





# Effect of the Slope of Symmetric Saturated Activation Functions on Deep Learning

Maja Lutovac Banduka<sup>1</sup>  [0000-0003-4446-706X], Vladimir Poučki<sup>2</sup>  [0009-0007-0482-2449],  
Vladimir Mladenović<sup>3</sup>  [0000-0001-8530-2312] and Miroslav Lutovac<sup>4\*</sup>  [0000-0002-3113-9414]

<sup>1</sup> RT-RK LLC Computer Based Systems, Belgrade, Serbia

<sup>2</sup> Poucki Technologies GmbH, Germering, Germany

<sup>3</sup> University of Kragujevac, Faculty of Technical Sciences Čačak, Čačak, Serbia

<sup>4</sup> Academy of Engineering Sciences of Serbia, Belgrade, Serbia

\* [lutovac@gmail.com](mailto:lutovac@gmail.com)

**Abstract:** *It is presented how the slope of symmetric activation functions with saturation affects class detection using symbolic analysis. Different activation functions can be used to increase the most likely detected classes. The main result is the determination of the highest slope of the activation function and the lowest slope of the activation function in terms of the number of neurons in the layer.*

**Keywords:** *class detection; probability; automated drawing; symbolic solving of the neural network*

## 1. INTRODUCTION

The activation function is usually the same during the training phase with known images and the detection of an unknown class. The training phase can take longer because it usually starts with randomly generated weighted coefficients until they reach a steady state for the second phase.

It is not necessary to use the same activation function in the training phase and the class detection phase because it approaches the steady state very slowly for known images. This paper presents a slope analysis of symmetric saturated activation functions on deep learning for a faster training phase.

An overview of the symbolic analysis of neural networks is presented in [1]. Also, the code can be downloaded from the same site [1]. The same numerical example is used to demonstrate the proof of concept as in [2].

In this paper, we start with an application package [3] in an environment based on the Wolfram language [4]. We initialize the primary palette which loads the built-in drawing knowledge as presented in [3]. We use the palette for non-linear schemes. Element options are changed using the palette extension. The graphical user interface (GUI) enables fine-tuning of the presentation of elements.

The original software may be embedded with additional functions, such as copy-move-paste, left-to-right, or up-down, or rotating elements several times for 90°. Any element of the schematic description can be easily replaced by another element. The system solution is obtained

in the time domain by simply calling the solve button.

Any element from the system specification in the netlist can be replaced by one or more related components as in [5]. Schematic description is adapted to specific analysis [5].

A symbolic analysis of neural networks is also presented in [6]. The advantages of symbolic analysis of neural networks are shown in [7].

Tips and tricks for fixed point implementation are presented in [8] and [9].

The second section presents the properties of activation functions. The third part discusses the symbolic design method. It continues with the main results and achievements of the proposed original method.

## 2. ACTIVATION FUNCTIONS

There are a number of activation functions, such as the most popular sigmoid, hyperbolic tangent, ReLU and Leaky ReLU. In this paper, we analyze only symmetric saturated activation functions; which are hyperbolic tangents and a modified sigmoid with symmetric properties. ReLU and Leaky ReLU functions are not symmetric. We expect that positive and negative values of the input signal contribute equally to the detected classes. This is the main reason why we do not consider ReLU or classical sigmoid functions.

As explained in [8] and [9], to speed up the computation, we use a small number of binary shifts and adders. To find the best function, we analyze a linear function from some input signal range and with saturation to ensure non-linearity.

We do not consider the training phase, but consider the existing trained network to find what happens to the discovered classes if we change the activation function. Initial training classes consist of 19 type 1 classes, 3 type 2 classes, 18 type 3 classes, and 60 type 4 classes. The main achievement we expect is to find a symmetric activation function with saturation and linear properties for values around 0 of the input signal.

Note that in this paper we use the term linear activation function for a symmetric activation function with linear behavior from -1 values to 1 over 0, and with saturated values of -1 and 1 outside that region.

The first step is to determine the properties of the activation functions.

