Volume 9, Issue 4, April – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR934

Activation Functions for Neural Networks: Application and Performance-based Comparison

[1]Ajay Kumar
M. Tech Student
Defence Institute of Advanced Technology, Pune

[2]Dr. Nilesh Ware
Assistant Professor
Defence Institute of Advanced Technology, Pune

Abstract:- Past decade has seen explosive growth of Deep Learning (DL) algorithms based on Artificial Neural Networks (ANNs) and its applications in vast emerging domains to solve real world complex problems. The DL architecture uses Activation Functions (AFs), to perform the task of finding relationship between the input feature and the output. Essential building blocks of any ANN are AFs which bring the required non-linearity of the output in the Output layer of network. Layers of ANNs are combinations of linear and nonlinear AFs. Most extensively used AFs are Sigmoid, Hyperbolic Tangent (Tanh), Rectified Linear Unit (ReLU) etc to name a few. Choosing an AF for a particular AF depends on various factors such as Nature of Application, Design of ANN, Optimizers used in the network, Complexity of Data etc. This paper presents a survey on most widely used AFs along with the important consideration while selecting an AF on a specific problem domain. A broad guideline on selecting an AF based on the literature survey has been presented to help researchers in employing suitable AF in their problem domain.

Keywords:- *Artificial Neural Network, Activation Functions, RNN.*

I. INTRODUCTION

AFs are processing units based on mathematical equations that determine the output of ANN model. Each neuron in ANN receives input data, applies a linear transformation (weighted summation) and then pass the result through an AF typically a Sigmoid or ReLU, in order to bring non-linearity to the input data. This process allows ANN to capture complex, non-linear relationships within data which otherwise is not possible by conventional Machine Learning algorithms. Performance of ANN is dependent on its efficiency of convergence so as to stabilise the weights assigned to various input which are provided to a Neuron or a set of Neurons in each layer. ANN is said to have converged when no further significant change to the assigned weights is possible in subsequent iterations. Depending on the selection of AFs, a network may converge faster or may not converge at all. AF is chosen so as to limit the output in the range to any value between -1 to 1 , 0 to 1 , or $-\infty$ to $+\infty$ depending upon the AF used in various neuron in different layers of ANN.

II. ACTIVATION FUNCTIONS

Primarily there are 04 types of AFs prevalent in ANN namely Sigmoid, ReLU, Exponential Unit and Adaptive Unit based AFs. In addition to these primary functions, there are a number of use-case based variations of primary AFs which are well suited for a specific application area such as Leaky-ReLU is suitable for Convolution Neural Network (CNN). **Chigozie Enyinna Nwankpa** et al has presented a detailed summary of various AFs used in ANN. Author has identified that there are 04 flavors of ReLU AFs which are prevalently used in neural networks the rectified linear units (ReLU) namely Leaky-ReLU, Parametric-ReLU, Randomized-ReLU and S-shaped-ReLU. Furthermore, the Sigmoid has two variants namely Hyperbolic tangent (TanH) and Exponential Linear Squashing Activation Function (EliSH). The Softmax, Maxout , Softplus, Softsign, and Swish functions has no variants [1]. Sigmoid is generally used in Logistic Regression whereas TanH is predominantly used in LSTM cell in NLP tasks using Recurrent Neural Networks (RNN).

The **Exponential Linear Unit** (ELU) has negative values which works as Batch Normalisation thus speeding up the convergence but with lower computational complexity.

