Research Paper

Open Access

APTx: Better Activation Function than MISH, SWISH, and ReLU's Variants used in Deep Learning

Ravin Kumar[1*]

[1]Department of Computer Science, Meerut Institute of Engineering and Technology, Meerut-250005, Uttar Pradesh, India.
E-mail: ravin.kumar.cs.2013@miet.ac.in

Abstract

Activation Functions introduce non-linearity in the deep neural networks. This nonlinearity helps the neural networks learn faster and efficiently from the dataset. In deep learning, many activation functions are developed and used based on the type of problem statement. ReLU's variants, SWISH, and MISH are goto activation functions. MISH function is considered having similar or even better performance than SWISH, and much better than ReLU. In this paper, we propose an activation function named APTx which behaves similar to MISH, but requires lesser mathematical operations to compute. The lesser computational requirements of APTx does speed up the model training, and thus also reduces the hardware requirement for the deep learning model.

Keywords: *Activation functions, ReLU, Leaky ReLU, ELU, SWISH, MISH, Neural networks*

1. Introduction

The ability of deep learning models to learn features directly from the data have made it a default approach to solve many complex problems. A simple artificial neuron is linear in nature, also expressed in Equation (1).

$$y = \Sigma w_i x_i + b \qquad \qquad ...(1)$$

where,

y is output from the neuron;

x_i is the input to the neuron;

w_i is the associated weights; and

b is the associated bias.

When the output of this neuron is passed to an activation function the nonlinearity gets introduced in the network. When considering an activation function one important thing is that the derivative of an activation function should not be the same in its domain. Generally, activation function ***f*** is applied to the output of the neurons in the hidden layers to make the neural network learn complex features as expressed in Equation (2).

* Corresponding author: Department of Computer Science, Meerut Institute of Engineering and Technology, Meerut-250005, Uttar Pradesh, India. E-mail: ravin.kumar.cs.2013@miet.ac.in

$Output = f(y)$ ...(2)

The SWISH activation function is considered better than the ReLU function and its variants. But, recently developed activation function MISH is considered equivalent or even better than SWISH activation function in some cases.

In this paper, we propose an activation function APTx which behaves similar to the MISH activation function but requires lesser mathematical operations. It means lesser computation is required in APTx to calculate output in the forward propagation, as a result significantly reducing the hardware requirements for training and inference phases. The derivative of APTx also has lesser operations than MISH, hence making neural networks train faster compared to MISH activation function.

2. Related Works

Vinod Nair and Geoffrey (2010) studied the effect of Rectified Linear Units (ReLU) on Restricted Boltzmann Machines. Agarap (2022) made use of ReLU with convolutional neural networks on the MNIST dataset which outperformed the CNN with softmax on classification task. Glorot *et al.* (2011) and Sun *et al.* (2014) discussed the sparsity of ReLU as a reason for its better performance. Szandala (2021) performed a comparative analysis showing tan*h* and sigmoid function both having vanishing gradient problems overcome by ReLU, and showing the dying-ReLU problem for negative values. Mass *et al.* (2013) presented an improved version of ReLU called Leaky-ReLU where instead of having zero value for negative input the function will have some negative number output. Clevert *et al.* (2015) proposed an ELU function that was faster and better than both ReLU and Leaky-ReLUs. Ramachandran *et al.* (2017) presented SWISH activation function having superior performance than ReLU and its variants. Misra (2019) proposed an activation function MISH having similar, and in some cases even better performance than SWISH activation function.

3. Proposed APTx Activation Function

We are proposing an activation function named as "Alpha Plus Tan*h* Times" or APTx in short. Our APTx function is presented as ϕ in Equation (3), and its derivative is shown in Equation (4).