### 2.1. Properties

Let us denote the activation function by the symbol  $f$ . The symmetric activation function satisfies the following conditions:

$$f(-x) = -f(x). \tag{1}$$

The symmetric activation function satisfies the zero condition:

$$f(0) = 0. \tag{2}$$

The saturated activation function is bounded by an upper value of 1:

$$f(x) \leq 1. \tag{3}$$

The first derivative of the activation function is always positive or 0:

$$f'(x) \geq 0. \tag{4}$$

The maximum slope of the activation function is 1, which is equal to the slope of  $\tanh'(0)$ :

$$\max(f'(x)) = 1. \tag{5}$$

The maximum slope is 1, because for an input to the activation function that can be 1, the saturated value must be 1. A large slope is not expected because the saturation will become 1 before the maximum input value of 1.

The minimum slope of the activation function depends on the number of neurons in the layer:

$$\min(f'(x)) = \frac{1}{1+\#\text{Neurons In Layer}}. \tag{6}$$

For a neural network with 4 neurons per layer, the minimum slope is  $1/(4+1)=1/5$ . The maximum input value of the activation function is 4 for each neuron in the layer multiplied by the maximum weight parameter, which is also 1, plus one for the bias parameter. For a maximum input value to the activation function of 5, the activation function

must reach a maximum value of 1. The slope can be lower, but the activation function cannot reach the maximum value of 1.

### 2.2. Examples

For a neural network with four neurons per layer, we present the possible activation functions from lowest to highest slope. The most popular feature is the tanh shape. A linear activation function that has the same slope for  $x=0$  has the largest slope. Of course, the non-linearity is ensured by saturation for input values greater than 1 and less than -1. The main characteristics of the activation function with the highest slope are shown in Table 1.

**Table 1.** The function with the highest slope

Function	$\tanh_{\alpha}$
$f(x)$	$\begin{cases} f = -1, & x < -1 \\ f = x, & -1 \leq x \leq 1 \\ f = 1, & 1 < x \end{cases}$
$f'(x)$	$\begin{cases} f' = 0, & x < -1 \\ f' = 1, & -1 \leq x \leq 1 \\ f' = 0, & 1 < x \end{cases}$

The characteristics of the most popular activation function are shown in Table 2. The maximum value is 0.9999 because the input value cannot be greater than 5. Also, the slope of the function is slightly greater than 0 for extreme input values.

**Table 2.** The most popular function

Function	$\tanh(x)$
$f(x)$	$\begin{cases} -1 < f < -0.9999, & x < -5 \\ -0.9999 < f < 0.9999, & -5 \leq x \leq 5 \\ 0.9999 < f < 1, & 5 < x \end{cases}$
$f'(x)$	$\begin{cases} 0 < f' < 0.0002, & x < -5 \\ 0.0002 < f' < 1, & -5 \leq x \leq 5 \\ 0 < f' < 0.0002, & 5 < x \end{cases}$

Table 3 presents the linear activation function for input values between -2 and 2, as well as the slope of the modified logistic sigmoid activation function. Nonlinearity is provided by saturation for input values greater than 2 and less than -2.

**Table 3.** A function with a slope of 0.5

Function	$(2\sigma(x)-1)_{\alpha}$
$f(x)$	$\begin{cases} f = -1, & x < -2 \\ f = 0.5x, & -2 \leq x \leq 2 \\ f = 1, & 2 < x \end{cases}$
$f'(x)$	$\begin{cases} f' = 0, & x < -2 \\ f' = 0.5, & -2 \leq x \leq 2 \\ f' = 0, & 2 < x \end{cases}$

The main features of the very popular Logistic Sigmoid activation function are presented in Table 4. The maximum value is 0.9866 because

the input value cannot be greater than 5; the slope of the function is slightly greater than 0 for extreme input values.

**Table 4.** The modified Logistic Sigmoid function

Function	$(2\sigma(x)-1)$
$f(x)$	$\begin{cases} -1 < f < -0.9866, & x < -5 \\ -0.9866 \leq f \leq 0.9866, & -5 \leq x \leq 5 \\ 0.9866 < f < 1, & 5 < x \end{cases}$
$f'(x)$	$\begin{cases} 0 < f' < 0.0133, & x < -5 \\ 0.0133 < f' < 0.5, & -5 \leq x \leq 5 \\ 0 < f' < 0.0133, & 5 < x \end{cases}$

The main characteristics of the linear activation function, which has a smaller slope, are shown in Table 5. Nonlinearity is provided by saturation for input values greater than 5 and less than -5. Saturation will never happen because input values are never greater than 5. So this is actually an activation function with linear behavior for all input values.