III. NEED FOR ACTIVATION FUNCTION

ANNs enables an algorithm in making faster decisions without human intervention as they are able to infer complex non-linear relationship among the features of the dataset. The primary role of the AF is to transform the calculated summation of weights input from the node and bias into an output value to be fed to the next hidden layer or as output in the output layer [Figure-01]. The output produced by the AF of output layer is compared to the desired value by means of Loss function and accordingly a Gradient is calculated using some Optimization Algorithm (usually Gradient Descent) in order to achieve the local or global minima for the ANN using backpropagation. The resultant weight vector that contains the hidden characteristics in the data. These AFs are often referred to as a Transfer Functions. A typical ANN is biologically inspired computer programmes, inspired by the working of human brain. These collection of neurons are called networks because they are stacked together in form of layers with each set of neurons performing specific function and infer knowledge by detecting the relationships and patterns in data using past experiences known as training examples. In the absence of AFs every neuron will behave as an Identity function and will only perform the summation of the weighted input features using the weights and biases because irrespective of number of layers in ANN all the neurons would simply output the summation of input features without transforming them. The resultant linear function will not be able to capture the non-linear pattern in the input data, even though, the ANN will become simpler. However, ANN would work well for linear regression problems where the predicted output is same as the weighted sum of the input features and bias.

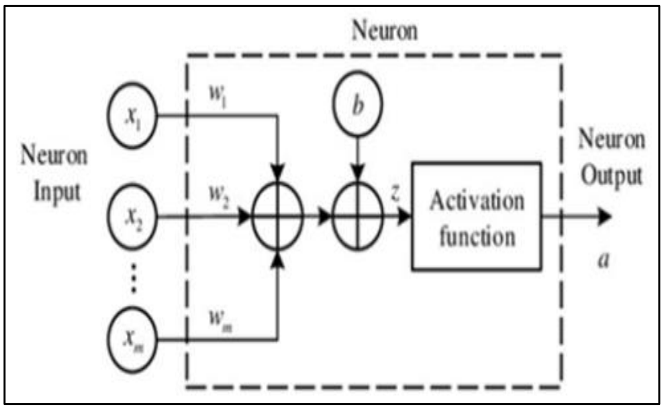

Fig 1 Building Blocks of Neural Network

IV. LITERATURE SURVEY

- **A Wibowo, W Wiryawan, and I Nuqoyati** experimented on Cancer classification using using microRNA feature. In his experiment Gradient Descent, Momentum, RMSProp, AdaGrad, AdaDelta, and Adam optimizers were used. The result showed that ReLU AF produced 98.536% and 98.54762% accuracy with Adam and RMSProp optimizer resp [2].
- **Bekir Karlik and A. Vehbi Olgac** attempted to analyze the performance of Multi Layered Perceptron architectures using various AFs such as sigmoid, Uni-polar sigmoid, Tanh, Conic Section, and Radial Bases Function (RBF) with varying number of Neurons in hidden and output layers. Results shows that when Tanh AF was used in both hidden & Output layers, best accuracy of around 95 and 99% was observed for 100 and 500 iterations resp [3].
- **Hari Krishna Vydana, Anil Kumar Vuppala** studied the influence of various AFs on speech recognition system on TIMIT and WSJ datasets. The result shows that on smaller datasets (TIMIT) ReLU worked better producing the minimum phone rete error of 18.9 whereas for larger dataset (WSJ), Exponential Linear Unit (ELU) produced better result of reduced phone rate error of 19.5. It shows whenever we have a larger dataset for speech recognition, we should first employ ELU. [4]
- **Giovanni Alcantara** conducted an empirical analysis on the effectiveness of using AFs on the MNIST classification of handwritten digits (LECUN). Experiments suggests that ReLU, Leaky ReLU, ELU and SELU AFs all yield great results in terms of validation error, however, ELU performed better than all other models [5].
- **Farzad et al** employed Long Short-Term Memory based Recurrent ANN (RNN) for sentiment analysis task on IMDB and Movie Review and observed that **Elliott** AF with one hidden layer for classification of records in the IMDB, Movie Review, and MNIST data sets. Elliott AF and its modifications demonstrated the least average error and better results than the sigmoid AF which is more popular in LSTM networks [6].
- **Dubey and Jain** et al have used ReLU and Leaky-ReLU AFs for deep layers and softmax AF for Output layer on MNIST dataset classification. Result obtained shows that CNN with ReLU showed better results than Leaky ReLU in terms of model accuracy and model loss. Same findings have been confirmed by Banerjee et al in his experiment on MNIST dataset [7].
- **Castaneda et al** experiment on Object detection, Face detection, Text and Sound dataset using ReLU, SELU and Maxout shows that ReLU is best suited for Object, Face and Text detection, whereas, SELU and Maxout are better for sound / speech detections [8].
- **Shiv Ram Dubey et al** have used CIFAR10 and CIFAR100 datasets for the image classification experiment over different CNN models. It is observed that the Softplus, ELU and CELU are better suited with MobileNet. The ReLU, Mish and PDELU exhibit good performance with VGG16, GoogleNet and DenseNet. The ReLU, LReLU, CELU, ELU, GELU, ABReLU, and PDELU AFs are better for the networks having residual connections, such as ResNet50, SENet18 and DenseNet121 [9].
- **Tomasz Szandała** et al experimented data-set CIFAR-10 with just two CNN convolution layers to compare efficiency of ReLU, Sigmoid, TanH, Leaky-ReLU, SWISH, Softsign and Softplus. The training/ test data were split in 5:1 ratio [Fig-02] presents the accuracy of various AFs [10]. The performance of ReLU and Leaky ReLU were the most satisfying and both of them produced above 70 % accuracy in classification in spite of the dying ReLU condition. Rest all other AFs resulted in lower accuracy of less than 70 % [Table-01].