$\phi(x) = (\alpha + \tanh(\beta x)) * \gamma x$ ...(3)

$\phi'(x) = \gamma(\alpha + \tanh(\beta x) + \beta x \operatorname{sech}^2(\beta x))$ ...(4)

By updating the values of the parameters α, β and γ we can make the function Φ behave like a MISH activation function. The updated function Φ and it's derivative is shown in Equation (5) and (6), where $\alpha = 1$, $\beta = 1$ and $\gamma = \frac{1}{2}$

$\phi(x) = (1 + \tanh(x)) * x/2$ ...(5)

$\phi'(x) = (1 + \tanh(x) + x \operatorname{sech}^2(x))/2$ ...(6)

For the detailed visual analysis of the behavior of our APTx its graph is shown in Figure 1, and the graph of its derivative is shown in Figure 2.
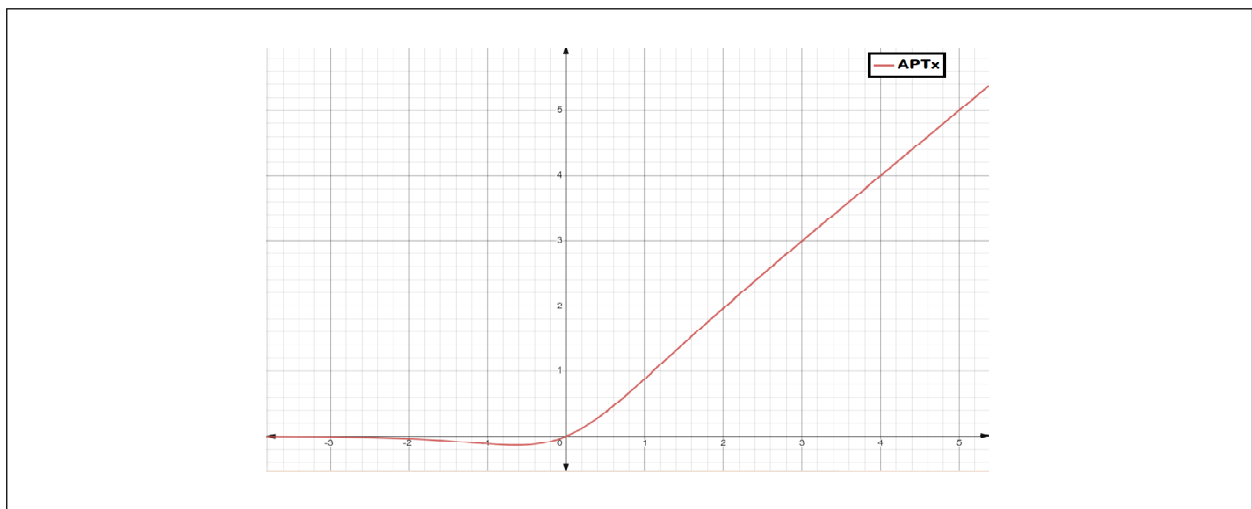

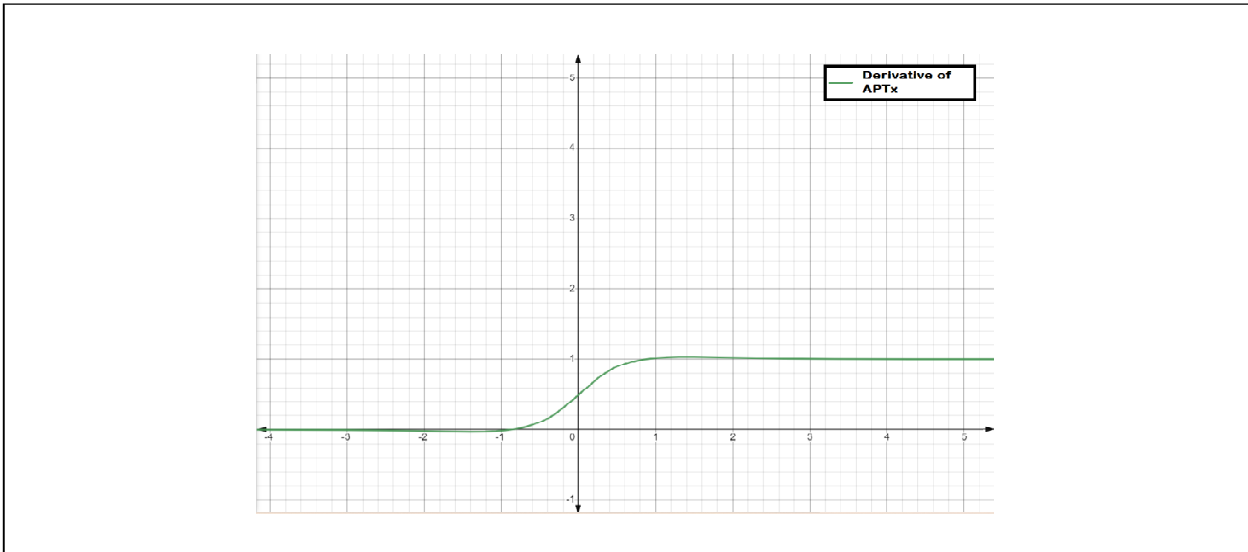**Figure 1: Graph of our Proposed APTx Activation Function at $\alpha = 1$, $\beta = 1$ and $\gamma = \frac{1}{2}$**

