Replicator Neural Network (Novelty Detector)

Razin Zuhair
Dept. of Electrical and Computer Engineering, M.S
University of Texas at Arlington
razin.zuhair@mavs.uta.edu

***Abstract*- This paper gives you information about the behavior and performance of a Replicator Neural Network with regards to removal of random Gaussian Noise and how the input signal is ideally replicated on the output side. This paper also shows the self-organizing behavior of the RNN and the encryption and decryption mechanism being implemented in order to form an optimal-compression replicator network.**

Keywords- Replicator Neural Network(RNN)

TABLE OF CONTENTS

I. INTRODUCTION

Self-organizing behavior has been long recognized as one of the necessary elements of intelligence, thus it is desirable that an artificial system that mimics biological intelligence be able to perform both spatial and temporal operations. Example includes Self learning in Arrays[1]. This behavior helps in forming a Neural Network so intelligent, that it can mimic the functioning of encoder and decoder. The RNN gathers the input data signal and transmits it from the input stage to the hidden layers in the form of rapid bursts of data packets which would be un-decipherable by any other exterior intruder trying to harm the network. The RNN then receives these bursts at the output nodes and the subsequently operated data packets are then retraced back into it's original form by removing the associated noise and in this way signal re-generation takes place and a RNN is then successfully implemented.

II. MODELING & SIMULATION

Here we consider an RNN whose number of inputs and number of outputs are the same, the number of nodes in the first and the third hidden layer are the same with the activation function as tanh. The RNN architecture is shown in figure 1. The activation function for the third hidden layer is modeled as follows:

$$S3(\theta) = [1/2 + \tfrac{1}{2}(N-1)\sum_{j=1}^{N-1} \tanh[a(\theta - j/N)]$$

The simulation is done by a code written in the MATLAB simulation software and is implemented in such a way that it works for any number of desired inputs, outputs and also the nodes in the hidden layer mechanism. The weights and biases at every node in the RNN are supposedly defined and are put into effect. The aforementioned activation function forms a smooth stairstep function with N treadles essentially quantizes the vector of middle hidden layers outputs into M = N^n grid points[2]. The activation function is shown in figure 2.

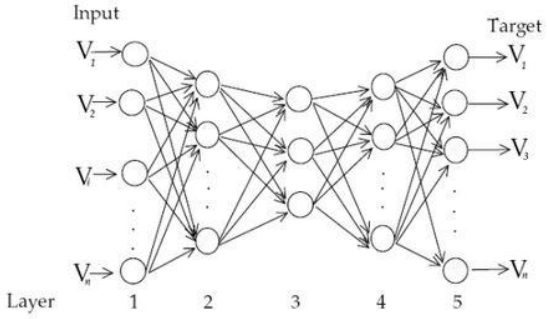


Figure 1

A Cost function is predicted as a reference and the error is minimized over a range of iterations. The built-in MATLAB function used here is 'fminunc', which stands for Minimum of Unconstrained Multivariable Function and the term unconstrained means that there is no restriction is placed on the range of x. Although several applications in image and speech processing have used the RNN for its data compression capabilities [3,4], we believe the current study is the first to propose its use as a signal recreation tool by

means of encryption and decryption. The RNN we use here is a feed-forward multi-layer perceptron with three hidden layers sandwiched between input and output layers.

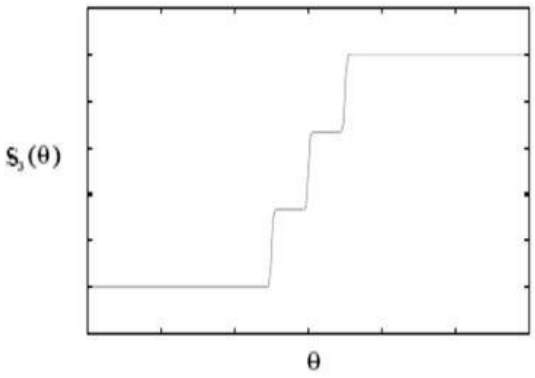
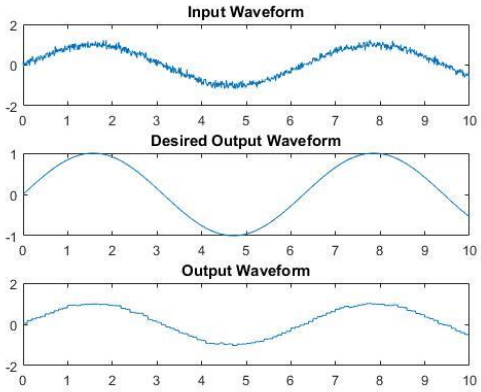


