Performance Analysis of Various Activation Function on a Shallow Neural Network

Kalyani Ingole[1], Bachelor of Engineering III-year, Electronics Department,
Vivekanand Education Society's Institute of Technology, Mumbai
Narayani Patil[2], Bachelor of Engineering III-year, Information Technology Department,
Vivekanand Education society's Institute of Technology, Mumbai

Abstract: In this paper we've completed a brief study on various Activation Function and the way they affect the accuracy of a Shallow Neural network used for binary classification. This paper additionally tries to explore the analysis of Activation Functions with regards to the number of neurons required to induce the most accuracy. Through this paper, we can get a clear understanding and statistical Comparison between various activation Function. It also aims to perform analysis of the different activation functions and provide a benchmark of it.

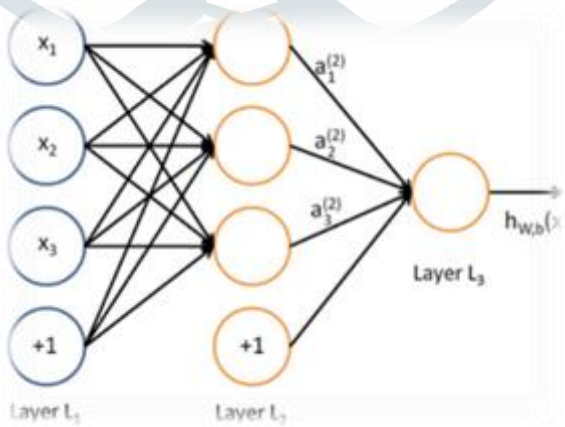
Index Terms **– Activation Functions, Machine Learning, Shallow Neural Network, Accuracy**

I. INTRODUCTION

In the early years of Machine Learning Logistic regression was one of the most used methods for binary classification process but with the introduction and progress of neural network, Logistic regression lost its charm against the Artificial neural network due to the significantly high accuracy difference among them. Artificial neural networks (ANN) is an information processing system which is inspired by the models of biological neural networks [1]. Further, due to the discovery of various Activation Function, Selection of a proper activation Function became a very important factor while training any neural network.

One of the major problems facing researchers is the selection of hidden neurons using neural networks (NN)[1]. The number of neurons to be used any neural network always changes with change in the Activation function of the neural network. The usage of a greater number of neurons in the hidden layers does not always guarantee an increase in the accuracy of a neural network. In some cases, using a large number of neurons in the hidden layer for smaller data can also lead to an overfitting problem.

In this paper, we have used the Planar dataset with supervised Learning for binary classification. Single hidden layer Neural network is deployed with the backpropagation technique to improvise the learning of the neural network [2], the sigmoid activation function is applied to the output layer neuron as the binary classification is to be done. A different activation function is applied to the hidden layer of the neural network each time and accuracy is determined and analyzed by varying the number of neurons in the neural network. This paper can be studied as a smaller piece for a deep neural network used for binary classification to improve the accuracy of the network as whole. Following is a simple representation of a Neural Network with single hidden layer.



II. Literature Survey

Artificial Neural Networks or ANN is an information processing paradigm which is inspired by biological nervous systems as the brain processes information. It is composed of a large number of highly interconnected processing elements(neurons) working in unison to solve a specific problem. These elements are usually constructed in the layers i.e. input layer, hidden layer, and output layer. So, the goal of the neural network is to feed the input layer so that it will give the correct results in the output layer [5]. When it comes to solving the problems related to Classification, matching, Clustering and Pattern recognition of data Neural Networks (ANN) are considered to be the most effective technique in the field of Artificial Intelligence [3]. In 1957 the first Neural Network model was built and it was created

by Neurobiologist Frank Rosenblatt by assuming Artificial connections between neurons could change through a Supervised Learning process that reduces the misfit between actual and expected output. The expected output comes from a training dataset [3].

In this paper our interest is to analyze performance of various Activation Functions on the Shallow Neural Networks for Binary Classification. Shallow Neural Network is a network with less number (Preferably Single Hidden Layer Network) of hidden layers [10][11]. This network can fit in any function. Our goal can be achieved by applying functions on the hidden layer of the network (ANN). Binary classification is the process of classifying a given dataset on the basis of predefined classes [9]. There are various approaches to resolve binary classification such as data mining models like Decision tree, Neural network model, SVM [9]. Also, this can also be done by logistic regression, it is a statistical Inference technique that relates binary output with the set of input variables [4]. By performing Logistic Regression and Neural networks over 180,000 samples it is concluded that logistic regression supports well on linear dataset and concluded that Neural Networks are superior to logistic regression [4].

