Self-Organizing Echoing Reservoir (SOER)

Shannon Birchell

University of North Florida

Jacksonville, USA

n000180071@unf.edu

*Abstract*— there has been tremendous amount of work done on Neural Networks (NN) over the past 50 years. As time has progressed NN have incurred a vast domain of methodologies as well as successful application in many areas. An area of interest is Recurrent Neural Networks (RNN). This subfield, in itself, has many methodologies but all have the common component call a reservoir. A reservoir is a recurrent network that itself does not learn but the interfacing connections perform learning. Even a more specific kind of RNN has been used in more recent research called the Echo State Network (ESN). In a ESN there are specific properties that the reservoir must possess. A theoretical aspect of ESN reservoir is infinite historical

Keywords—component; formatting; style; styling; insert (key words)

# Introduction

There are several implementation methodologies for neural networks (NN). A known issue with NN is the processing time required when training and classifying. These intensive processing is intrinsic to NN do to complexity and learning techniques such as back propagation.

A subfield of NN is Recurrent NN (RNN) and differs in that there is a reservoir. The reservoir has a sparsely cyclic connected structures that have properties paralleling the biological counterpart. We can further drill down to a specific implementation of RNN called Echo State Network (ESN).

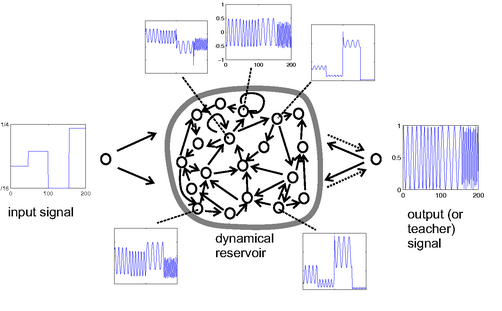
ESN does not change the reservoir in the sense of training. This network learns only by manipulating the output weights that are connected to the reservoir. The reservoir must be “tuned” in the sense there has to be a configuration that simulates a dynamic space. To assist with dynamic behavior Random NN (RandNN) have been introduced to inject excitation to the system. This dynamic space requires control to reduce feed forward complications which is done through dampening. Further, to guarantee a dynamic reservoir, the concept of spectral radius is used to guide configuration.

The need for spectral radius reduces generality and requires more thorough development of the reservoir as opposed to random creation. In this research an attempt to prove that a NN with exciter and inhibitory neurons can train the reservoir of an ESN, this can provide a dynamic system eliminating the need for spectral radius and damping.

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

# Discription of Echo State Reservoir

Echo State Reservoir



# New Reservoir traning method

A widely used training method for NN is gradient decent. When a neuron is expected to adhere (Supervised Learning) to a state the input connections that facilitate this behavior are strength and ones that do not are weakened. Therefor over several training sessions the neuron attempts to optimize itself through weight changes. Overtime the NN becomes stable and the outputs converges to a predictive model[[1]](#footnote-1).

In SOER a similar methodology is performed but from a different perspective. As gradient decent is performed by comparing desired behavior SOER learns by pushing the desired behavior from sub neurons. Although the desired behavior is different as the goal of this training is different. Therefore activated child neurons tend to make the inactive parent neuron fire after training. Inactive children are of an active parent are considered non-productive and connections removed. Finally when the parent is active as well as the child, the connection is slightly weakened (parent may be its child). This results in a positive feed forward learning technique with aspects of network dampening and connectivity reduction.

Figure - Algorithm Pseudo code

nw = Create fully connected network of n nodes

For P impressions

impression = get random impression

For T training time

func train(impression)

end training

end impressions

func train(impression)

for each node in network

if active node

for all active children

//weaken connection

func ChangeWeight(weight,-small)

for all inactive children

//remove connection

func ChangeWeight(weight,-large)

else

for all active children

//strengthen connection

func ChangeWeight(weight,large)

end for each

end func

func changeWeight(connectionWeight, learningRate)

connectionWeight =

connectionWeight+learningRate\*connectionWeight

end func

There are several implementation methodologies for neural networks (NN). A known issue with NN is the processing time required when training and classifying. These intensive processing is intrinsic

# Results

Testing was done on randomly generated networks of 100 neurons. The networks are fully connected and the weights normalized. The following learning rates is used:

Anti-learning

Active links -0.1

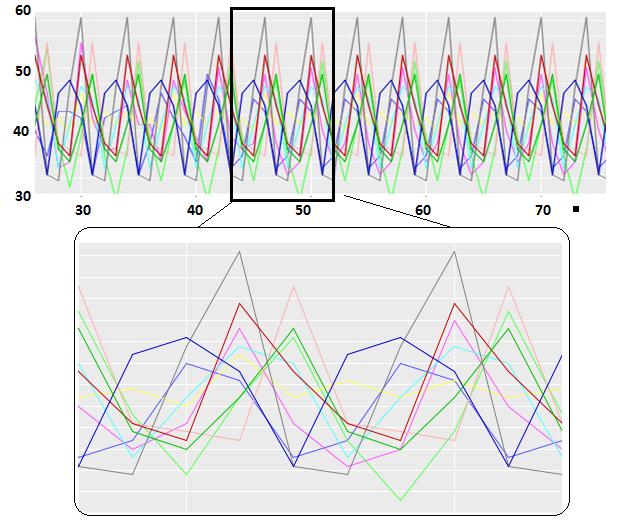
Inactive links -10

Learning rate of 10

The network was impressed with a randomly generated activation vectors representing all nodes within the network. This was done 50 times with an each impression lasting for 5 time units.

