Computing, Information Systems, Development Informatics & Allied Research Journal
Vol. 7 No. 3.September, 2016 - www.cisdijournal.net

Evaluation of Performance of Deep Neural Networks on Very Small Datasets Using a Modified Hyperbolic Tangent Activation

V.I.E Anireh
Department of Computer Science
Rivers State University
Port-Harcourt, Nigeria
anireh.ike@ust.edu.ng

***E.N. Osegi**
Department of Information Technology
National Open University of Nigeria
Lagos State, Nigeria
emmaosegi@gmail.com

ABSTRACT

Several researchers have proposed several techniques for improving the run-times or speed of standard feed-forward neural networks. This research paper proposes the use of a modified hyperbolic tangent activation function. The performance and importance of this activation function is demonstrated on training standard feed-forward neural networks with very deep architectures and using small dataset examples. Typical evaluation measures are used including the run-time and mean squared error. The results are compared with the standard hyperbolic tangent activation and a new activation called the exponential linear unit. The results indicate that MODHTAN can outperform the other activations with very low number of epochs.

Keyword - Deep neural networks, epochs, hyperbolic tangent activation, neural activations, vanishing gradient problem

CISDI Journal Reference Format
Anireh, V.I.E. & Osegi, E.N. (2018): Evaluation of Performance of Deep Neural Networks on Very Small Datasets Using a Modified Hyperbolic Tangent Activation. Computing, Information Systems, Development Informatics & Allied Research Journal. Vol 8 No 1. Pp 1-6. Available online at www.cisdijournal.net

1. INTRODUCTION

The challenge of standard feed-forward networks is one that still persists till this day. Though recent progresses have been made in computer architectures to facilitate computing speed, it still remains a requirement and a challenge for researchers to develop neural network algorithms and theories for standard grade computers and using the bare-bones of such algorithms and theories.
Previous work on the use of modified hyperbolic tangent activation function (MODHTAN) showed that lower run-times compared with the standard hyperbolic tangent activation function (HTAN), and the exponential linear unit (ELU) is achievable for shallow back-propagation trained neural networks trained on very small datasets. However, back-propagation method can be very slow in learning, particularly when the gradients become very small. Thus, faster methods have been suggested (see Reed et al, 1998). This paper introduces the MODHTAN activation and compares the performance with the popular HTAN and a recent activation called ELU in a feed-forward neural network trained by the very popular back-propagation algorithm. We have applied this architecture to a very small dataset; by a small dataset, we mean set of example inputs and corresponding targets less than or equal to 10 and greater than or equal to 4. Our primary motive here is to see if appreciable learning at reasonable speeds can still be obtained even with very deep neural architectures. In this paper, we considered using a deep-neural network with 15 hidden layers; we have also scaled the inputs to very small values to increase the learning difficulty – very small values makes it difficult for such neural networks to learn at faster rates.

2. RELATED LITERATURE

Over the years, the challenge of improving the speed or training times of standard (back-propagation trained) feed-forward artificial neural networks (ANN) with very deep architectures have persisted. Some promising techniques have been proposed to speed-up training times such as careful weight initialization and optimal initialization of the synaptic coefficients (Yam and Chow, 2000; Yam and Chow, 2001); the use of specialized neural architecture based on the principle of constant error carousel to eliminate vanishing/exploding gradients (Hochreiter, 1991; Hochreiter and Schmidhuber, 1997) and the use of binarized neural networks (Courbariaux et al, 2016; Hubara et al, 2016).

Other approaches that have hardware related features include the exploitation and optimization of Single Instruction Multiple Data (SIMD) to improve performance and speed of neural networks implemented in Central Processing Units (CPUs), (Vanhouke et al, 2011), in Graphic Processor Units (GPUs), (Scanzio et al 2010a; Scanzio et al, 2010b; Takizawa et al, 2009) and in Field Programmable Gate Arrays (FPGA), (Tiware and Khare, 2015). While these techniques have been useful in many benchmark applications, it still brings with it the challenge of increased complexity, unnecessary mathematical rigor, and cost. Interestingly, previous research work has shown the importance of careful selecting and monitoring the performance of different types of neural activations particularly for deep neural networks employing the standard back-propagation feed-forward neural networks (Glorot and Bengio, 2010). This has instigated further research on more cost-effective and less complex solutions. In this research, we explore the potential of an adaptive hyperbolic tangent activation with a modifiable exponential function that have been introduced earlier in (Anireh and Osegi, 2017), as a possible solution to the slow running speed found in deep-neural network when trained with very small datasets.

