An Analytical Neural Network for Arithmetic Logic Unit of Microprocessors*

***Mainak Basu*[1]**
***Abhishek Pandey*[2]**
***Abhijit Mustafi*[3]**

ABSTRACT:

In this article, we describe a new approach for analog computation using Deep Learning Neural Networks to perform multiple types of mathematical operations within one machine cycle. The weights and biases of the pre-trained ANNs can be used to fabricate the network in hardware either using a FPGA system or discrete components. The trained network has shown the potential to execute simple and complex mathematical operations alongside basic signal processing within an acceptable error limit, without retraining or reconfiguration.

1. INTRODUCTION:

Digital computation offers significant advantages over analog computation, as any mathematical problem can be broken into a set of Boolean primitives that can be operation on by the Arithmetic Logic Unit (ALU) or the Floating Point Unit (FPU) of the microprocessor. Several microprocessors/microcontrollers implement also implement several modifications to operate on more complex mathematical functions in a reduced period of time. Despite these advantages, analog computation offers a greater speed of operation, using reduced number of components but requires application specific circuits for each operation.

In this article, we propose a radical redesign of the Arithmetic Logic Unit (ALU) or the Floating Point Unit (FPU) of the microprocessor/microcontroller. Unlike conventional circuits which use discreet logic sections to perform a specific task, we propose the use of a hardware based Artificial Neural Network (ANN) circuit, which can perform several mathematical operations without retraining or changes in

components. ANNs have been fabricated in hardware to perform a variety of tasks including adaptive computation [1] and increasing the interconnection density network-on-chip routers [2].

---

*This Article is reprinted from the IISRR-IJR, Vol-1, Issue-2;
[1&2]Dept. of Electronics & Communication Engineering, Birla Institute of Technology, Mesra, Ranchi, India;
[3] Dept. of Computer Science Engineering, Birla Institute of Technology, Mesra, Ranchi, India;
e-mail: mainak.basu87@gmail.com

There have also been several attempts to build mathematical processing circuits using Neural Networks [3, 4]. Several articles have also been fabricated to simulate the properties of artificial neurons [5 - 7]. In order to retain the training and adaptive capability of a neuron, the networks have been developed with FPGA kits, which enables a simplified implementation of the ANN, but still uses digital logic to employ the mathematical operations. There have also been several attempts to develop standalone circuits using more conventional hardware [8 - 10].

In the present work, the Neural Network is trained in an application specific manner, using a data pre-processing that is implemented before the training takes place, hence dimensionally increasing the data that is to be input into the ANN. The construction of the ANN is made of standard artificial neurons utilizing tan-sigmoid and linear transfer functions in the hidden and output layers respectively. This also enables simpler circuitry, using a reduced number of components. Once the network is trained, the weights and the biases can be used to construct the electronic circuits for the individual neurons. Simulations have been performed to test several deep learning ANN architectures with two hidden layers employing a maximum of 20 neurons per layer. A single ANN having two hidden layers, trained with the application specific method as mentioned above to perform five mathematical operations (addition, subtraction, multiplication, division and power) alongside denoising of sine, cosine and gaussian profile signals. It has also been necessitated to develop a appropriate analog logic that can be used for practical implementation on a circuit. Efforts are currently underway to develop a practical implementation of the Neural Network circuit.

2. APPLICATION SPECIFIC TRAINING METHOD:

Every ANN is trained for a specific application and hence once trained is incapable of operating successfully for other purposes. Digital circuits reduce any mathematical operation into a set of Boolean primitives, which allows it represent any mathematical operation as a sequence of Boolean addition. Thus, this method only requires the hardware implementation of a Boolean adder. On the contrary, Analog processors require a specific circuit for each separate operation type. Thus the objective has been to train the ANN to operate for several applications, without retraining or reconfiguration. The training set, containing the data vector and the targets of an ANN can be shown as:

(1)

Where w_i is the training vector containing 'm'-attributes, and z_i is the training target. All other symbols have their usual meaning.

The modified training vector after data pre-processing can be shown as:

(2)

Where a_k and b_l are the extra variables called the 'Decoding Variables' or 'DV'.