From approximately six decades, to reach the state of absolute performance in neural network architectures, Activation Function have been used to accomplish diverse computations between the hidden layers and the output layer of a network. Activations Functions such as Sigmoid, TanH, Hard TanH, Softmax, SoftPlus, Softsign, ReLU, Leaky ReLU, DReLU, Swish, Selu, DSiLU all are summarized as per their advantages, Disadvantages and their purposes [7]. By applying various Activation Functions on the MNIST dataset which consists of 70,000 handwritten digits, 60,000 training images and 10,000 test images it is found the best results for the MNIST dataset using the Leaky Rectifier nonlinearity when the neural network contained 316 nodes in each layer and was trained with stochastic gradient descent at 75 training epochs[6].

Now, we know that from past few years a variety of Activation Functions have been developed; So, it is necessary that we should know how to choose Activation Function which will fully fill our need of solving problems. Mhaskar and Micchelli explained in their research how to choose Activation Function [8], as we have seen that there are a number of activation functions are there [7]. Author proposed a General Theorem in their research by considering the assumption that derivatives of every function up to a certain order should exist and this theorem tells us about the estimate of the number of neurons which are necessary for a Neural Network with a single hidden layer (Basically Shallow NN) [8].

III. Overview of Activation Functions

Activation functions is a concept which makes sense of something which is very complicated. The main use of this function is to convert an input signal of a node in an ANN to an output signal and This output signal is used as input to the next layer in the stack. Basically, Activation function decides whether a neuron should be activated or not by calculating the weighted sum and further adding bias to it. The motive is to introduce non-linearity into the output of a neuron. In 1958 John von Neuman introduced the idea to explain the behavior of neurons in terms of digital units: every neuron can be "On" or "Off. In case of activation, it transmits signals to other neurons forming a neural aggregate. For the study of complex data convolutional Neural Network (briefly, ConvNet), introduced by LeCun et al [3].
For ANN, output after applying Artificial Function is given by,

$y = \alpha (w1\ x1 + w2\ x2 + ... + wn\ xn + b)$ [7] where α is the activation function, x is input, w is weight and b is biases.

This section highlights study of advantages-disadvantages, behavior, Applications and evolution of Activation Functions in Artificial Neural Networks. Activation Functions we are presenting in our study are Sigmoid, Hard Sigmoid, Tanh, Hard Tanh, Softmax and Softsign and are categorized as Sigmoid, Hyperbolic Tangent, Softmax and Softsign Functions [7]. Functions we have discussed in this study are as follows:

A. *Sigmoid Function*

1) Sigmoid Activation Function

In some cases, Sigmoid Function is also referred as Logistic Function [7] and mainly applicable in Deep Learning. Mostly this function successfully works on shallow NN. But a major disadvantage of this function is it suffers from sharp damp gradients during back propagation [7]. From the graph i.e. figure.1 function curve looks like 'S' shaped. Range of Sigmoid Function is (0, 1). Since probability of anything exists between 0 to 1, therefore Sigmoid is preferred Function

Previously a single neuron was designed and is implemented using "Field Programmable Gate Array" technique which allows use of any number of neurons with any number of hidden layers and output neurons, provided its activation function type is Sigmoid [12] Computation cost of Sigmoid function is more [7].

2) Hard Sigmoid Activation Function

Hard Sigmoid Function is derived from the Sigmoid Activation Function [7]. In hard sigmoid from graph figure.2 we can see that only two cut points are there hence computation cost is less. Hard sigmoid function helps to compress the Neural Network parameters and it can greatly reduce the calculation amount and ensure the real-time capability of this network [13]. this function is piece-wise linear approximation of the sigmoid function. It is equal to 0 on the range [-∞; -2.5), then linearly increases from 0 to 1 on the range [-2.5; +2.5] and stays equal to 1 on the range (+2.5; +∞].

