 20 (98%) for generalization in certain cases and 88% for some of the damages.

3.2. Second experiments: modifications of the LSM

3.2.1. Different kinds of basic neurons

In attempts to restore the robustness to damage, we experimented with the possibility that a different kind of basic neuron might result in a more resilient network. Accordingly, we implemented the LSM with various variants of "leaky integrate and fire neurons", e.g. with history dependent refractory period (Manevitz & Marom, 2002) and by using the model of neurons due to Izhikevich (2003). The results under these variants were qualitatively the same as the standard integrate and fire neuron. (The Izhikevich model produces a much more dense activity in the network and thus the detector was harder to train but in the end the network was trainable and the results under damage were very similar.) Accordingly, we report only results with the standard integrate and fire neuron as appears, e.g. in Maass' work (Maass, 2002).

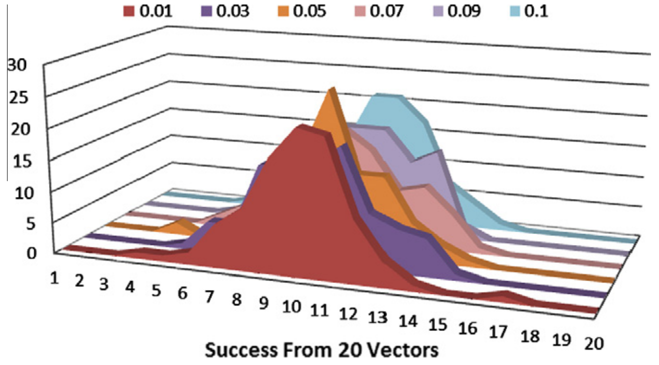

Fig. 2. Results of identification of random vectors on an untrained LSM with uniform random connections. This is a baseline. The result is a Gaussian distribution around 10 vectors.

[4] A neural Circuit *SIM*ulator: http://www.lsm.tugraz.at/csim/.
[5] LIF = leaky integrate and fire.

3.2.2. Allowing detectors to have memory

In trying to consider how to make the model more robust to damage, we investigated the fact that the detector has no memory. Perhaps, if we allow the detector to follow the development of the network for a substantial amount of time, both in training and running, it would be more robust. To check this, we took the most extreme other case; we assumed that the detector system in fact takes as input a full time course of 20 iterations of the output neurons of the liquid. This means that instead of a NN with input of 170; we had one with 20 times 170 time course inputs. It seemed reasonable that (i) with so much information, it should be relatively easy to train the detector; (ii) one could hope that damage in the liquid would be local enough that over the time period, the detector could correct for it. In order to test this, we re-implemented the LSM detector to allow for this time entry.

Our detector was trained and tested as follows. There were 170 output units. At a ''signal point'' each of them was sampled for the next 20 iterations and all of these values were used as a single data point to the detector. Thus the detector had 170 times 20 inputs. We chose separate detector points typically at intervals of 50. We then used back propagation on these data points. This means that eventually the detector could recognize the signal at any of the ''signal points''; after training there was no particular importance to the choice of separation of the signal points except that there was no overlap between the data points. While we did not control for any connections between the intervals of data points (i.e. 50, and we also checked other time intervals) and possible natural oscillations in the network, we do not believe there were any. As anticipated, there was no significant trouble in training the network to even 100% of recognition of the training data.

The ''detectors'' were three level neural networks, trained by back-propagation. We also did some experiments with the Tempotron (Gutig & Sompolinsky, 2006); and with a simple Adaline detector (Widrow & Hoff, 1960). Training for classification could be performed in the damage-less environment successfully with any of these detectors. Then we exhaustively ran tests on these possibilities.

In all of these tests, following Maass (2002), Maass and Markram (2002) and Maass et al. (2002a), we assumed that approximately 20% of the neurons of the liquid were of the inhibitory type. The architecture of the neural network detector was 204 input neurons (which were never taken from the neurons in the LSM which were also used as inputs to the LSM) 100 hidden level neurons and one neuron for the output. Results running the Maass et al. architecture are presented in Fig. 4 and Table 4 and can be compared with a random connected network of 10% average connectivity, see Table 2.

The bottom line (see the results section) was that even with low amounts damage and under most kinds of connectivity, the networks would fail; i.e. the trained but damaged network loss of function was very substantial and in many cases could not perform substantially differently from a random selection.

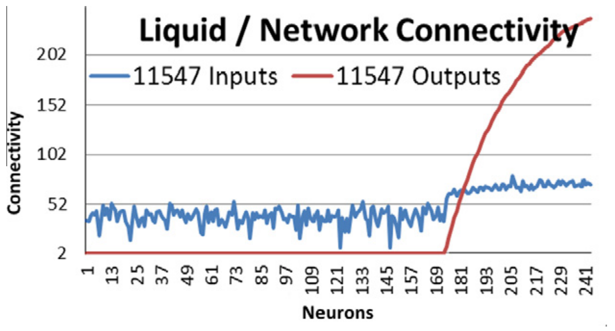

Fig. 3. Histogram of connection distributions when the output connections were randomly selected chosen according to a power-law. Note that the input histogram is different than the output histogram.