In Figure 2 - Activations counts by time, a window of time from 20 to 80 time units is shown (horizontal axis). The number of activations at each time unit is show to be between 30 and 60 (vertical axis). Reviewing the graph one can see evidence of cyclic behavior where some cycles big as 5 time units. Also it is evident that the cycles are not offset of each other and thus have unique behaviors for each randomly generated activation impression. Further the network is energy stable and will oscillate indefinitely.

Figure - Activations counts by time



Referring to Table 1 - Testing untrained vs trained mean activation counts, a comparison to the control model is performed. This is done by aggregated activation counts into buckets of size 5. As shown the model are significantly different for all time counts and models. Training has changed the models uniquely and do have behavior different than the control.

Table - Testing untrained vs trained mean activation counts

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T-test of mean  df=4 | Time: 0-5 | | Time: 6-10 | |
| Average | P-value | Average | P-value |
| Trained 1 | 47.6 | 0.01 | 44.4 | 0 |
| Trained 2 | 48.4 | 0.013 | 46.2 | 0 |
| Trained 3 | 51 | 0.021 | 54.2 | 0 |
| Trained 4 | 52.2 | 0.005 | 52.6 | 0 |
|  | Time: 11-15 | | Time: 16-20 | |
| Trained 1 | 44 | 0 | 45.6 | 0 |
| Trained 2 | 47.6 | 0 | 48.6 | 0 |
| Trained 3 | 56 | 0 | 55.2 | 0 |
| Trained 4 | 54.2 | 0 | 52.6 | 0 |

Table - Correlation of activation counts by time

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| PACF | Lag in time units | | | | |
| 1 | 2 | 3 | 4 | 5 |
| Trained 1 | -0.139 | -0.633 | -0.146 | 0.895 | -0.109 |
| Trained 2 | -0.632 | 0.42 | -0.661 | 0.891 | -0.646 |
| Trained 3 | 0.002 | -0.719 | -0.11 | 0.494 | -0.353 |
| Trained 4 | -0.597 | -0.126 | -0.683 | 0.272 | 0.031 |
|  | 6 | 7 | 8 | 9 | 10 |
| Trained 1 | -0.569 | -0.175 | 0.836 | -0.104 | -0.546 |
| Trained 2 | 0.384 | -0.628 | 0.866 | -0.614 | 0.367 |
| Trained 3 | -0.29 | 0.126 | 0.318 | -0.089 | -0.079 |
| Trained 4 | 0.073 | -0.243 | -0.025 | -0.009 | -0.113 |

Table - Number active at time 100

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Activation  Count | Impulse Number | | | | |
| 1 | 2 | 3 | 4 | 5 |
| Trained 1 | 61 | 58 | 57 | 56 | 54 |
| Trained 2 | 62 | 61 | 59 | 58 | 57 |
| Trained 3 | 54 | 54 | 53 | 53 | 53 |
| Trained 4 | 58 | 52 | 49 | 47 | 46 |
|  | 6 | 7 | 8 | 9 | 10 |
| Trained 1 | 53 | 53 | 53 | 51 | 50 |
| Trained 2 | 57 | 56 | 56 | 56 | 56 |
| Trained 3 | 53 | 52 | 51 | 49 | 49 |
| Trained 4 | 43 | 41 | 41 | 36 | 36 |

# Conclusion

Identify applicable sponsor/s here. If no sponsors, delete this text box (*sponsors*).

##### References

[1] Basterrech, S., & Rubino, G. (2012). *Echo State Queueing Network: a new reservoir computing learning tool.* France: INRIA-Rennes.

[2] Jaeger, H. (2002). *Short Term Memory in Echo State Networks.* Germany: GMD report - Forschungszentrum.

[3] Jaeger, H. (2013). *A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "Echo Network" approach.* Bremen: Fraunhofer Institute for Autonomous Intelligent Systems.

[4] Lukosevicius, M. (2012). *A Practical Guide to Applying.* Bremen: Springer.

[5] Schmidt, T. P., A.Wiering, M., Rossum, A. C., Elburg, R. A., Andringa, T. C., & Valkenier, B. (2007). *Robust Real-Time Vowel Classification.* Netherlands: University of Groningen.

1. This assumes the model is capable of solving the decision space [↑](#footnote-ref-1)