3. METHODOLOGY

3.1 Modified Hyperbolic Tangent Function with Adaptive Normalization

The modified hyperbolic tangent activation (MODHTAN) have been developed in (Anireh and Osegi, 2017) is largely based on a modification to the Euler number, 'e'. This modification is based on a so-called Real Number Formula (RNF) designed in (Osegi and Anireh, 2016). The MODHTAN is presented in Eq.1 while the RNF model is given in Eq.2.

$$\begin{cases} f_1 + f_2 + f_3, & f_1 = \left(\dfrac{k_o}{1 + RNF^{-2*x_1}}\right) - 1, \quad x_1 = \dfrac{x}{(x + offset_1)} * (x \geq x_{cutoff}) \\ & f_2 = \left(\dfrac{k_o}{1 + RNF^{-2*x_2}}\right) - 1, \quad x_2 = \dfrac{x}{(x + offset_1)} * (x \leq -x_{cutoff}) \\ & f_3 = \left(\dfrac{k_o}{1 + RNF^{-2*x_3}}\right) - 1, \quad x_3 = x * (x \geq -x_{cutoff} \vee x \leq +x_{cutoff}) \end{cases} \tag{1}$$

.where,

While the RNF is described as:

$$RNF_o = \left(\frac{a - n}{a - (m + x)}\right)^a \approx e^x \tag{2}$$

$$for \quad a = 10^7, n = 1, and \ m = 1$$

x_1, x_2 are the constrained activations and f_1, f_2 the corresponding adaptive functions.

It has been observed in (Anireh and Osegi, 2017) that RNF_o is cheaper to construct than e^x and the derivative of HTAN can be used without any loss in precision. The parameters $offset_1$ and x_{cutoff} in turn represents the adaptive scaling (normalization) and threshold factors respectively. It is important to note that $offset_1$ provides immunity to the vanishing gradient problem (see Hochreiter and Scmidhuber, 1998; Anireh and Osegi, 2017) and is defined by x_{cutoff}.

3.2 Competing activations

The exponential linear unit (ELU) and Hyperbolic Tangent Activation (HTAN) have already been described in (Anireh and Osegi, 2017). Readers are referred to the literature for more details on the functionality of these activations. Further details on the ELU can also be found in (Clevert et al, 2015).

4. EXPERIMENTS

4.1 Training and Testing Environment

Experiments were carried out using a Zinox® system using Pentium(R) Dual-Core processor, clock speed at 2.30GHz with 2GB Random Access Memory. The program was developed and run in Matlab ® 7.5 environment running on a Windows 7 Operating System.

4.2 Datasets

Two datasets are considered. The first being a very small hypothetical dataset based on the popular multiplication time table operation and the second being a very small subset of a real world data containing price fluctuations in premium motor spirit (PMS) demand; the second data is obtained from the Nigerian Bureau of Statistic (NBS) website.

4.3 Training and Test Scheme

We train the network using the three different activation functions for a maximum of 500 epochs and for several trial runs taking note of the performance goal; we consider only those results for which the performance goal is met and we record the number of epochs taken by the different activations in addition to the prediction error; here the epochs refer to the number of training iterations before learning converges and the performance goal refers to a pre-specified ANN dependent goal (used as a stopping criterion) that must be met just before convergence. We continue these experiments until the desired number of tests (10 example tests) is attained. It is important to note that for each test example result obtained, it takes about 10 trials to achieve the desired performance goal.

We used the back-propagation feed-forward neural network trained with the trainlm (Levenberg-Marquardt back propagation) function in MATLAB and an adaptive normalized constraint described in the previous section. The key parameters of the Neural Network employed are summarized in Table 1.

5. RESULTS AND DISCUSSION

5.1 Hypothetical Dataset

The run-times and error values of each function using a hypothetical dataset composed of a sub-set of the multiplication table operation (2-times Table) are shown in Tables 2 to 4. From the results it is clear in terms of the error accuracy that all activations performed well i.e. the error values are relatively very small for all activations. It can also be seen clearly from these tables and from Figure 1 that MODHTAN outperforms all the other activations.

5.2 Real Dataset:

The run-times and error values of the MODHTAN in an ANN using a subset of premium motor spirit (PMS) dataset obtained from the Nigerian Bureau of Statistic (NBS) website is as shown in Table 5. Apart from the second and ninth example tests, the performance of MODHTAN activated neural network was appreciable showing that very low training epochs are also attainable. In a nutshell, it goes to show that MODHTAN activation is promising for neural network applications trained with very small datasets.