3.3. Third experiments: changing the architecture

Our next approach, and ultimately the successful one, was to experiment with different architectures. The underlying intuition is that the recurrent nature of the liquid results in feedback of information making the network dynamics too sensitive to changes in the network. Since one can look at ''damages'' as

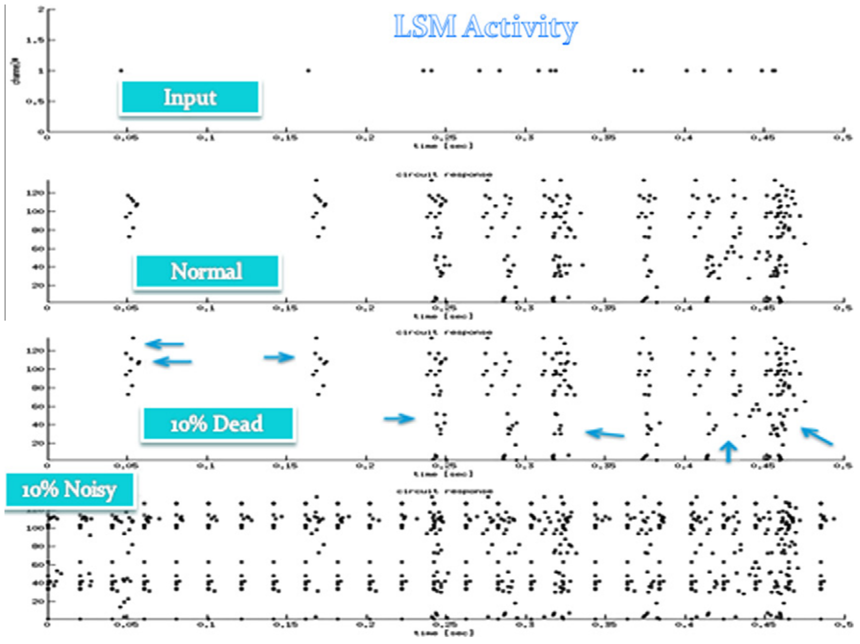

Fig. 4. Maass LSM (a) normal operation; (b) with 10% dead damage; (c) with 10% noise. One can easily discern the large change in the reaction of the network.

instantaneous changes in the architecture, it seems reasonable to design architectures that can somehow "filter" out minor changes.

The liquids were varied in their topologies in the following ways:

1. Random connectivity. Each neuron in the network is connected to 20% of the other neurons in a random fashion. (i) In the original Maass topology the connections are chosen with a larger bias for nearby neurons (see Maass, 2002; Maass et al., 2002a; Maass, Natschläger, & Markram, 2002c). This is the literature standard and is what is usually meant as LSM. (ii) We also tested a network without such bias; i.e. the connections are chosen to 20% of the other neurons randomly and uniformly. The results presented below showed that these architectures are not robust.
2. Reducing the connectivity to 10% and 5% in the above arrangement. The intuition for this was that with lower connectivity, the feed-back should be reduced. The results presented below show that this intuition is faulty and that these networks are even less robust than the above (see Tables 1, 2, 5 and 6).
3. Implementation of "Hub" topologies in either input connectivity or output. The intuition here is that the relative rarity of "hubs" results in their damage being a very rare event. But when they are not damaged, they receive information from many sources and can thus filter out the damage thus alleviating the feedback in the input case. In the output hub case, the existence of many hubs should allow the individual neurons to filter out noise.

   The construction of hubs was done in various fashions:

   a. Hand design of a network with one hub for input. See Appendix A for a full description of this design.
   b. Small world topologies. Since small world topologies follow power law connectivity, they produce hubs. On the other hand such topologies are thought to emerge in a "natural" fashion (Albert & Barabási, 2000; Barabási, 2000; Barabási & Albert, 1999; Varshney et al., 2011) and appear in real neuronal systems (Albert & Barabási, 2000; Bassett & Bullmore, 2006), see Fig. 3. Note however, that in our context there are two directions to measure the power law: input and output connectivity histograms for the neurons. We checked the following variants:

Table 1
Five percentage uniform random connectivity without memory input to the detector.[a]

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	100%	55%	53%	52%	51%	49%
Noisy neurons	100%	63%	54%	55%	51%	50%
Dead and noisy	100%	55%	52%	52%	50%	50%
Generalization	100%	93%	88%	80%	75%	78%

[a] For all the tables that are shown in this paper, 50% is the baseline of random classification.

Table 2
Ten percentage uniform random connectivity without memory input to the detector.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	100%	56%	53%	51%	51%	49%
Noisy neurons	100%	73%	58%	54%	51%	52%
Dead and noisy	100%	59%	54%	52%	52%	51%
Generalization	100%	100%	93%	88%	83%	81%

i. Input connectivity is power law. That is we assign a link from a uniformly randomly chosen neuron to a second neuron chosen randomly according to a power law. In this case the input connectivity follows a power law; while the output connectivity follows a Gaussian distribution.
ii. Output connectivity is power law. That is we reverse the above. In this case the input connectivity is Gaussian while the output connectivity is power law.
iii. Replacing "Gaussian" with "uniform" in case (i) above.
iv. Replacing "Gaussian" with "uniform" in case (ii) above.
v. We also tried choosing a symmetric network with power law connectivity (i.e. for both input and output.) Note that in this case, the same neurons served as "hubs" both for input and output.
vi. Finally, we designed an algorithm to allow distinct input and output power law connectivity. In this case the hubs in the two directions are distinct. Algorithms 1 and 2 below accomplish this task.