Table 1 Relative Accuracy Ratio of AFs

	sigmoid	tanh	relu	leaky_relu	swish	softsign	softplus
Final accuracy	0.6166	0.6757	0.7179	0.7295	0.6989	0.6901	0.6598

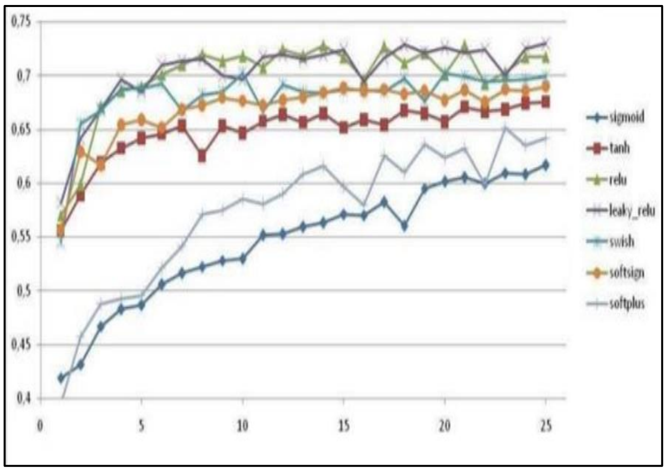

Fig 2 Relative Accuracy Graph of AFs

V. IMPORTANT CONSIDERATIONS FOR SELECTION OF AF

The selection of an AF is a critical decision when designing and training ANNs. Different AFs can have a significant impact on the performance and convergence of a ANN. Here are some critical factors that affect the choice of AF in an ANN:

➢ *Differentiability:*

Two important properties of AFs are that they should be differentiable and that their gradient should be expressible using the function itself. The first essential properties make them suitable for the backpropagation which is the essence of ANNs. Many optimization techniques, such as gradient descent, rely on the derivative of the AFs. Hence, it's essential that the AF is differentiable or has well-defined gradients. This allows for the efficient training of the network. The second desirable property reduces the computation time of ANNs because at times the ANN is trained on millions of complex data points. Functions of Sigmoid and TanH AFs and their derivative is as shown below along with their graph plot [Fig-03] :