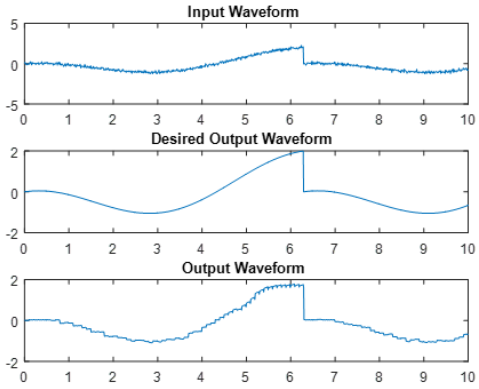
Figure 2

III. SIMULATION, RESULTS AND ANALYSIS

A custom input signal wave which was specifically designed for this project was implemented as part of my research and the signal was added with recurrent noise and fed into the RNN. The signal was successfully recreated at the output layer and the noise was removed depending on the performance of the RNN. Following are some handy graphics that provide concrete evidence to the claims:
A generalized Sine wave as input yielded the following the results:



As part of my research, I fiddled around by changing various parameters and generated a custom input signal which was a mixture of Cosine wave and Tangent wave and the results were:



In both the cases, it was observed that the signal was not completely regenerated as the efficiency of the RNN was not at its best but further work in the field would lead to RNNs with optimal replication capabilities.

IV. CODE

The MATLAB Simulation model and code is followed at the end of this paper.

V. FUTURE WORK/CONCLUSION

Based on the work done to date in terms of Neural Networks and their performance over the years, every advancement leaves potential for furthermore work in the field. Different sets of networks involving complex data structures such as reading animated Gif images would make an interesting spectacle in the future. Finally, there were some observations during this experiment:

- Reducing the value of a in the activation function to a lower value leads to it becoming similar to tanh function.
- Nominal value of a leads to a much better and quicker signal recreation.
- The processing time of the RNN varies from one version of MATLAB to the other.

VI. ACKNOWLEDGEMENT

I would like to thank Dr. Niestroy for expertly guiding the students through this project and his unwavering enthusiasm for Controls that kept the class constantly engaged with Research owing to his personal generosity as well. I would also like to express my deep gratitude towards him for providing the students with a platform to learn in a way that replicates the functioning of the industry.

VII. REFERENCES

[1] Janusz A. Starzyk, Zhen Zhu and Yue Li (2005, August).*Associative Learning in Hierarchical Self Organizing Learning Arrays*. Available: http://www.ohio.edu/people/starzykj/network/Research/Papers/Associative%20learning%20in%20SOLAR%202005.pdf

[2] Robert Hecht-Nielsen (1995, September). *Replicator Neural Networks for Universal Optimal Source Coding*. Available: https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxtaWNoYWVsbmllc3RveXxneDo3MDA0N2IxNmQ2ZmY4Mjdm

[3] Simon Hawkins, Hongxing He, Graham Williams and Rohan Baxter. (2002, February). *Outlier Detection Using Replicator Neural Networks*. Available: http://togaware.com/papers/dawak02.pdf

[4] Janusz Starzyk and Jing Pang.(2010, June). *Evolvable Binary Artificial Neural Network For Data Classification*. Available: http://www.ohio.edu/people/starzykj/Webcad/Papers/Recent%20conferences/EvolvableBinary.PDF