Algorithm 1

```
  Generate a random number between min and max value
   with Power law distribution, Input: min,max, size,
   How_many_numbers, counter Arry = array, Magnify = 5
  for $i$ = 1 to How_many_numbers
  index = random(array.start,array.end)
  end_array = array.end
  candidate = array[index]
  AddCells(array, Magnify);
  for $t$ = 0 to Magnify
    array[end_array + $t$] = candidate
  end for
  shuffle(array)
  output_Array[$i$] = candidate
  counterArry[candidate]++
end for
shuffle(counterArry)
Output output_Array,counterArry
```

Algorithm 2

```
  Create the connectivity matrix for the liquid network using
   the Algorithm 1 as an Input weight_Matrix
use algorithm 1 to creart (arraylist, counterArry)
counter = 0
for i=1 to counterArry.lenght
  for t=1 to counterArry[i]
    weight_Matrix[i, arraylist[counter]]=true
    counter++
  end for
end for
```

One problem with the various algorithms for designing power law connectivity is that under a "fair" sampling, the network might not be connected. This means that such a network actually has a lower, effective connectivity. We decided to eliminate this problem by randomly connecting the disconnected components (either from an input or output perspective) to another neuron chosen randomly but proportionally to the connectivity. (This does not guarantee connectivity of the graph, but makes it unlikely, so that the effective connectivity is not substantially affected.)

4. Results

4.1. First experiments: LSM is not robust

First, there was not much difference between the detectors; so eventually we restricted ourselves to the back-propagation detector. (Note that none of units of the liquid input were accessed by the detectors were allowed to be input neurons of the liquid.) It turned out that while the detector is able to learn the randomly chosen test classes successfully, if there is sufficient average connectivity (e.g. 20%), almost any kind of damage caused the detector to have a very substantial decay in its detecting ability (see Table 3). Note that even with lower connectivity, which has less feedback, the same phenomenon occurs. See Table 1 (5% connectivity) and Table 2 (10% connectivity).

When the network is connected randomly but with bias for geometric closeness as in Maass' distribution, the network is still very sensitive (although a bit less so). Compare Table 4 to Table 3.

After our later experiments, we returned to this point (see concluding remarks, below). In Fig. 4 we illustrate the difference in reaction of the network by a raster (ISI) display. Note that with 10% damage, it is quite evident to the eye that the network diverges dramatically from the noise free situation. In Tables 1–4 one can see this as well with 5% noise for purely random connectivity. Actually, with low degrees of damage the detectors under even the Maass connectivity (see Table 4) show dramatic decay in recognition although not to the extremes of random connectivity. These results (see Tables 1–4) were robust and repeatable under many trials and variants.

Accordingly, we conclude that the LSM, either as purely defined with random connectivity, or, as implemented in Maass et al. (2002a) cannot serve as a biologically relevant model.

4.2. Second experiments: varying the neurons and allowing the detectors to have memory

4.2.1. Variants of neurons (history dependent refractory period and izhikevich)

The results under these variants were qualitatively the same as the standard integrate and fire neuron. (The Izhikevich model produces a much more dense activity in the network and thus the detector was harder to train but in the end the network was trainable and the results under damage were very similar.) Accordingly, we report only results with the standard integrate and fire neuron as appears, e.g. in Maass' work.

4.2.2. Detectors with memory input

The "detectors" in our experiments were either three level neural networks, trained by back-propagation, the Tempotron (Gutig & Sompolinsky, 2006); or with a simple Adaline detector (Widrow & Hoff, 1960). Training for classification could be performed in the damage-less environment successfully with any of these detectors. We exhaustively ran tests on these possibilities; including damage degree and kinds and detector types.

Tables 5–8 show the results with different uniform connectivity in the liquid when there is memory input to the detector. Table 8

Table 3
Twenty percentage uniform random connectivity without memory input to the detector.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	99%	60%	53%	51%	51%	50%
Noisy neurons	99%	86%	65%	58%	52%	50%
Dead and noisy	99%	65%	55%	53%	50%	51%
Generalization	99%	100%	97%	94%	87%	84%

Table 4
Twenty percentage connectivity under Maass's distribution preferring local connections.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	90%	60%	52%	51%	50%	50%
Noisy neurons	90%	78%	57%	52%	52%	52%
Dead and noisy	90%	54%	52%	53%	50%	50%
Generalization	90%	96%	93%	93%	84%	84%

Table 5
Five percentage uniform random connectivity with memory input to the detector.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	100%	55%	53%	53%	51%	50%
Noisy neurons	100%	63%	54%	54%	53%	51%
Dead and noisy	100%	56%	53%	52%	51%	51%
Generalization	100%	93%	87%	80%	75%	79%

Table 6
Ten percentage uniform random connectivity with memory input to the detector.[a]

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	100%	58%	55%	53%	49%	50%
Noisy neurons	100%	74%	59%	57%	54%	50%
Dead and noisy	100%	61%	54%	55%	50%	50%
Generalization	100%	96%	92%	85%	82%	82%

[a] For all the tables that shown in this paper, 50% is the baseline of random classification.