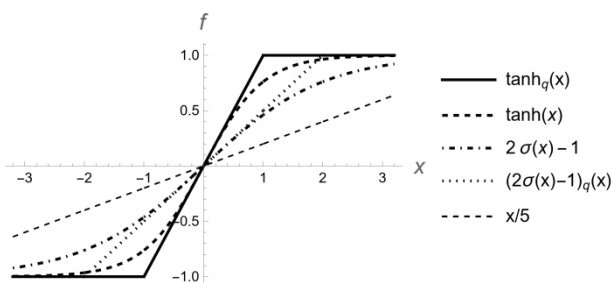
This means that the activation function with the smallest slope is determined by the number of neurons plus 1 (for the bias parameter).

Actually, the case with the activation function with the lowest slope is no longer the activation function of the neuron because it never has nonlinear behavior, but this is the limiting case that occurs with the network in the limiting case.

**Table 5.** The function with the least slope

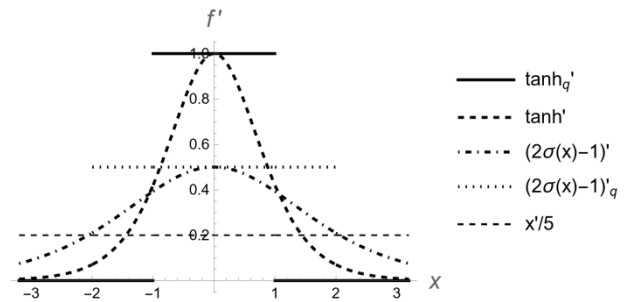
Function	$x/5$
$f(x)$	$\begin{cases} f = -1, & x < -5 \\ f = x/5, & -5 \leq x \leq 5 \\ f = 1, & 5 < x \end{cases}$
$f'(x)$	$\begin{cases} f' = 0, & x < -5 \\ f' = 0.2, & -5 \leq x \leq 5 \\ f' = 0, & 5 < x \end{cases}$

All introduced activation functions are illustrated in Fig. 1.



**Figure 1.** Activation functions  $f$ .

The slope of all activation functions is shown in Fig. 2. A better overview is for inputs between -3 and 3, because the other values are rare in practical examples.



**Figure 2.** Slope of activation functions  $f$ .

The slope of the symmetric saturated activation function with linear behavior around 0 is between  $1/(1+\text{number of neurons in one layer})$  and 1. Note that the slope of popular activation functions (tanh and logistic sigmoid) is less than  $1/(1+\text{number of neurons in one layer})$  for input values greater than 1 and less than -1.

### 3. RESULTS

The training strategy can be based on a random number generator instead of zero initial weights and bias parameters. Assume that these initial parameter solutions correspond to the exact values of the expected classes. Initial training classes according to [2] consist of 19 type 1 classes, 3 type 2 classes, 18 type 3 classes, and 60 type 4 classes. We assume two nonzero input parameters and four the expected classes.

For simplicity, we generate a neural network with 4 inputs, 4 neurons per layer, two hidden layers (for deep learning) and one output layer with four detected classes. For the two unnecessary inputs, we specify zero weight and bias parameters, so that these zero parameters do not change during training. The results of processing with five activation functions are shown in Table 6.

**Table 6.** Activation functions, integral of slope, and probability  $\{p_1, p_2, p_3, p_4\}$

Function	$\int_0^1$	$\int_0^{1.5}$	Probability
$\tanh_q$	1.00	1.00	{0.16, 0.02, 0.16, <b>0.65</b> }
$\tanh$	0.76	0.91	{0.18, 0.03, 0.18, <b>0.60</b> }
$(2\sigma-1)_q$	0.50	0.75	{0.22, 0.05, 0.24, <b>0.49</b> }
$2\sigma-1$	0.46	0.64	{0.26, 0.06, 0.28, <b>0.40</b> }
$x/5$	0.20	0.30	{ <u>0.33</u> , 0.10, <u>0.34</u> , <b>0.24</b> }

The integral of the first derivative is used to illustrate the slope.

It is evident from Table 6 that a higher slope gives a higher probability of the fourth class.

Table 7 gives the expected classes versus the exact known classes during training. We assume that tanh corresponds to the correct detection of all classes as in [2].