Figure 2: Graph of the Derivative of APTx Activation Function at α = 1, β = 1 **and** γ = ½

Although one decides activation functions based on the type of the problem statement, there are some popular activation functions whose comparisons were already done in existing research works. First, we discuss how MISH activation function is better than SWISH, ELU, Leaky-ReLU, ReLU, Tan*h* and Sigmoid activation function for general scenarios. Afterwards, we compared the MISH activation function with our proposed APTx function.

4. Comparative Analysis of Existing Activation Functions

The sigmoid activation function is mathematically expressed in Equation (7), and comparison of its derivative with the derivative of tan*h* is shown in Figure 3. One can easily notice in Figure 3 that the range of tan*h* derivatives is larger than sigmoid derivatives, but for numbers away from zero both tan*h* and sigmoid have very less output, this introduces the Vanishing Gradient Problem (Szandala, 2021) in the larger neural networks.

$Sigmoid\,(x) = 1/(1+e^{-x})$ ...(7)
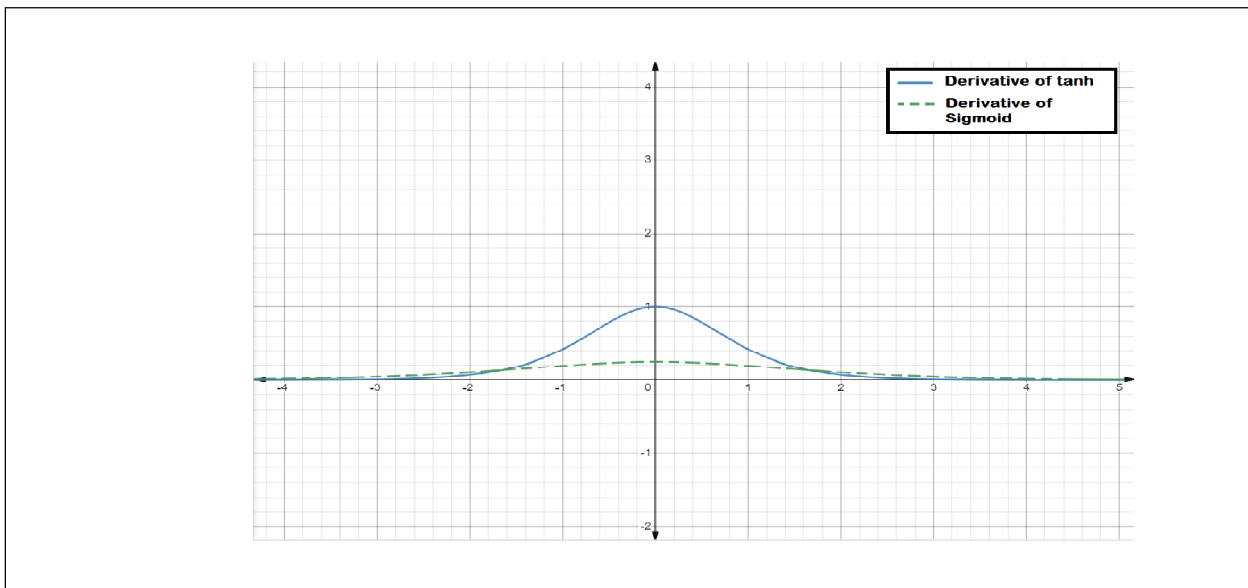

Figure 3: Graph Showing Derivatives of tan*h* and Sigmoid Activation Functions