Table 7
Twenty percentage uniform random connectivity with memory input to the detector.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	100%	63%	55%	52%	50%	50%
Noisy neurons	100%	87%	67%	61%	54%	52%
Dead and noisy	100%	68%	57%	52%	50%	49%
Generalization	100%	98%	97%	95%	89%	86%

Table 8
Maass's distribution like in Table 4 but with memory input to the detectors.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	100%	61%	53%	49%	49%	50%
Noisy neurons	100%	79%	60%	55%	51%	49%
Dead and noisy	100%	64%	55%	52%	51%	52%
Generalization	100%	100%	96%	93%	84%	85%

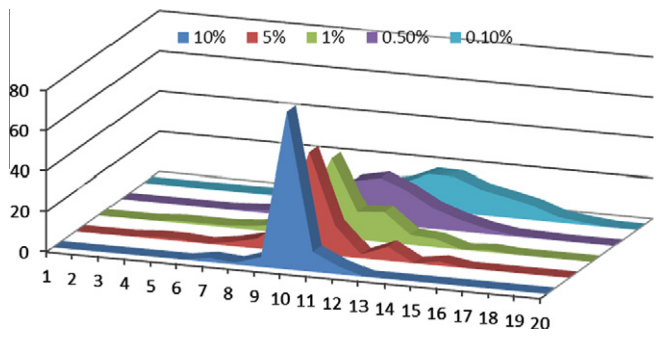

Fig. 5. Histographs of correctness results in LSM networks with 20 time interval input, different amounts of "dead" neuron damage, average connectivity of 20% with a uniform random distribution on the connections.

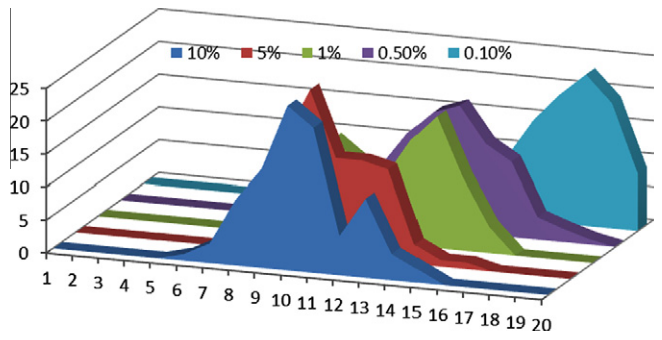

Fig. 6. Histographs of correctness results in LSM networks with 20 time interval input, different amounts of "noise generator" neuron damage, average connectivity of 20% with a uniform random distribution on the connections.

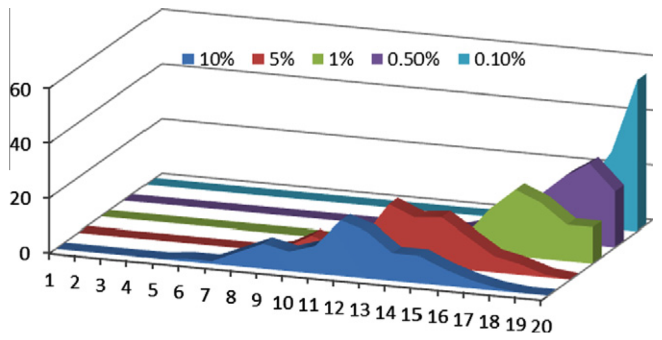

Fig. 7. Histographs of correctness results in LSM networks with one hub distribution with different amounts of "noise generator" neuron damage.

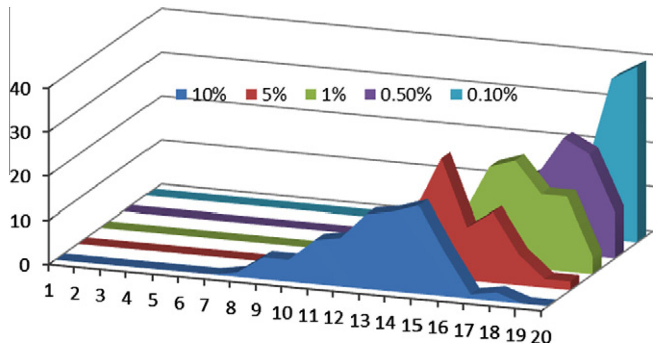

Fig. 8. Histographs of correctness results in LSM networks with different amounts of "dead" neuron damage with one hub distribution.

shows similar result (like in Table 4) for the Maass connectivity with memory input to the detector. Histographs of sample results with 5% and 10% damage for the neural network detectors are presented in Figs. 5–13 . (Since the results for the other detectors were similar, we did not run as many tests on them) Note, Figs. 5–13 refer to the various kinds of hub architectures with the memory in the detector.

In all of these tests, following Maass, we assumed that approximately 20% of the neurons of the liquid were of the inhibitory type. The architecture of the neural network detector was 204 input neurons (which were never taken from the neurons in the LSM which were also used as inputs to the LSM $\times$ 30 times) 3 hidden level neurons and one neuron for the output. For 20% connections the Maass et al. architecture without memory in the detector as presented in Table 4 can be compared with a uniform random connected network of 20% average connectivity without memory in the detector in Table 3, and can be compared as well with the Maass topology with memory in Table 8 and with uniform random of 20% connectivity with memory in Table 7. Note that Table 1 can be also compared to Table 5 and Table 2 can be compared to Table 6. Since this paper is about robustness Figs. 5 and 6 present the full distribution of the experiments of Table 7 under these conditions with different degrees of damage. Note that with damage over 1%, the histogram deteriorates dramatically.