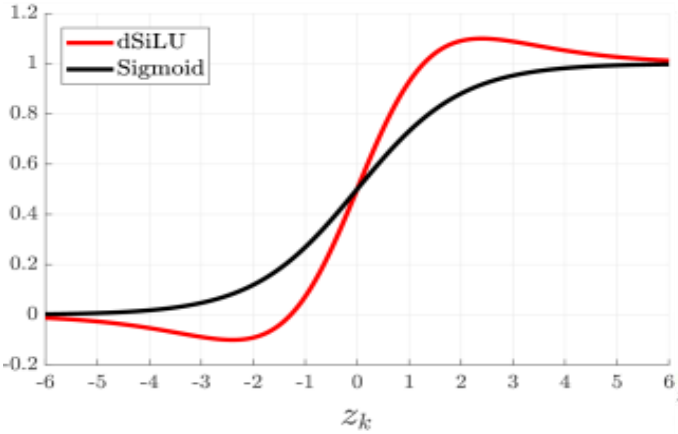


figure.1

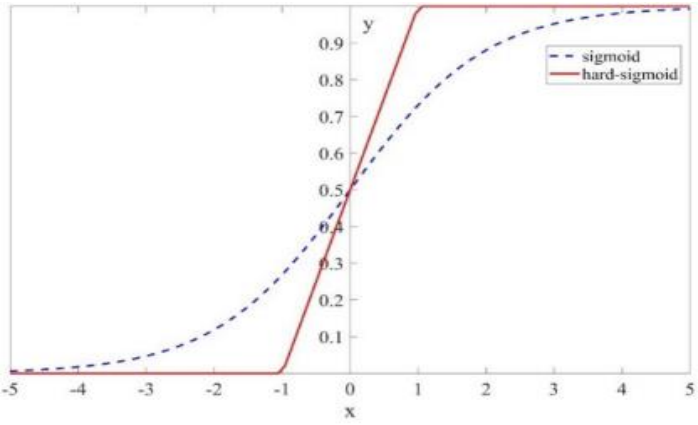


figure.2

3) dSiLU Activation Function

dSiLU is referred as derivative of Sigmoid Linear Unit. From above figure.1 it is seen that Hard Sigmoid is more volatile than Standard Sigmoid Function. dSiLU outperformed the Sigmoid Function. The dSiLU is used for gradient-descent learning updates for the neural network weight parameters [7].

B. *Hyperbolic Tangent Function*

1) Tanh Activation Function

Tanh is referred to as Hyperbolic Tangent Function. Tanh is better than logistic regression. The advantage of[11] the Tanh function is its negative input will be mapped strongly negative and zero mapped near zero in the graph. Tanh is differentiable. The important purpose of this function is used in classification problems. Tanh can also be used in CNN model [14]. Tanh is a zero centered function which is the main advantage of this function. From figure.3 Range of Tanh Function is (-1 to +1). In case of multilayer NN Tanh performs better than Sigmoid However, the disadvantage of this function is it could not solve the vanishing gradient problem.[7].

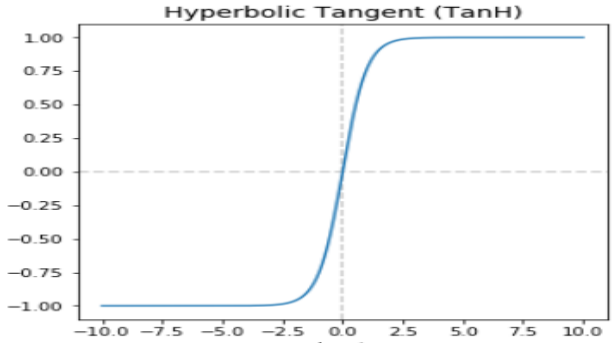


figure.3

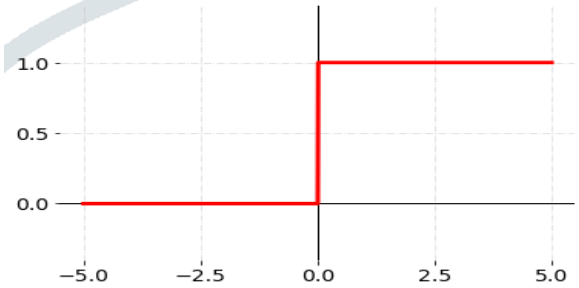


figure.4

2) Hard Tanh activation Function

Hard Tanh is also known as Hard Hyperbolic. It is variant of Tanh Function used in Deep Learning applications [7]. Hard Tanh activation Function is computationally cheaper and efficient than Tanh Function. From figure.4 we can see that range of Hard Tanh is (-1 to +1). The hardtanh function has been applied successfully in natural language processing [14], with the authors reporting that it provided both speed and accuracy improvements.

C. *Softmax Function*

Softmax Activation Function is used neural computing, it is used to compute probability from vector of real numbers [7]. Difference is Sigmoid is used in binary classification while the Softmax is used for multivariate

classification task [7]. Figure.5 shows curve for Softmax function and its range is (0 to 1). Softmax function is the monotone gradient map of the log-sum-exp function [16]. Softmax Function is given by relationship, f(x)
Also, for applications in Reinforcement application [16].