The ReLU activation function provided a solution to the Vanishing Gradient Problem at least for the positive inputs (Glorot *et al.*, 2011; Sun *et al.*, 2014), but for the negative inputs it suffers from the Dying-ReLU problem (Szandala, 2021), as its derivative for negative value is Zero. Leaky-ReLU (Mass *et al.*, 2013) was able

to solve the Dying-ReLU problem upto some extent. ELU (Clevert *et al.,* 2015) showed better performance than Leaky-ReLU in most of the tasks as it tends to converge cost to zero faster and produce accurate results. For the positive input ReLU, Leaky-ReLU, and ELU all behave in the same manner, but the difference lies for the non-positive values as shown in Figure 4 and also presented in Equations (8), (9), and (10) respectively.
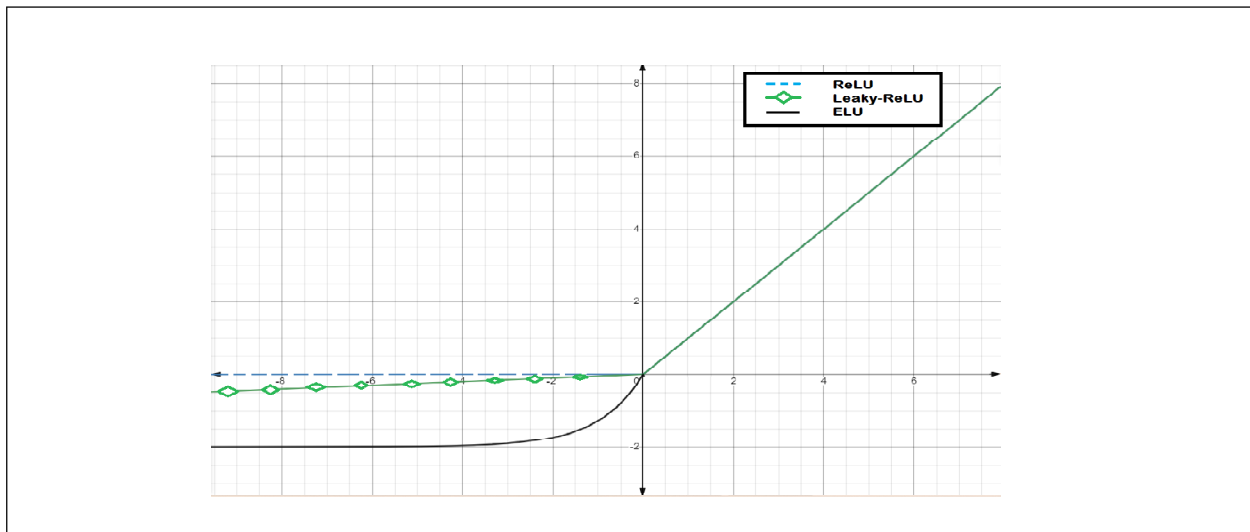

Figure 4: Graph of ReLU, Leaky-ReLU (with α = 0.05), and ELU (with α = 2)

$ReLU(x) = max\,(0,\,x)$ ...(8)

$Leaky\,ReLU(x) = \{\alpha x,\, x<=0\}$, and $\{x,\, x>0\}$ ...(9)

$ELU(x) = \{\alpha(e^x-1),\, x<=0\}$, and $\{x,\, x>0\}$ ...(10)

SWISH activation function (Ramachandran *et al.,* 2017) performs better than ReLU activation function, and also its variants because none of these variants have managed to replace the inconsistent gains (i.e., calculation of derivatives). SWISH can be considered a type of self-gated function, also expressed in Equation (11).