The bottom line of all these comparisons is that decreasing connectivity and adding memory to the detector slightly increases the robustness performance with low amounts of damage, but even with low amounts of damage and under all our variants of random connectivity, the networks would fail. That is, the trained but damaged network loss of function was very substantial and in many cases could not perform substantially differently from a random classification (see Figs. 5 and 6).

4.3. Third experiments: varying the architecture of the network

4.3.1. Hand chosen one-hub topology

Since the Maass et al. topology and the uniform random distribution topology showed a high level of vulnerability to any small amount of damage, and since adding memory to the detector helped only marginally to recover from damage in the liquid; we started to create different topologies to test the robustness of the liquid with the same parameters as those set by Maass et al. (see Jaeger, 2001a; Lukosevicius & Jaeger, 2009; Maass, 2002; Maass et al., 2002a, 2002c, 2002d; Natschläger, Maass, & Markram, 2002a; Natschläger, Markram, & Maass, 2002b). One of those topologies is the hub topology that is described in detail in Appendix A. In this case, one can see from Table 9, Figs. 7 and 8, that the robustness was substantially increased. However, under the construction as presented in Appendix A, there can appear substantial disconnected components in the liquid. Moreover, in results not presented in this paper, the signal has weaker persistence, i.e. detectors are able to recognize the signals in a substantially smaller time window.

4.3.2. Small world topologies

The general connectivity in the human brain has been held to have some small world properties (Achard et al., 2006) Algorithm 1 is designed to obtain a hub topology using a more "natural" algorithm thus creating a topology that will be robust to damage in the liquid and to be more "natural" in its construction. One of the properties of the small world is the power-law distribution.

The results of the small world with a power-law distribution (see Table 10, Figs. 9 and 10) were, however, very similar to the Maass topology and to the uniform random topology in the

Table 9
One hub network with memory input to the detector.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	100%	95%	88%	85%	76%	67%
Noisy neurons	100%	97%	91%	86%	70%	62%
Dead and noisy	100%	96%	89%	86%	75%	68%
Generalization	100%	100%	97%	97%	96%	95%

Table 10
Small world with a power-law distribution with memory input to the detector.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	100%	55%	51%	51%	50%	51%
Noisy neurons	100%	79%	58%	53%	50%	51%
Dead and noisy	100%	58%	51%	50%	48%	50%
Generalization	100%	100%	97%	93%	90%	89%

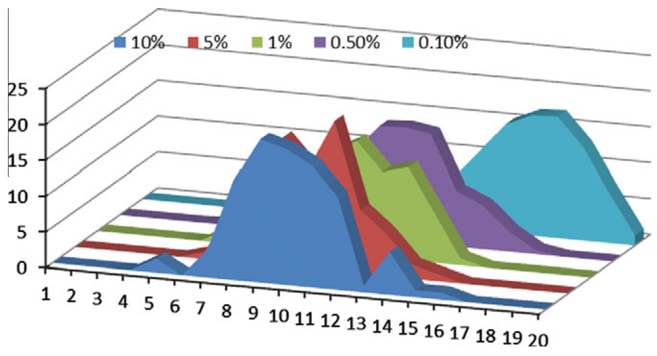

Fig. 9. Histographs of correctness results in LSM networks with different amounts of "dead" neuron damage with small world topology obtained with a power law distribution.

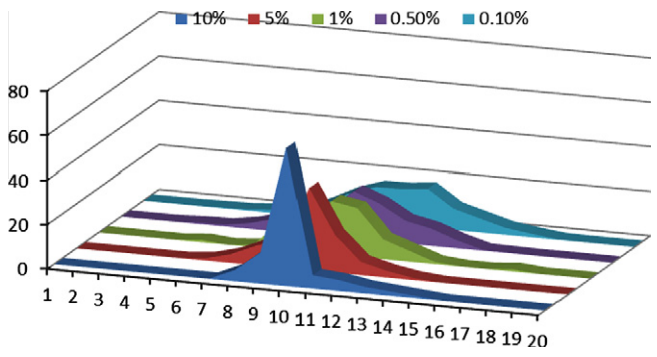

Fig. 10. Histographs of correctness results in LSM networks with different amounts of "noise generator" neuron damage for small world topology obtained with a power-law distribution.

robustness from damage in the liquid. On the other hand, they had improved generalization capability (see Table 10).

Looking closer at the distribution, as can be seen from Fig. 3, Algorithm 1 actually creates a power-law distribution in terms of total connections, but when we separate the connections to input and output connections, we see that while the output has a power law distribution, the input connections have a roughly random uniform distribution.

4.3.3. Small world topologies with double power-law distribution

Accordingly, using Algorithms 1 and 2 we created a double power-law distribution (using the reverse order for input connections and output connections as in Fig. 11). The robustness and the generalization ability was much improved The best results were with a double-power law where the distributions are over distinct neurons and these are the results presented here in Tables 11 and 12 and Figs. 12 and 13 .