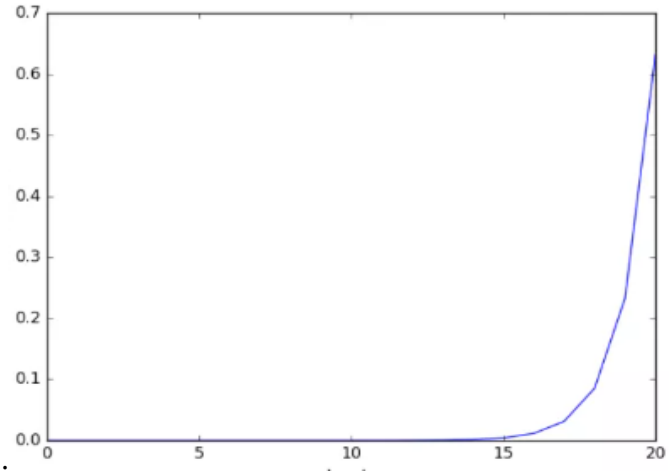


figure.5

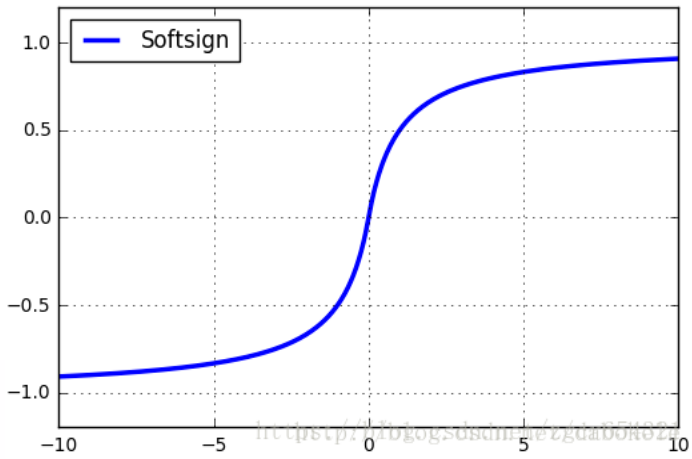


figure.6

D. *Softsign Function*

Softsign is an Earliest Activation Function used in NN. The Softsign function was introduced by Turian et al., 2009 and the Softsign is another non-linear AF used in DL application [17]. From figure.6 we can see that range of Softsign Function is (-1 to +1). The main application of Softsign function is it helps in Regression Computation problems. Difference between Tanh and Softsign Function is Tanh Converges exponentially while Softsign converges in a polynomial manner [7]. Softsign Activation Function is a quadratic polynomial [7]. From figure.3 and figure.6 we can see that graphs for TanH and Softsignt are quite similar to each other.

IV. MATHAMATICAL MODEL

Mathematical representation of Model which we have deployed for our shallow Neural Network:
In our Mathematical model 4 parameters are there i.e. $W^{[1]}$, $b^{[1]}$, $W^{[2]}$, $b^{[2]}$, Where

$W^{[1]}$ is a weight matrix of shape (Size of Hidden Layer (1), Size of Input Layer)
$b^{[1]}$ is a bias vector of shape (Size of Hidden Layer (1), 1)
$W^{[2]}$ is a weight matrix of shape (Size of Output Layer, Size of Hidden Layer (1))
$b^{[2]}$ is a bias vector of shape (Size of Output Layer (1), 1)

Model can be proposed as:

X is input for model, here for calculating probabilities we have applied forward propagation.

Step 1: Structure of NN has been defined.

$$z^{[1](i)} = W^{[1]}x^{(i)}+b^{[1]}$$

Step 2: Models parameter has been initialized.

$$a^{[1](i)} = \text{Activation Function}(z^{[1](i)})$$

Step 3: Forward propagation has been applied.

$$z^{[2](i)} = W^{[2]}a^{[1](i)} + b^{[2]}$$

Step 4: Sigmoid output of Activation Function has been returned.

$$a^{[2](i)} = \text{Sigmoid}(z^{[2](i)})$$

Step 5: $y^{\wedge(i)} = a^{[2](i)} = \sigma(z^{[2](i)})$

Step 6: Predictions can be donned as:

$$y^{(i)}_{prediction} = \begin{cases} 1 & \text{if } a^{[2](i)} > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

In Step 2 we have applied various Activation Functions to analyze Accuracies.

V. RESULTS AND DISCUSSION

5.1 Individual Study of Functions

1]Sigmoid Activation Function

Applying Sigmoid activation function to above discussed Neural networks' hidden layer, it gives 50% accuracy with 4 neurons used in the hidden layer of the neural network but with an increase in the number of neurons in the hidden layer accuracy shoots up to 90% with 45 neurons used in the hidden layer. Following is the detailed graph tracing the accuracy with respect to change in number of neurons in the hidden layer.