$SWISH(x) = x \times Sigmoid\,(x)$ ...(11)

Although introduction of SWISH solved both vanishing gradient and providing consistent gains, development of MISH activation function (Misra, 2019) turned out to provide equivalent and in many tasks it had even better performance than SWISH activation function. Its mathematical form is presented in Equation (12).

$MISH(x) = x \times tanh\,(\ln(1+e^x))$ ...(12)

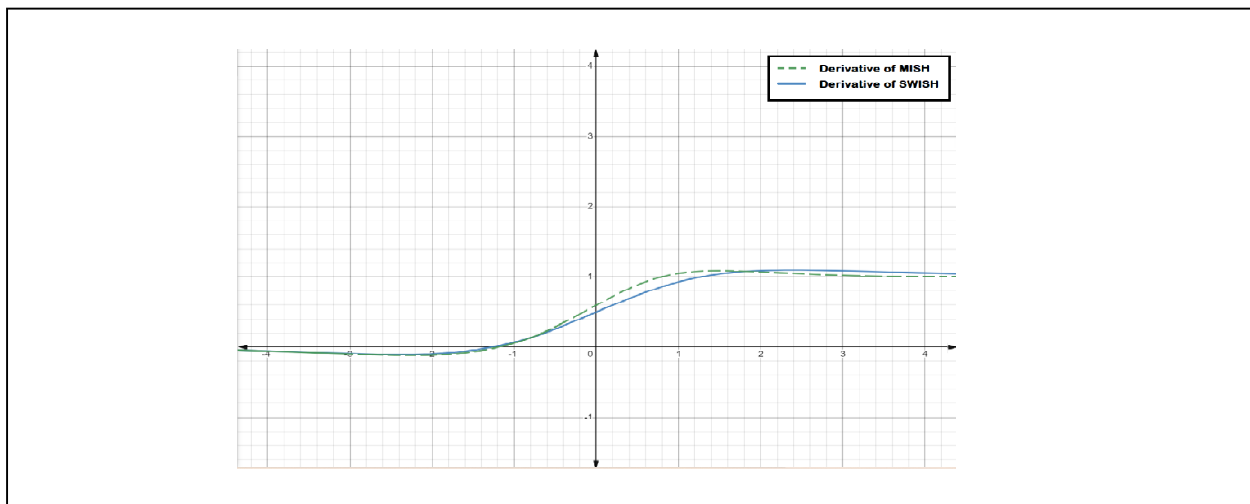Graphs of the derivatives of SWISH and MISH functions are plotted in Figure 5.

Figure 5: Graph of the Derivatives of SWISH and MISH Activation Functions

5. Comparative Analysis of MISH with Proposed APTx

During the forward propagation the mathematical operations required to calculate APTx expressed in Equation (5) are lesser than the MISH activation function shown in Equation (12). But, similar to the MISH function, APTx is bounded below and unbounded above.

The biggest advantage of APTx lies during the training phase while performing backpropagation. Back Propagation requires calculation of derivatives for each epoch and APTx requires fewer mathematical operations to compute its derivative than the MISH activation function. The derivative of MISH is expressed in Equation (13), and for comparative analysis the derivative of APTx is stated again in the Equation (14), where $\alpha = 1$, $\beta = 1$ and $\gamma = \frac{1}{2}$.

$$MISH'(x) = (e^x(4(x + 1) + 4e^{2x} + e^{3x} + e^x(4x + 6)))/(2e^x + e^{2x} + 2)^2 \qquad \text{...(13)}$$

$$\phi'(x) = (1 + \tanh(x) + x \operatorname{sech}^2(x))/2 \qquad \text{...(14)}$$