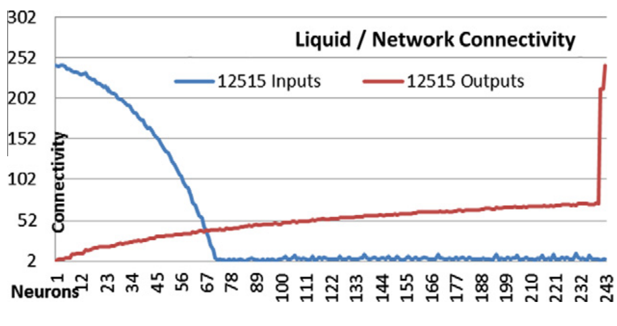

Fig. 11. Connection distribution of small-world with double power-law.

Table 11
Small world with a double power-law distribution with memory input to the detector.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	96%	95%	87%	83%	74%	69%
Noisy neurons	96%	99%	93%	88%	72%	64%
Dead and noisy	96%	97%	89%	84%	70%	66%
Generalization	96%	99%	99%	98%	97%	97%

Table 12
Small world with a double power-law distribution without memory input to the detector.

Damage	Non	0.1%	0.5%	1%	5%	10%
Dead neurons	62%	83%	67%	61%	56%	53%
Noisy neurons	62%	91%	75%	66%	54%	55%
Dead and noisy	62%	86%	69%	65%	52%	55%
Generalization	62%	100%	96%	95%	93%	91%

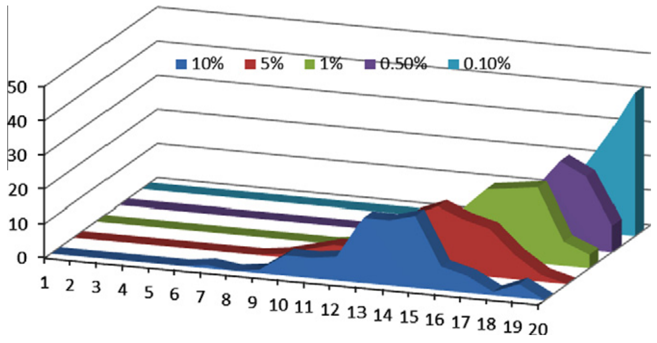

Fig. 12. Histographs of correctness results in LSM networks with different amounts of "dead" neuron damage with small world topology obtained with a double power law distribution.

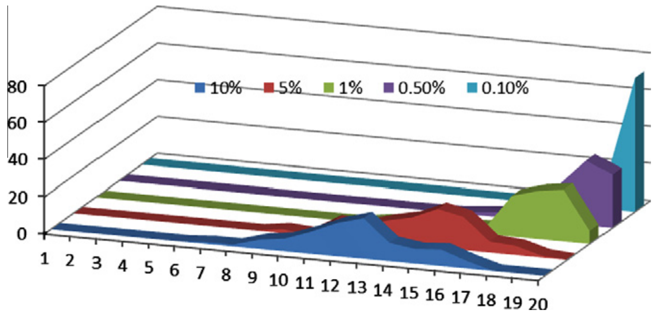

Fig. 13. Histographs of correctness results in LSM networks with different amounts of "noise generator" neuron damage for small world topology obtained with a double power-law distribution.

5. Discussion

In this work, we looked at the robustness of the LSM paradigm and by experimenting with temporal sequences showed that the basic structural set up in the literature is not robust to two kinds of damages; even at small levels of damages.

We also investigated this for various degrees of connectivity. While lowering the average degree of connectivity resulted in decreased sensitivity in all architectures to some extent, the bottom line is that decreased connectivity is ineffective. In addition, it became evident that lowering the connectivity also decreases the strength the network has in representability and, importantly,

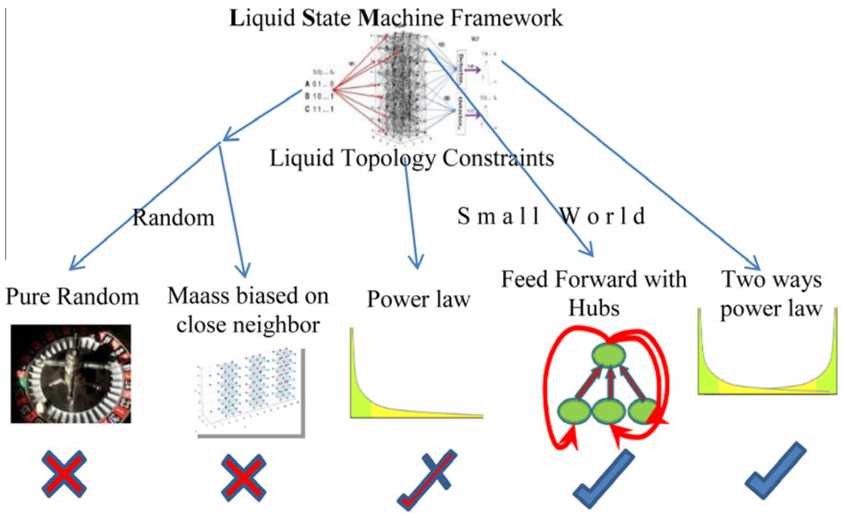
Liquid State Machine Framework
Liquid Topology Constraints
Random
Small World
Pure Random
Maass biased on close neighbor
Power law
Feed Forward with Hubs
Two ways power law

Fig. 14. A graphical summary of the results presented in this paper. The "standard" LSM topologies either uniform or in Maass's original papers are not robust; but small world topologies show an improvement, which is most marked in the case of a two-way power law distribution.

in the persistence of the signal. (That is, a low degree of connectivity causes the activity to die down quickly because of the lack of feedback. Thus the network is bounded in time and cannot recognize an "older" input signal.) Thus we see, as is to be expected from the analysis in Jaeger (2001a, 2001b, 2002) and Maass et al. (2002a) that a higher connectivity gives a larger set of "filters" that separate signals, but on the other hand makes it more sensitive to changes.