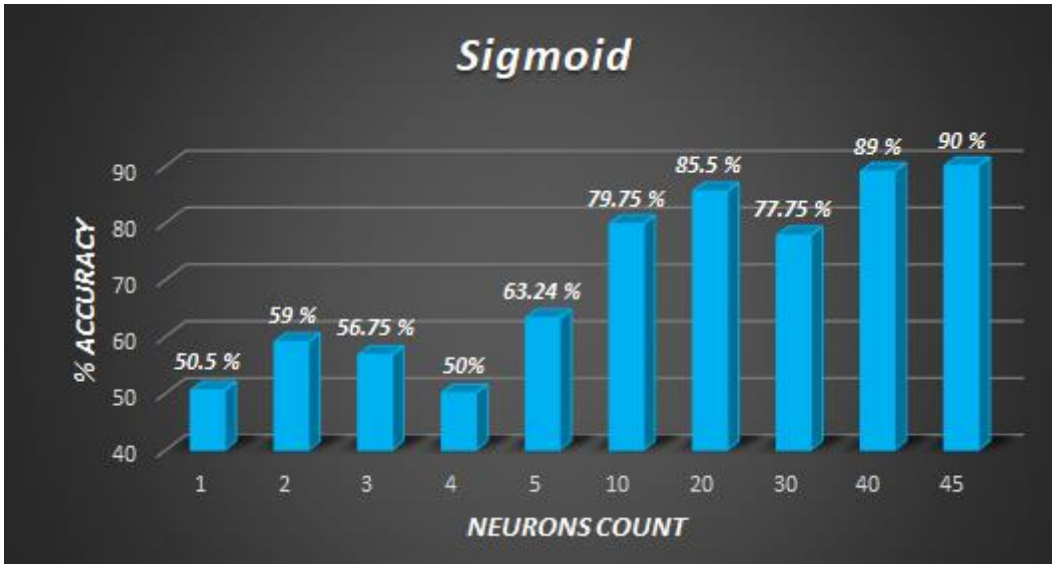


figure.7

2]Hard Sigmoid Activation Function

Hard sigmoid is a non-smooth function and is used as an activation function to dissolve the cons of sigmoid activation function as the sigmoid function is comparatively slower to compute due to the exp() function. Applying Hard Sigmoid activation function to above discussed Neural networks' hidden layer, it gives 63.25% accuracy with 4 neurons used in the hidden layer of the neural network but with an increase in the number of neurons in the hidden layer accuracy shoots up to 87.25% with 45 neurons used in the hidden layer. Following is the detailed graph tracing the accuracy concerning the change in number of neurons in the hidden layer.

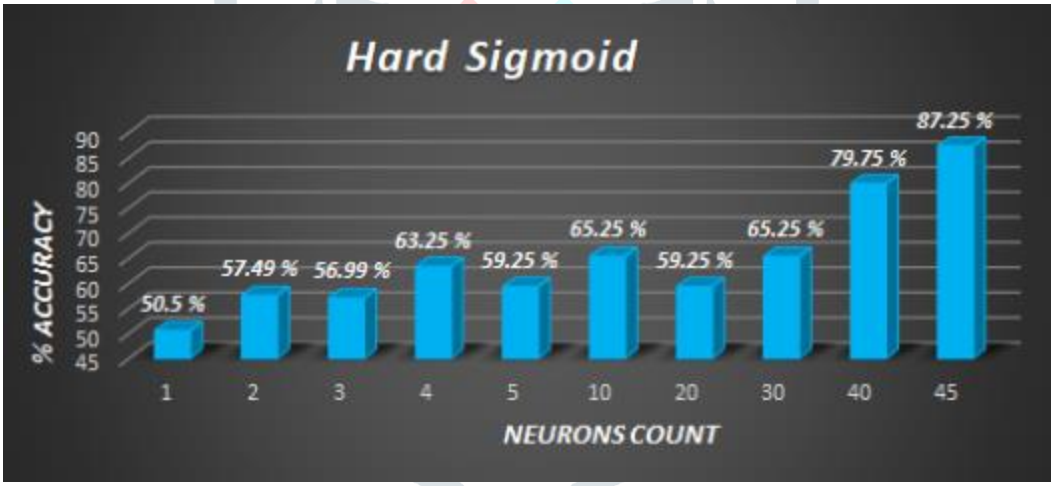


figure.8

3] dSiLU Activation Function

dSiLU is a steeper and overshooting version of the Sigmoid function. Applying dSiLU activation function to above discussed Neural networks' hidden layer, it gives 66.75% accuracy with 10 neurons used in the hidden layer of the neural network but with an increase in the number of neurons in the hidden layer accuracy shoots up to 86.25% with 50 neurons used in the hidden layer. Following is the detailed graph tracing the accuracy concerning the change in number of neurons in the hidden layer.