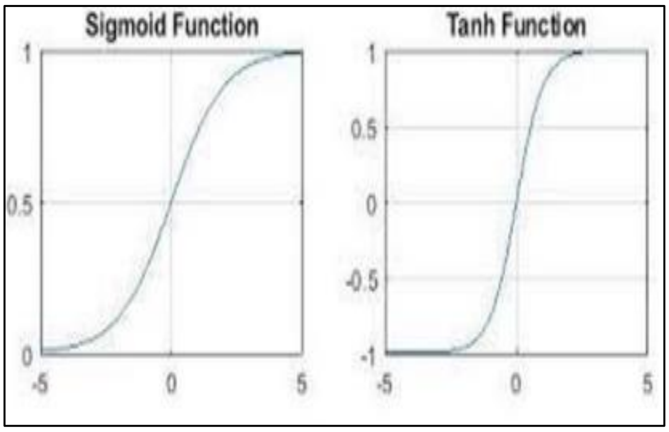

Fig 3 Sigmoid & TanH Graph

➢ *Vanishing and Exploding Gradients:*

Some AFs such as Sigmoid, Tanh are prone to vanishing gradient problem, where gradients become very small during backpropagation leading to delayed convergence. Adjoining figure shows the plot of a Sigmoid AF and its derivative. The maximum value of the Sigmoid derivative is 0.3 (approx) [Fig-04]. In the backpropagation when chain rule is applied as a process of updating the weights associated with neurons in the hidden layer, the derivatives tend to get smaller exponentially and at a deeper layer it will become insignificant to cause any significant change in the weights. This means, in deeper architectures, no learning happens for the deeper neurons or it is remarkably slower rate as compared to learning in shallower layers. This can make training slow and less effective. If we choose an optimizer that is sensitive to vanishing gradients (e.g., plain SGD), you might want to avoid using AFs that make this problem worse. Since, derivative of ReLU for positive input is always '1', it addresses the problem of vanishing gradient in ANN. However, a '0' gradient for negative input it leads to dead activation. A modified version Leaky ReLU addresses the issue of dead activation by replacing '0' with alpha times x (alpha = 0.01) such that derivative of LReLU is slightly greater than '0' [Fig-05]. Similarly, Parametric ReLU (PReLU), ELU are often preferred because they are less prone to vanishing gradients [11]

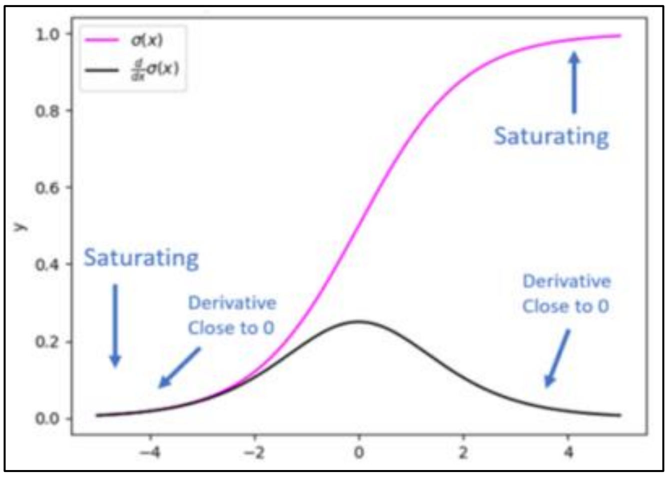

Fig 4 Sigmoid and its Derivative Curve

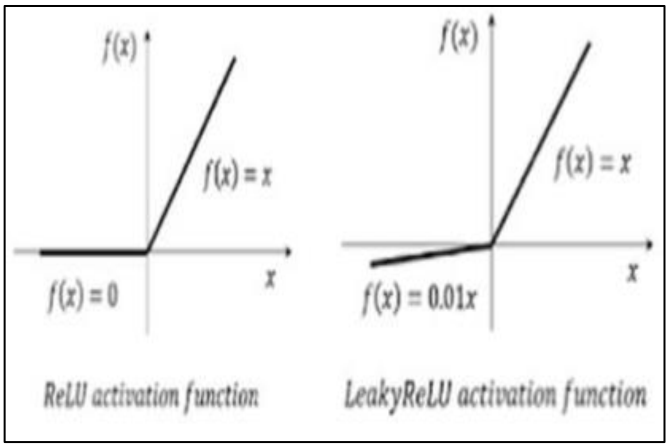

Fig 5 ReLU and Leaky ReLU Graph

➢ *Sparsity of Neurons:*

As the number of layers and hence the number of neurons increases in an ANN, the networks start overfitting. The very first step to avoid overfitting is to reduce the complexity of the ANN by reducing the number of hidden layers thereby number of neurons in network. Some AFs introduce sparsity in the network by allowing some neurons to remain inactive. This situation is called Dead-Activation. It is observed in ReLU AF which is the result of negative input to the neuron which results in '0' output for all negative data. In some scenario, the dead activation has been found useful such as Feature identification using CNN. It is desirable that while trying to detecting a feature are computationally efficient compared to more more complex ones like Sigmoid or TanH. Owing to lesser computational complexity in ReLU family compared to Sigmoid, TanH AFs, ANN employing ReLU tends to converge faster than later AFs. An experiment conducted on CIFAR-10 dataset using ReLU and TanH AFs shows that ReLU converges six times faster than TanH [Fig-06] [12].