In any case, even with low connectivities, the random topology was not robust; nor was the Maass topology. (While not at random levels of identification, as we have seen, e.g. in Tables 1 and 2 it suffered very substantial decays with even small amounts of damages. In addition, other experiments (not shown here) with connectivities below 15–20%, show that the networks do not maintain the trace for very long.)

We also investigated some variants in the kinds of neurons. It seems that the LSM (or "reservoir computing" concept) does not change much vis a vis robustness to internal noise based on these choices.

We did see substantial improvement when supplying a window of time input to the detector rather than an instant of time. However, alone this was not sufficient.

The major affect was changing the topology of connectivity to accommodate the idea of hubs, power law and small world connectivity. Under these topologies, with the best result occurring when we have power law histogram of both input and output connectivity to the neurons with separate neurons as hubs in both directions, the liquids are robust to damages.

6. Conclusions

We have shown experimentally that the basic LSM is not robust to "damages" in its underlying neurons and thus without elaboration cannot be seen to be a good fit for a model for biological computation. (We mention (data not shown here) that this result holds even if training is continued while the network is suffering damage.)

However, choosing certain power law topologies of the connectivity can result in more robust maintenance of the pertinent information over time. A graphical summary of the results for robustness under different topologies is given in Fig. 14.

In the papers (Bassett & Bullmore, 2006; Varshney et al., 2011), a distribution was chosen for biological reasons to allow preference for close neurons. This distribution is superior to the totally random one, but is still not sufficiently robust. Choosing a power law distribution and being careful to making the assignments differently for in and out connectivity proved to be the best. Since this is thought of as a potentially biological arrangement (Barabási & Albert, 1999; Bassett & Bullmore, 2006); LSM style networks with this additional topological constraint can, as of this date, be considered sufficiently biological. Other distributions may also work.

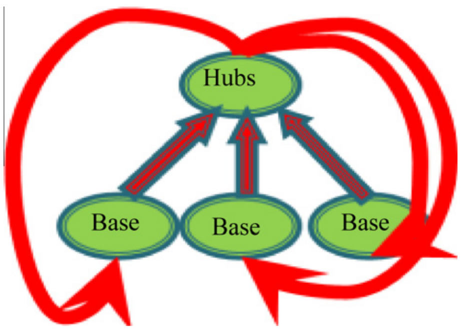

Fig. A1. Hub topology.

Acknowledgements

We want to acknowledge the distinction and support provided by Sociedad Mexicana de Inteligencia Artificial (SMIA) and the 9th Mexican International Conference on Artificial Intelligence (MICAI-2010) in order to enhance, improve, and publish this work.

We thank the Caesarea Rothschild Institute for support of this research. The first author thanks Prof. Alek Vainstein for support in the form of a research fellowship. A short version of this work was presented in the MICAI-2010 meeting (Manevitz & Hazan, 2010) whom we thank for inviting us to write this extended version. We also thank the Maass laboratory for the public use of their code.

Appendix A

The architecture one hub was made as the following:

- Divide all the neurons (240) to groups; the size of each group is randomly chosen between 3 and 6 neurons in one group. Each neuron in the entire group connects to 2 of his neighbors in the same group.
- Choose ¼ of the groups to be hubs and the rest of the groups we will call them the base.
- For 20% connection (that is 11,472 connections) 90% of the connections are from the base groups to the hub groups, 7% are from the hub group to base group and 3% are connections between the hub groups. To accomplish that:
  - Choose (10324 times) random neurons from the base groups and connect each one with a randomly neuron from a hub group.
  - Randomly choose (803 times) a randomly neuron and connect it to a randomly chosen neuron from the base neurons.
  - Connect (345 times) randomly one of the neurons from the hub neurons to anther neuron but from a different group (see Fig. A1).

References

Achard, S., Salvador, R., Whitcher, B., Suckling, J., & Bullmore, E. (2006). A Resilient, low-frequency, small-world human brain functional network with highly connected association cortical hubs. *The Journal of Neuroscience, 26*(1), 63–72. doi:10.1523/JNEUROSCI.3874-05.2006.

Albert, R., & Barabási, A.-L. (2000). Topology of evolving networks: Local events and universality. *Physical Review Letters, 85*(24), 5234–5237. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/11102229>.

Barabási, G. B. A.-L. (2000). Competition and multiscaling in evolving networks. cond-mat/0011029. Retrieved from <http://arxiv.org/abs/cond-mat/0011029>.

Barabási, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. *Science, 286*(5439), 509–512. doi:10.1126/science.286.5439.509.

Bassett, D. S., & Bullmore, E. (2006). Small-world brain networks. *The Neuroscientist, 12*(6), 512–523. doi:10.1177/1073858406293182.