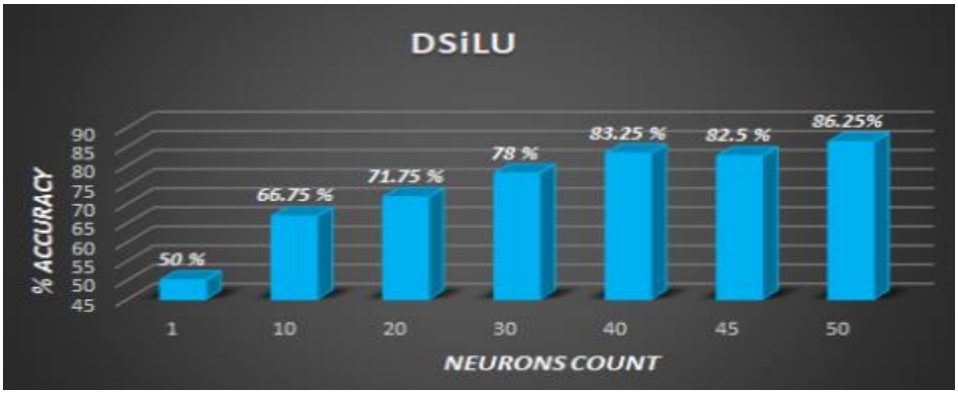


figure.9

4] TanH Activation Function

TanH is a non-linear function in nature, so layers can be stacked. The gradient is stronger for TanH than the sigmoid function. Applying TanH activation function to the above discussed Neural networks' hidden layer, it gives maximum accuracy of 91.25% with 5 neurons used in the hidden layer of the neural network but even with the increase in the number of neurons the accuracy maintains its stability in the same range of 90%.

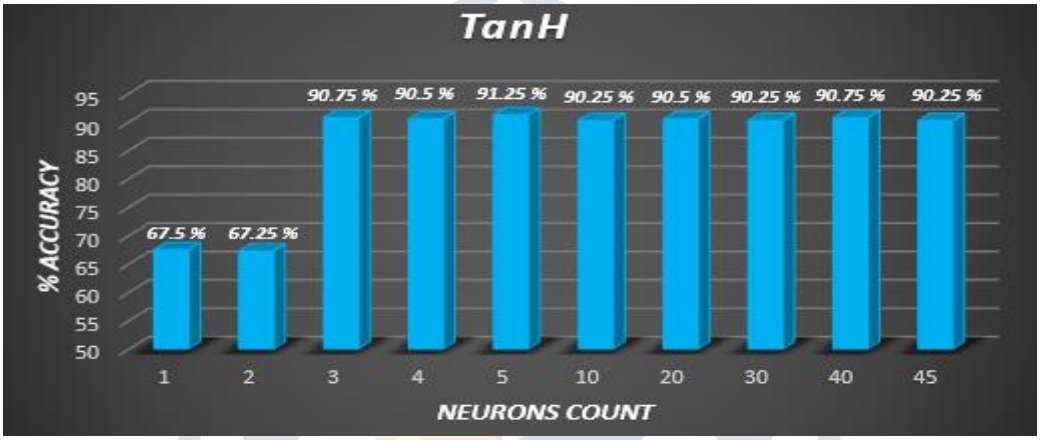


figure.10

5] Hard TanH Activation Function

Hard TanH is computationally cheaper than TanH. Applying Hard Sigmoid activation function to above discussed Neural networks' hidden layer, it gives 67.5% accuracy with 3 neurons used in the hidden layer of the neural network but with an increase in the number of neurons in the hidden layer accuracy shoots up to 92.25% with 10 neurons used in the hidden layer. Following is the detailed graph tracing the accuracy concerning the change in number of neurons in the hidden layer.

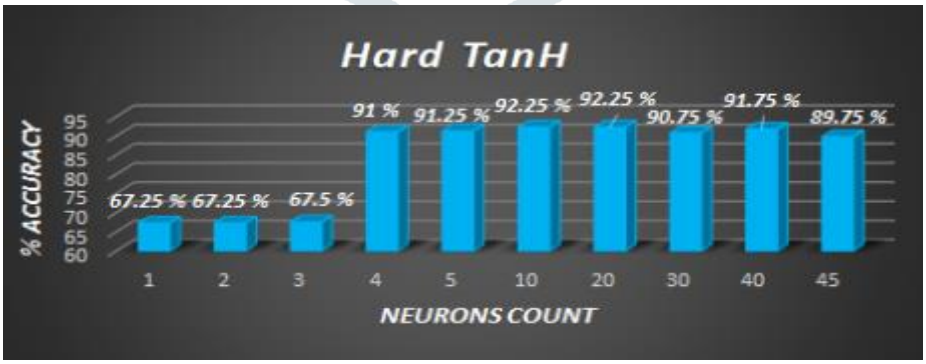


figure.11

6] SoftMax Activation Function

SoftMax Activation Function mainly developed to map non-normalized data. This Function converts numeric output of NN in probabilities then it normalizes each number by sum of exponents so entire output vectors is equal to one. Applying SoftMax activation function to above discussed Neural networks' hidden layer, it gives 54% accuracy with 4 neurons used in the hidden layer of the neural network but by changing count of neurons in the hidden layer, Function gives steady accuracy lies in range of 50-55%. Softsign Function gives maximum accuracy up to 55.5% with 5 neurons

used in the hidden layer. Following is the detailed graph tracing the accuracy with respect to change in number of neurons in the hidden layer.