6. CONCLUSION

Algorithms using the exponential function for computing its activations may lead to slow network in practical real world applications. In this research work, we have presented our experimental findings on very small datasets and using also deep neural architectures for training. We show that the MODHTAN activation is indeed superior to the other similar activations and can be a promising activation for standard deep feed-forward neural networks trained with very small examples of real world data. We have also shown that it is indeed still possible to obtain appreciable error results on real world data. Future research directions should be focused on reducing the number of trials required for convergence attainment which is still yet to be explored. In addition, research should also be directed at validating the effectiveness of the proposed MODHTAN activation in embedded hardware.

Table 1 Used Neural Network Parameters.

Number of Hidden Neurons	Number of Epochs	Learning Parameter
2	500	Gradient descent with momentum

Table 2 Simulation Run for the Multiplication-Table Matching Problem Using HTAN Activation; set goal is $1.0 * 10^{-07}$

Trial Case	Mean Squared Error	No. of Training Epochs
1	$5.35 * 10^{-06}$	79
2	$5.89 * 10^{-06}$	57
3	$6.06 * 10^{-06}$	136
4	$1.75 * 10^{-06}$	19
5	$6.25 * 10^{-06}$	61
6	$6.38 * 10^{-06}$	377
7	$3.46 * 10^{-06}$	24
8	$6.12 * 10^{-06}$	265
9	$1.12 * 10^{-09}$	7
10	$6.06 * 10^{-06}$	355

Table 3 Simulation Run for the Multiplication-Table Matching Problem Using MOD-HTAN Activation; set goal is $10^{-07}10^{-07}$

Trial Case	Mean Squared Error	No. of Training Epochs
1	$2.80 * 10^{-06}$	45
2	$3.43 * 10^{-08}$	50
3	$4.57 * 10^{-09}$	45
4	$1.13 * 10^{-08}$	104
5	$9.39 * 10^{-10}$	45
6	$1.91 * 10^{-09}$	22
7	$9.62 * 10^{-08}$	25
8	$1.58 * 10^{-07}$	17
9	$1.22 * 10^{-08}$	38
10	$5.09 * 10^{-08}$	35

Table 4 Simulation Run for the Multiplication-Table Matching Problem ELU Activation; set goal is $1.0 * 10^{-07}$

Trial Case	Mean Squared Error	No. of Training Epochs
1	$3.12 * 10^{-21}$	500
2	$3.84 * 10^{-22}$	20
3	$9.57 * 10^{-21}$	90
4	$2.06* 10^{-09}$	400
5	$7.52 * 10^{-08}$	350
6	$1.15 * 10^{-07}$	60
7	$9.03 * 10^{-07}$	11
8	$8.94 * 10^{-08}$	130
9	$3.29 * 10^{-10}$	430
10	$1.7\ 9* 10^{-09}$	200

Table 5 Simulation Run for the Small PMS Dataset Using MOD-HTAN Activation; set goal is $1.0 * 10^{-07}$ and target month is July (i.e. 7th Month)

Trial Case	Mean Squared Error	No. of Training Epochs	Actual Value	Predicted Value	Prediction Error
1	$3.12 * 10^{-04}$	17	110.57	110.5800	$1.00 * 10^{-04}$
2	$8.50 * 10^{-01}$	12	110.57	109.3600	$1.46 * 10^{-00}$
3	$4.52 * 10^{-04}$	22	110.57	110.5701	$1.00 * 10^{-08}$
4	$8.37 * 10^{-04}$	30	110.57	110.2200	$1.22 * 10^{-01}$
5	$8.55 * 10^{-05}$	16	110.57	110.5645	$3.00 * 10^{-05}$
6	$4.79 * 10^{-04}$	80	110.57	110.5565	$1.82 * 10^{-04}$
7	$1.24 * 10^{-04}$	15	110.57	110.5648	$2.67 * 10^{-05}$
8	$1.51 * 10^{-07}$	81	110.57	110.5700	$8.34 * 10^{-11}$
9	$6.72 * 10^{-06}$	38	110.57	109.7403	$6.80 * 10^{-01}$
10	$6.73 * 10^{-04}$	25	110.57	110.5684	$2.62 * 10^{-06}$