Fern, C., & Sojakka, S. (n.d.). Pattern recognition in a bucket. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.97.3902>.

Gutig, R., & Sompolinsky, H. (2006). The tempotron: A neuron that learns spike timing-based decisions. *Nature Neuroscience, 9*(3), 420–428. doi:10.1038/nn1643.

Hazan, H., & Manevitz, L. M. (2010). The liquid state machine is not robust to problems in its components but topological constraints can restore robustness. In *IJCCI (ICFC-ICNC)* (pp. 258–264).

Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks, 14*(6), 1569–1572. doi:10.1109/TNN.2003.820440.

Jaeger, H. (2001a). *The "echo state" approach to analysing and training recurrent neural networks (No. GMD Report 148)*. German National Research Center for Information Technology. Retrieved from <http://www.faculty.iu-bremen.de/hjaeger/pubs/EchoStatesTechRep.pdf>.

Jaeger, H. (2001b). *Short term memory in echo state networks (No. GMD Report 152)*. German National Research Center for Information Technology. Retrieved from <http://www.faculty.iu-bremen.de/hjaeger/pubs/STMEchoStatesTechRep.pdf>.

Jaeger, H. (2002). *Adaptive nonlinear system identification with echo state networks*. Retrieved from <http://www.faculty.iu-bremen.de/hjaeger/pubs/esn_NIPS02>.

Lukosevicius, M., & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review, 3*(3), 127–149. doi:10.1016/j.cosrev.2009.03.005.

Maass, W. (2002). Paradigms for computing with spiking neurons. In J. L. van Hemmen, J. D. Cowan, & E. Domany (Eds.). *Models of neural networks. Early vision and attention* (Vol. 4, pp. 373–402). New York: Springer.

Maass, W., Legenstein, R. A., & Markram, H. (2002). A new approach towards vision suggested by biologically realistic neural microcircuit models. In *Proceedings of the 2nd workshop on biologically motivated computer vision. Lecture notes in computer science*. Springer. Retrieved from papers/lsm-vision-146.pdf.

Maass, W., & Markram, H. (2002). Temporal integration in recurrent microcircuits. In M. A. Arbib (Ed.), *The handbook of brain theory and neural networks* (2nd ed.. Cambridge: MIT Press.

Maass, W., & Markram, H. (2004). On the computational power of circuits of spiking neurons. *Journal of Computer and System Sciences, 69*(4), 593–616. doi:10.1016/j.jcss.2004.04.001.

Maass, W., Natschläger, T., & Markram, H. (2002a). Computational models for generic cortical microcircuits. In J. Feng (Ed.), *Computational neuroscience: A comprehensive approach*. CRC-Press. Retrieved from papers/lsm-feng-chapter-149.pdf.

Maass, W., Natschläger, T., & Markram, H. (2002b). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation, 14*(11), 2531–2560. Retrieved from papers/lsm-nc-130.pdf.

Maass, W., Natschläger, T., & Markram, H. (2002c). A model for real-time computation in generic neural microcircuits. In *Proceedings of NIPS 2002* (Vol. 15, pp. 229–236). Retrieved from papers/lsm-nips-147.pdf

Maass, W., Natschläger, T., Markram, H. (2002d). A fresh look at real-time computation in generic recurrent neural circuits, Tech. Report, Institute for Theoretical Computer Science, TU Graz, Graz, Austria.

Manevitz, L., & Hazan, H. (2010). Stability and topology in reservoir computing. In G. Sidorov, A. Hernández Aguirre, & C. Reyes García (Eds.), Advances in soft computing. Lecture notes in computer science (Vol. 6438, pp. 245–256). Berlin/Heidelberg: Springer. Retrieved from <http://dx.doi.org/10.1007/978-3-642-16773-7_21>.

Manevitz, L. M., & Marom, S. (2002). Modeling the process of rate selection in neuronal activity. *Journal of Theoretical Biology, 216*(3), 337–343. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/12183122>.

Natschläger, T., Maass, W., & Markram, H. (2002). The "Liquid Computer": A novel strategy for real-time computing on time series. *Special Issue on foundations of information processing of TELEMATIK*, 8 (1), 39–43. Retrieved from papers/lsm-telematik.pdf.

Natschläger, T., Markram, H., & Maass, W. (2002). Computer models and analysis tools for neural microcircuits. In R. Kötter (Ed.), A practical guide to neuroscience databases and associated tools. Boston: Kluwer Academic Publishers. Retrieved from papers/lsm-koetter-chapter-144.pdf.

Pitts, W., & McCulloch, W. S. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology, 52*(1–2), 99–115. discussion 73-97.

Varshney, L. R., Chen, B. L., Paniagua, E., Hall, D. H., & Chklovskii, D. B. (2011). Structural Properties of the Caenorhabditis elegans Neuronal Network. *PLoS Computational Biology, 7*(2), e1001066. doi:10.1371/journal.pcbi.1001066.

Widrow, B., & Hoff, M. (1960). Adaptive switching circuits. 1960 {IRE} {WESCON} Convention Record, Part 4 (pp. 96–104). {IRE}. Retrieved from <http://isl-www.stanford.edu/~widrow/papers/c1960adaptiveswitching.pdf>.