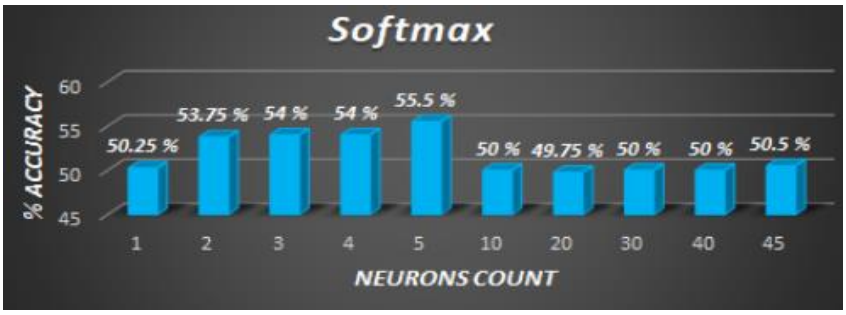


figure.12

7] Softsign Activation Function

Softsign is another non-linear Activation Function which can be considered as alternative for TanH Function since it too does not saturate as easily as hard clipped functions. From figure.10 we can see that highest accuracy for TanH Function is 91.25% which is greater than highest accuracy of Softsign, but this function is proposed to as it converges in a polynomial manner. Applying Softsign activation function to above discussed Neural networks' hidden layer, it gives 87.25% accuracy with 4 neurons used in the hidden layer of the neural network but with an increase in the number of neurons in the hidden layer accuracy shoots up to 90.75% with 50 neurons used in the hidden layer. Following is the detailed graph tracing the accuracy with respect to change in number of neurons in the hidden layer.

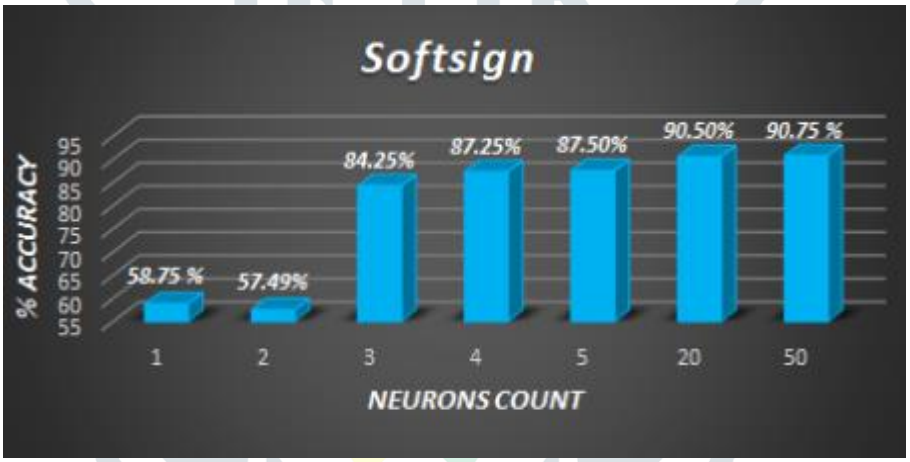


figure.13

5.2 Comparative Study of Functions

Every Activation function discussed in 5.1 can be easily and systematically understood in terms of their maximum accuracy and the number of neurons needed to achieve it from the following figure.14. For the dataset we used, the SoftMax function worked the worst whereas the Hard TanH activation function worked the best in terms of accuracy.