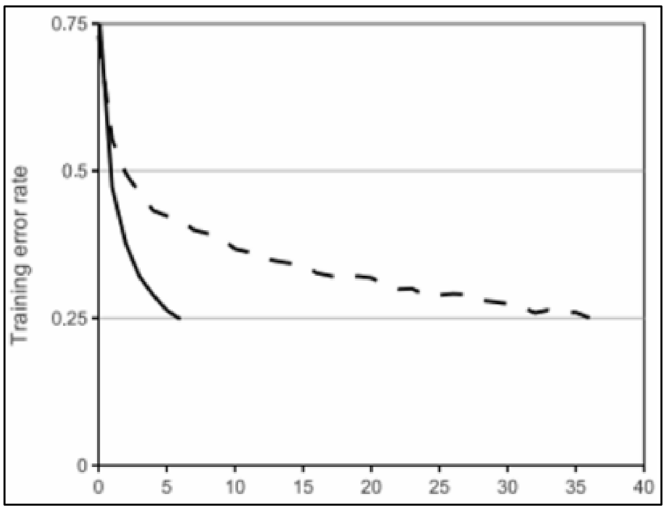

Fig 6 Sigmoid and its Derivate Curve

➤ *Architectural Considerations:*

The choice of AF can also be influenced by the overall architecture of the ANN. Different AFs have different properties and are more suitable for certain types of network designs and tasks. It's important to experiment with different AFs and monitor their impact on the network's training and performance to select the one that works best for specific task. Some critical factors to consider before selecting an AF based on the ANN design considerations are :

- *Output Layer:*

The choice of AF for the output layer depends on the nature of the problem. For binary classification, the sigmoid function is often used, while Softmax is common for multi-class classification. Logistic Regression, being a two-class problem, uses Sigmoid AF.

- *Hidden Layer:*

AFs that are used in the hidden layer should be ***Non-linear,*** to let the ANN learn non-linear relationships, ***Unbounded,*** to enable faster learning and avoid saturating early and ***differentiable,*** to enable back-propagation and learning. Typically, AFs like ReLU or its variants (e.g., Leaky ReLU) are used in hidden layers of an ANN.

- *Range of Output:*

The range of values that an AF can produce also matters. Some AFs squash their input into a specific range, like the sigmoid function which maps input to values between 0 and 1. Others, like the ReLU, produce unbounded outputs. The choice of AF depends on the problem and the desired output range. In a regression problem, we use the linear (identity) AF with one node. In a binary classifier, we use the sigmoid AF with one node.

VI. APPLICATION BASED SELECTION

The choice of AF can also depend on the specific problem which we are trying to solve. For instance, Softmax is commonly used in the output layer in classification tasks, while other functions such as Sigmoid, Tanh might be more suitable for sentiment analysis. For Logistic Regression, default choice is Sigmoid AF whereas, ReLU and its variations are widely used with CNN due to their effectiveness in image processing tasks. Selecting an AF for a network in output layer is made based on the nature of the problem and the detected variables. As a rule of thumb, for hidden layer, we must employ ReLU as a default AF. Only in cases where the performance is not optimum should we move to other AFs. We should avoid using Sigmoid, Tanh functions in hidden layers as they have inherent problem of Vanishing Gradient. Swish function is used in ANNs having a depth greater than 40 layers. ReLUs, among all the other viable contenders have the cheapest computational budget, as well as simple to implement**.** If ReLU doesn't give promising results, we may use either a Leaky ReLU or ELU. If compute resources are at premium and network is deep, Leaky-ReLU is preferred else ELU works better. If lot of computational budget and time is available, PReLU may be used. With Parametric ReLU, a whole bunch of parameters to be learned are added to optimisation problem so it needs lots of training data. Randomized-ReLU can be useful if your network suffers with overfitting. In nutshell, there is no thumb rule to apply a particular AF in a particular situation or a problem domain. There are advantages and disadvantages of every AF and it completely depends on the given situation which enables the choice of an AF.