Interestingly, despite the fact that the derivative of the APTx function requires fewer operations than the derivative of MISH and also SWISH. The derivative graphs of APTx and MISH are presented in Figure 6 showing similar behavior, useful for backpropagation.
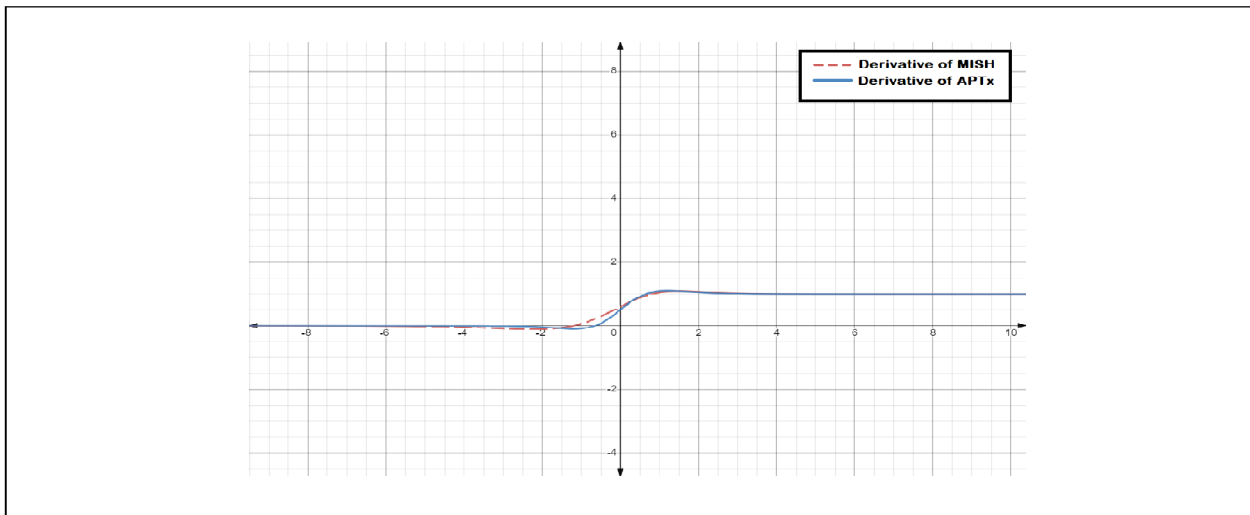

Figure 6: Graph of Derivatives of MISH and APTx with $\alpha = 1$, $\beta = 1$ and $\gamma = \frac{1}{2}$

Even more overlapping between MISH, and APTx derivatives can be generated by varying values for α, β and γ parameters.

6. Conclusion

MISH has similar or even better performance than SWISH which is better than the rest of the activation functions. Our proposed activation function APTx behaves similar to MISH but requires lesser mathematical operations in calculating value in forward propagation, and derivatives in backward propagation. This allows APTx to train neural networks faster and be able to run inference on low-end computing hardwares such as neural networks deployed on low-end edge-devices with Internet of Things.

References

Agarap, A.F. (2018). Deep Learning Using Rectified Linear Units (ReLU). arXiv preprint arXiv:1803.08375. March 22.

Clevert, D.A., Unterthiner, T. and Hochreiter, S. (2015). Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). arXiv preprint arXiv:1511.07289, November 23.

Glorot, Xavier., Bordes, Antoine. and Bengio, Yoshua. (2011). Deep Sparse Rectifier Networks. In Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP, 15, 315–323.

Maas, Andrew, L., Hannun, Awni, Y. and Ng, Andrew, Y. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models. In ICML, 30.

Misra, D. Mish. (2019). A Self Regularized Non-monotonic Activation Function. arXiv preprint arXiv:1908.08681. August 23.

Ramachandran, P., Zoph, B. and Le, Q.V. (2017). Swish: A Self-gated Activation Function, 7(1), 5, arXiv preprint arXiv:1710.05941. October 16.

Sun, Yi., Wang, Xiaogang. and Tang, Xiaoou. (2014). Deeply lEarned Face Representations Are Sparse, Selective, and Robust. arXiv preprint arXiv:1412.1265.

Szandala, Tomasz. (2021). Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks. In *Bio-inspired Neurocomputing*, 203-224. Springer, Singapore.

Vinod, Nair. and Geoffrey E. Hinton. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines. 27th International Conference on International Conference on Machine Learning, 807-814.