These variables cycle between various values and the combination of the different values, creates discrete training data sets with the same training vector values. This allows one ANN to train for multiple requirements and generate a trained network that with more post-processing may allow for analytical decision making in the field of intelligent electronics.

3. SIMULATION:

Simulation was performed simultaneously using the Matlab Neural Network toolbox and the Neuroph Studio software. In order to minimize the number of components that are required, several simulations were performed, to determine the best possible Network architecture as shown in Fig. 1.

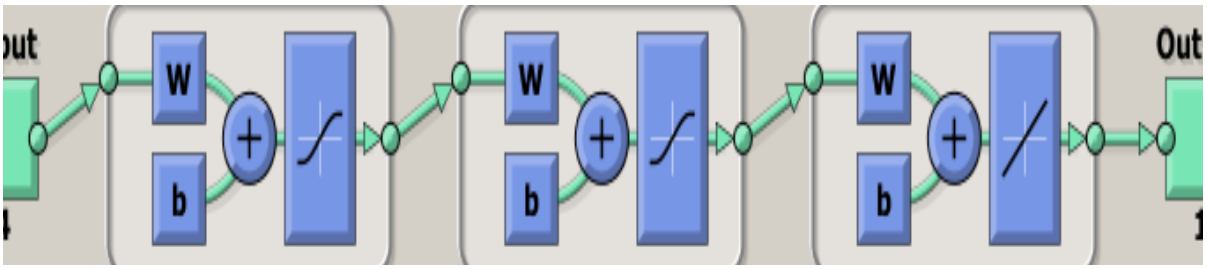


Figure 1: The architecture of the ANN using two hidden layers containing 20 neurons each

The network chosen was a simple Feed Forward Neural Network, for simplicity of both implementation and further hardware implementation. The values assumed by the DVs are as given in Table 1.

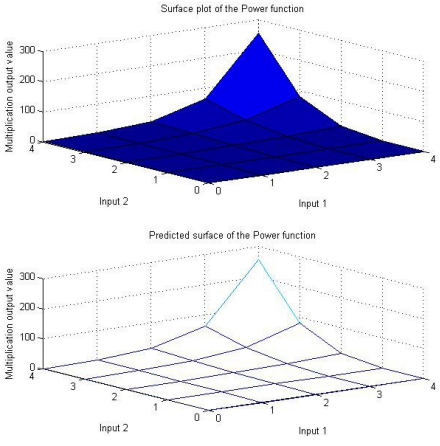
Table 1: Decoding Variable Values and the corresponding mathematical operations

Sl. No.	Decoding Variables		Mathematical Operation
	a_k	b_l	
1.	1	0	(+) Addition
2.	2	0	(-) Subtraction
3.	3	0	(*) Multiplication
4.	4	0	(/) Division
5.	1	1	(^) Power (upto 4)
6.	0	1	Denoising - Sine
7.	0	2	Denoising - Gaussian

The present network has been trained to perform the five mathematical operations (addition, subtraction, multiplication, division and power) alongside de-noising of Sine and Gaussian signals.

4. Results and Discussions:

Several Networks were trained for the above mentioned purpose, and the best network for the current problem was found to be a two hidden layer Deep Learning Neural Network having 20 neurons each in the hidden layers. The proposed network terminated its training with a validation error of < ~10^{-10}. The prediction accuracy of the trained ANN is shown in the three-dimensional surface plots of the respective functions in Figs. 2-4.



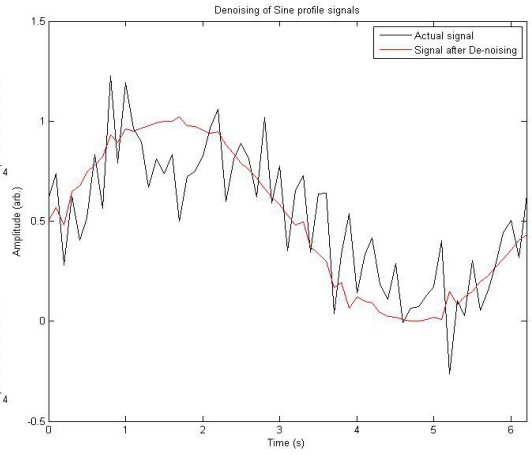


Figure 4: The 3D surface plot of the power function **Figure 5: De-noising of the Sine profile function**