➤ *However, Researchers may follow following Broad Guidelines while Trying out Various AFs: -*

- Best AF to start working on is ReLU as it obviates Vanishing Gradient and makes computations faster due to its simple nature of function and its derivative.
- If ReLU is resulting in overfitting of network due to complexity of ANN, we can use the leaky ReLU function to address dead activation, if it is not desirable.
- In cases of slow convergence due to Vanishing Gradient problem i.e. gradient reaching the value zero, try avoiding Sigmoid and Tanh functions.
- For NLP tasks using RNN a combination of sigmoid/ Tanh functions gives better results.
- ReLU should only be used in the hidden layers and not in the outer layer of ANN. [13]

VII. CONCLUSION

This paper has presented a very brief introduction to AFs used in an ANN with an aim to provide a broad guideline to researcher on selecting a particular AF in a given situation. Choosing an AF depends on variety of conditions such as the Complexity of ANN, Vanishing Gradients, Dead activation, Complexity of the function itself etc. AN effort has been made to highlight such conditions in order to consider a particular AF for a given problem. In summary, while variety of factors can affect the choice of an AF, there is no one-size-fits-all rule. It's essential to consider the characteristics of AF, as well as the specific requirements of your ANN and problem statement, when making these choices. Experimentation and fine-tuning are often necessary to find the best combination that yields optimal training and performance results for your dataset. Most of the AFs have not been discussed in much detail due to restriction on content, however, as a future work, a comprehensive analysis of all the AFs can be done while considering their advantages and disadvantages in a particular situation and a much more detailed comparison of various AFs would be carried out at an implementation level presenting a more enhanced guideline on AFs.

REFERENCES

[1]. N. C. Envinna, "Activation Functions: Comparision of trends in practice and reseachfor deep learning," CS.LG, 2018.
[2]. W. W. A Wibowo, "Optimization of ANN for Cancer microRNA biomarkers classification," Journal of Physics, 2019.
[3]. B. Karlik, "performance aalysis of various Activation functions in generalised MLP architectures of ANN," International Journal of Artificial Intelligence And Expert Systems (IJAE), vol. (1), no. (4), 2011.
[4]. H. K. Vydana, "investigation study of various Activation functions for speech recognition".
[5]. A. Giovanni, "Emperical analysis of non-linear Activation functionsfor deep ANN in classification tasks," CS.LG, 2017.
[6]. A. Farzad, "a comparative performance analysis of different activation functions in LSTM network for classification," Neural Comput & Applic 31, 2507–2521 , 2019.
[7]. V. J. Dubey AK, "Comparative study of convolution ANN's relu and leaky-relu AFs," Lecture Notes in Electrical Engineering, vol 553. Springer, Singapore, 2019.
[8]. G. Castaneda, "Evaluation of maxout activations in deep learning across several big data domains," Journal of Big Data, 2019.
[9]. S. R. Dubey, "Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark," Neurocomputing, vol. 503, 2022.
[10]. T. Szandała, "Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks," SCI, vol. 903, 2020.
[11]. S. V. Chekuri, "Online video lectures," Applied Roots, 2018.
[12]. A. Krizhevsky, "ImageNet Classification with Deep Convolutional".
[13]. S. Sharma, "ACTIVATION FUNCTIONS IN NEURAL NETWORKS," International Journal of Engineering Applied Sciences and Technology, vol. 4, no. 12, 2020.
[14]. R. Jiang, "Deep Neural Networks for Channel Estimation in Underwater Acoustic OFDM Systems," IEEE, 2019.
[15]. S. Sharma, "Activation Functions in Neural Networks," Toward data science, 2017.