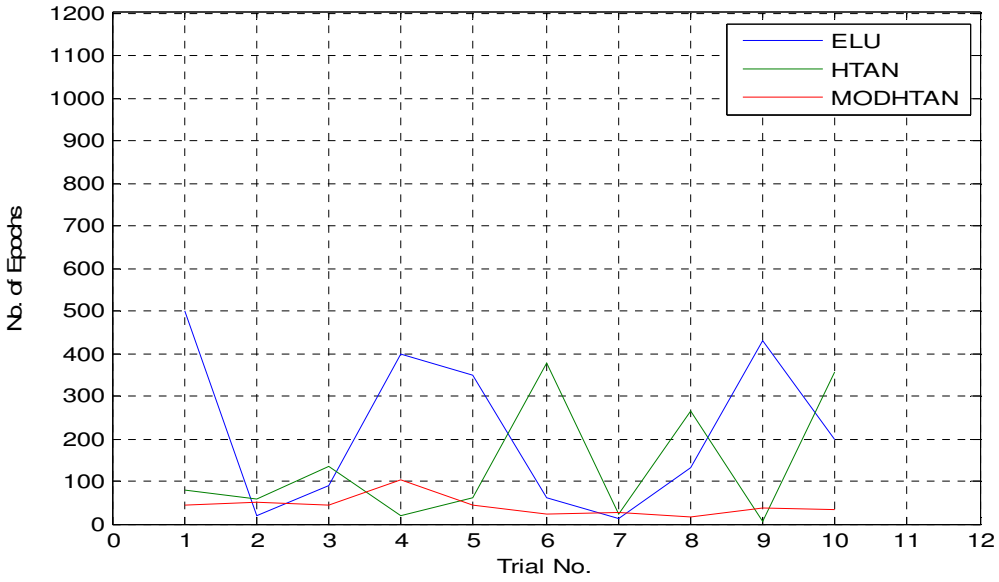

Fig.1 Line plots of training epochs for the different activations considered.

REFERENCES

1. Anireh, V.I.E. & Osegi, E.N. (2017): A Modified Activation Function with Improved Run-Times for Neural Networks. Advances in Multidisciplinary & Scientific Research Journal. Vol. 3. No.2. pp 33-44.
2. Clevert, D. A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*.
3. Glorot, X., & Bengio, Y. (2010, March). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (pp. 249-256).
4. Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., & Bengio, Y. (2016). Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*.
5. Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. *Diploma, Technische Universität München*, 91.
6. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.
7. Osegi, E. N., & Anireh, V. I. (2016). *A Real Number Formula for Approximating a Class of Irrational and Transcendental Numbers* (SURE-GP Inc, Nigeria and Rivers State University of Science and Technology, Nigeria).
8. Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., & Bengio, Y. (2016). Quantized neural networks: Training neural networks with low precision weights and activations. *arXiv preprint arXiv:1609.07061*.
9. Reed, R. D., & Marks, R. J. (1998). *Neural smithing: supervised learning in feedforward artificial neural networks*. MIT Press.
10. Scanzio, S., Cumani, S., Gemello, R., Mana, F., & Laface, P. (2010a). Parallel implementation of artificial neural network training. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on* (pp. 4902-4905). IEEE.
11. Scanzio, S., Cumani, S., Gemello, R., Mana, F., & Laface, P. (2010b). Parallel implementation of artificial neural network training for speech recognition. *Pattern Recognition Letters*, *31*(11), 1302-1309.
12. Takizawa, H., Chida, T., & Kobayashi, H. (2009). Evaluating computational performance of backpropagation learning on graphics hardware. *Electronic Notes in Theoretical Computer Science*, *225*, 379-389.
13. Tiwari, V., & Khare, N. (2015). Hardware implementation of neural network with Sigmoidal activation functions using CORDIC. *Microprocessors and Microsystems*, *39*(6), 373-381.
14. Vanhoucke, V., Senior, A., & Mao, M. Z. (2011, December). Improving the speed of neural networks on CPUs. In Proc. Deep Learning and Unsupervised Feature Learning
15. Yam, J. Y., & Chow, T. W. (2000). A weight initialization method for improving training speed in feedforward neural network. *Neurocomputing*, *30*(1), 219-232.
16. Yam, J. Y., & Chow, T. W. (2001). Feedforward networks training speed enhancement by optimal initialization of the synaptic coefficients. *IEEE Transactions on Neural Networks*, *12*(2), 430-434.