**Table 7.** Number of discovered classes

Function	Class 1	Class 2	Class 3	Class 4
Exact	<b>19</b>	<b>3</b>	<b>18</b>	<b>60</b>
$\tanh_q$	17(-2)	2(-1)	16(-2)	65(+5)
$\tanh$	19(+0)	3(+0)	18(+0)	60(+0)
$(2\sigma-1)_q$	22(+3)	5(+2)	24(+6)	49(-11)
$2\sigma-1$	26(+7)	6(+3)	28(+10)	40(-20)
$x/5$	32(+13)	10(+7)	34(+16)	24(-36)

If we use a large slope of the activation function, the detected most likely class is larger than the exactly known classes. Smaller slopes of the activation functions produce a smaller number of most likely classes. The smallest slope gives half the expected class, while the first and third classes are almost double the known classes.

This is the explanation why the nonlinear activation function is important for neural networks.

#### 4. DISCUSSION

The proposed original method establishes a relation between the most likely detected classes with respect to the slope of activation functions for neural networks that have equally expected positive and negative input values. Another achievement is the determination of the slope of the activation function with saturation for two limiting cases, one for the highest slope and the other for the lowest slope that depends on the number of neurons in the layer.

During the training phase of the neural network, for known classes, the activation function can be chosen to best fit certain classes. If the number of detected classes is less than the number of test cases, the slope of the activation function should be increased. If the number of discovered classes is greater than the number of test cases, the slope of the activation function should be reduced.

During the training phase, we can know the exact number of each possible class. So, after the training phase, we can know the accuracy of the detected classes, and therefore choose the most appropriate activation function.

The main advantage of the proposed approach is faster training and faster detection of unknown classes because the calculation of the activation function consists only in using the input value for slope 1, i.e. the binary shift (multiplying by 1/2) for the lowest slope of the symmetric saturated activation function with linear behavior around 0. The class detection accuracy is very similar to the

corresponding hyperbolic tangent or modified logistic sigmoid function with the same slope at 0.

#### 5. CONCLUSION

The paper shows the influence of the slope of the symmetric activation function with saturation for more precise detection of the tested known classes. Future work will be to find the impact of asymmetric activation functions for asymmetric inputs, such as inputs that are all positive. Appropriate choice of symmetric saturated activation function with linear behavior around 0 ensures faster computation and consequent power reduction in hardware implementations.

#### REFERENCES

- [1] Lutovac Banduka, M., Franc, I., Milićević, V., Zdravković, N., & Dimitrijević N. (2023). *Symbolic analysis of classical neural networks for deep learning*, 2023110446, Online doi.org/10.20944/preprints202311.0446.v1
- [2] Bernard, E. (2022). *Introduction to machine learning*, Champaign, IL, USA: Wolfram Media.
- [3] Lutovac Banduka, M., Milosevic, D., Cen, Y., Kar, A., & Mladenovic, V. (2023). Graphical user interface for design, analysis, validation, and reporting of continuous-time systems using Wolfram language, *JCSC*, #2350244, 32(14), 1-15.
- [4] Wolfram, S. (2023). *An elementary introduction to the Wolfram language*, 3<sup>rd</sup> ed. Champaign, IL: Wolfram Media.
- [5] Lutovac Banduka, M., Simović, A., Orlić, V., & Stevanović, A. (2023). Dissipation minimization of two-stage amplifier using deep learning, *Serbian Journal of Electrical Engineering*, 20(2), 129-145.
- [6] Milićević, V., Lutovac Banduka, M., Franc, I., Zdravković, N., & Dimitrijević, N. (2025). Symbolic analysis of classical neural networks for deep learning, *International Journal for Quality Research*, 19(1), in press.
- [7] Lutovac Banduka, M., Franc, I., Milićević, V., Zdravković, N., & Dimitrijević N. (2024). *The effect of the number of hidden layers and activation functions on deep learning*, unpublished.
- [8] Lutovac Banduka, M., Lutovac, M. (2024). *How to design multiplierless neural networks for deep learning?*, EasyChair Preprint #13105, 1-4.
- [9] Lutovac Banduka, M., Lutovac, M. (2024). Multiplierless neural networks for deep learning, *Mediterranean Conference on Embedded Computing (MECO)*, 11-14 June 2024, Budva, 262-265.