Shown above are a few of the representative functions that the ANN is capable of interpreting depending upon the combination of the Decoding Variables. Hence it seems that a properly trained ANN, can be used to perform a multiple number of tasks on the same data set or multiple tasks on multiple data sets. As the ANN performs the necessary computation within one cycle, unlike digital systems, that require increased number of cycles for more complex operations. The trained network has been successful in performing both mathematical and signal processing operations, without retraining and/or reconfiguration. This serves to indicate that a single

ALU platform can perform the roles of both a normal and a Digital Signal Processor, without the addition of excess hardware components. Also, one network, can be properly trained for performing highly complex mathematical operations, within one machine cycle. The lag of the hardware ANN circuit shall determine the maximum frequency of operation for the processors developed using this type of ALUs. Such ALUs can be implemented with standard FPGA kits which are available today, but their speed of operation would be severely restricted.

5. CONCLUSION:

We have simulated the capability of a singular ANN to perform multiple types of mathematical and simple signal processing operations, preferably within one machine cycle. The trained network was a Deep Learning Neural Network with two hidden layers having 20 neurons in each layer. The Network has been able to successfully perform both the mathematical operations and de-noising of sine profile signals. This makes the aforementioned ANN, a suitable replacement for current digital ALU systems, in terms of advantages gained in power to computational throughput and number of components used. Efforts are currently being made to fabricate the circuit of the ANN to test its characteristics and error tolerance and also further designs for more advanced networks to do the same.

References:

[1] B. E. Boser, E. Sackinger, J. Bromley, Y. Le Cun, and L. D. Jackel, 1991, "An analog neural network processor with programmable topology," *Solid-State Circuits, IEEE Journal of,* 26, pp. 2017-2025.

[2] S. Carrillo, J. Harkin, L. McDaid, S. Pande, S. Cawley, B. McGinley*, et al.*, 2012, "Advancing interconnect density for spiking neural network hardware implementations using traffic-aware adaptive network-on-chip routers," *Neural networks,* 33, pp.42-57.

[3] W. A. Fisher, R. J. Fujimoto, and R. C. Smithson, 1991, "A programmable analog neural network processor," *Neural Networks, IEEE Transactions on,* 2, pp. 222-229.

[4] J. Choi, S. H. Bang, and B. J. Sheu, 1993, "A programmable analog VLSI neural network processor for communication receivers," *Neural Networks, IEEE Transactions on,* 4, pp. 484-495.

[5] M. Pearson, I. Gilhespy, K. Gurney, C. Melhuish, B. Mitchinson, M. Nibouche*, et al.*, 2005, "A real-time, FPGA based, biologically plausible neural network processor," in *Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005*, ed: Springer, pp. 1021-1026.

[6] E. Z. Mohammed and H. K. Ali, 2013, "Hardware Implementation of Artificial Neural Network Using Field Programmable Gate Array," *International Journal of Computer Theory and Engineering,* 5.

[7]S. Cawley, F. Morgan, B. McGinley, S. Pande, L. McDaid, S. Carrillo*, et al.*, 2011, "Hardware spiking neural network prototyping and application," *Genetic Programming and Evolvable Machines,* 12, pp. 257-280.

[8] R. C. Frye, E. A. Rietman, and C. C. Wong, 1991, "Back-propagation learning and nonidealities in analog neural network hardware," *Neural Networks, IEEE Transactions on,* 2, pp. 110-117.

[9] C. S. Lindsey and T. Lindblad, "Survey of neural network hardware, 1995, " in *SPIE's 1995 Symposium on OE/Aerospace Sensing and Dual Use Photonics*, pp. 1194-1205.

[10] E. van Keulen, S. Colak, H. Withagen, and H. Hegt, "Neural network hardware performance criteria, 1994, " in *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, pp. 1955-1958.