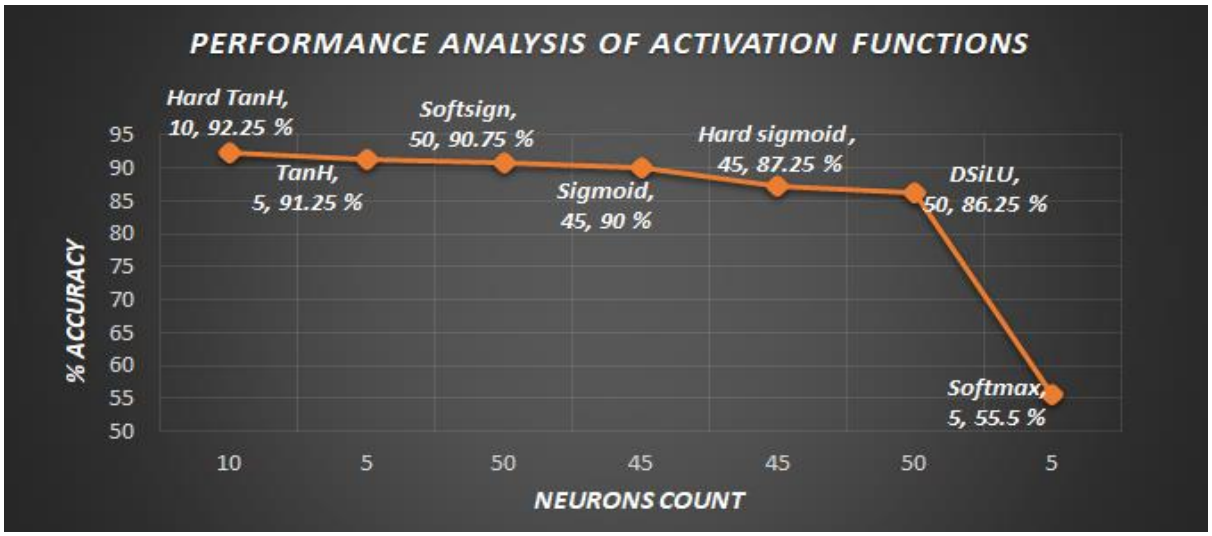


figure.14

VI. Conclusion

In this paper, we have performed a comparative study on how various activation functions work well in a sense of determining the accuracy of a shallow neural network, where different activation functions are tested only on the hidden layer. All Activation functions are studied in detail in Correspondence to changing the number of neurons in the hidden layer and their effect on the accuracy of the neural network. We have obtained estimates on the number of neurons necessary for each activation function to achieve minimum losses for a single layer neural network used for binary classification.

REFERENCES

[1] K. Gnana Sheela, S. N. Deepa. (2013). "Review on Methods to Fix Number of Hidden Neurons in Neural Network" https://doi.org/10.1155/2013/425740

[2] Buscema, Massimo. (1998). Back Propagation Neural Networks. Substance use & misuse. 33. 233-70. 10.3109/10826089809115863.

[3] Dell'Aversana, Paolo. (2019). ARTIFICIAL NEURAL NETWORKS AND DEEP LEARNING.ASIMPLE OVERVIEW.

[4] Adeodato, Paulo & Vasconcelos, Germano & Arnaud, Adrian & Santos, Roberto Angelo & Cunha, Rodrigo & Monteiro, Domingos. (2004). Neural Networks vs Logistic Regression: a Comparative Study on a Large Data Set.355-358.

[5] 569/Essays_Spring2018/Khan, "Emergent Behavior in Neural Networks".

[6] Manavazhahan, Meenakshi, "A Study of Activation Functions for Neural Networks" (2017). Computer Science and Computer Engineering Undergraduate Honors Theses. 44.

[7] arXiv:1811.03378v1 [cs.LG] 8 Nov 2018

[8] H. N. Mhaskar, c. A. Micchelli ,"How to Choose an Activation Function"

[9] Kumari, Roshan & Srivastava, Saurabh. (2017). Machine Learning: A Review on Binary Classification. International Journal of Computer Applications. 160. 11-15. 10.5120/ijca2017913083.

[10] arXiv:1608.03287v1 [cs.LG] 10 Aug 2016

[11] arXiv:1905.10161v2 [cs.LG] 24 Jun 2019

[12] Jamel, Thamer & Mohammed, Ban. (2012). IMPLEMENTATION OF A SIGMOID ACTIVATION FUNCTION FOR NEURAL NETWORK USING FPGA.

[13] W. Zhang, Z. Mi, Y. Zheng, Q. Gao and W. Li, "Road Marking Segmentation Based on Siamese Attention Module and Maximum Stable External Region," in *IEEE Access*, vol. 7, pp. 143710-143720, 2019, doi:10.1109/ACCESS.2019.2944993.

[14] Yingying Wang, Yibin Li, Yong Song and Xuewen Rong, "The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition", China, 2020, 20

[15] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural Language Processing (Almost) from Scratch," Journal of Machine Learning Research, vol. 12, pp. 2493–2537, 2011.

[16] Gao, Bolin & Pavel, Lacra. (2017). On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning.

[17] J. Turian, J. Bergstra, and Y. Bengio, "Quadratic features and deep architectures for chunking," in Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, vol. Companion Volume: 2009, pp. 245–248.