



UDK: 004.8.032.26
COBISS.SR-ID 148900617
DOI: 10.5281/zenodo.12685976
Original scientific paper

BILATERAL EXPONENTIAL AS SIGMOID IN MULTILAYERED NEURONAL NETWORKS

Dejan Djukic¹⁰²; Stefan Popovic¹⁰³

Abstract

We are witnessing a flurry of emerging smart technologies, facilitating human existence in general, and in particular in organising and managing life in cities. Establishing correct models of the environment, and producing a reliable mechanism for information association is an essential requirement for describing a technology as smart. A very popular class of models nowadays is formed by artificial neuronal networks arranged as multilayered perceptrons. The quality of a neuronal model depends essentially on the successful learning of the mathematical relation governing the information processing. The learning in multilayered perceptron networks is achieved through numerical optimisation methods, being performed repetitively in large number of cycles. Therefore, it is important to strive for a parsimonious use of computational resources. In this work, we propose the use of the bilateral exponential function as the non linear sigmoid map in perceptrons. This function is very similar in form to the usually employed sigmoid functions, such as the hyperbolic tangent. Yet, the computation of the bilateral exponential function saves the computational effort of one special function computation, compared to the hyperbolic tangent. In addition, gradient methods of machine learning require also the computation of the sigmoid function derivative. Here again, the use of the bilateral exponential saves an additional multiplication operation. Whilst it may appear trivial, one needs to be aware that the savings achieved in one neuron during one cycle of training are multiplied by the number of neurons times the number of cycles. In practice, this may amount to an improvement of computational speed of orders of magnitude, when compared to the networks using a conventional sigmoid function. In this work, we describe the newly proposed function, we show its adequacy of performance when compared to the hyperbolic tangent and some other sigmoid functions, and we estimate the computational savings obtained by its use.

Key words: smart city, intelligent systems, artificial neuronal networks, perceptron networks, machine learning, steepest descent optimisation, sigmoid functions

¹⁰² Dejan Djukic, 1965, PhD, Faculty of Information Technologies, dejan.djukic@alfa.edu.rs
<https://orcid.org/0000-0001-7581-148X>

¹⁰³ Stefan Popovic, 1983, MSc., Faculty of Information Technologies, +381638119729,
stefan.popovic@alfa.edu.rs <https://orcid.org/0000-0002-5288-4560>



Introduction

A prime model of artificial neurons has been the perceptron, proposed by McCulloch and Pitts in 1948. In their work, the computation performed by the neuron has been modelled as a function performing a non-linear map on the linear combination of the inputs. [1]

$$y = (X) = \sigma(WX - W_0) \quad (1)$$

Although initially the hard switching has been used as the non-linear transformation, other non-linearities, involving smooth functions, so called soft sigmoids, have been used almost exclusively in the contemporary practice. Although the most often used sigmoid function has been the hyperbolic tangent, other smooth sigmoids have also been tried. In this work, the authors propose yet another soft sigmoid function, called bi-lateral exponential function. In the sequel, it has been shown that this newly proposed function has similar properties as the sigmoid functions that are already in use. However, due to its modular construction, it may offer some advantages in being more adjustable and adaptable to some special applications.

As a basis for modeling neural systems, they proposed models of neurons in the form of binary boundary devices and stochastic algorithms that include instant changes from 0 to 1 or from 1 to 0 in the output of the neurons. The results of Rumelhart, Hinton, and Williams [2], dealing with the development of training algorithms for multilayer perceptrons, changed significantly the state of development of neural networks. Their basic method, often called the generalized delta rule for backward propagation learning, provides an effective training method for multilayer networks. Although it cannot be shown that this training algorithm converges to a solution in terms of analogous proof for a single-layer perceptron, the generalized delta rule has been successfully used in a number of problems of practical interest. This success established perceptron-like multilayer machines as one of the major neural network models currently in use. [3]

The application of neural networks is very broad, there is almost no area in which they have not played a key role in the functioning of automatic control systems, object recognition, decision systems and many other areas that we are surrounded by.[4]

Comparative analysis of the sigmoid functions

The sigmoid functions tested in this work are presented below. It has been shown that all the sigmoid functions presented here have essentially the same behaviour as to the function domains and ranges, the asymptotic behaviour. It has also been shown that, whilst the form of the first derivative, which is necessary for implementing the error gradient back propagation optimisation algorithm for multi layer perceptron networks, the behaviour of the second derivative may differ significantly.

Together with the formulae related to these sigmoid functions, the graphs of the sigmoid functions together with the graphs of their first derivatives have also been presented. [5]



The following sigmoid functions have been tried:

1. Bi-lateral exponential function

$$\sigma_1(x) = \text{sgn}(x)(1 - e^{-|x|}) = \begin{cases} 1 - e^{-x} & x \geq 0 \\ e^x - 1 & x < 0 \end{cases} \quad (3)$$

$$\frac{\partial \sigma_1(x)}{\partial x} = e^{-|x|} = \begin{cases} e^{-x} & x \geq 0 \\ e^x & x < 0 \end{cases} \quad (4)$$

$$\frac{\partial^2 \sigma_1(x)}{\partial x^2} = -\text{sgn}(x)e^{-|x|} = \begin{cases} e^{-x} & x \geq 0 \\ e^x & x < 0 \end{cases} \quad (5)$$

$$\sigma_1(x) = 0 \quad \lim_{x \rightarrow +\infty} \sigma_1(x) = +1 \quad \lim_{x \rightarrow -\infty} \sigma_1(x) = -1 \quad (6)$$

$$\frac{\partial \sigma_1(x)}{\partial x} > 0 \quad \lim_{x \rightarrow +\infty} \frac{\partial \sigma_1(x)}{\partial x} = 0 \quad \lim_{x \rightarrow -\infty} \frac{\partial \sigma_1(x)}{\partial x} = 0 \quad (7)$$

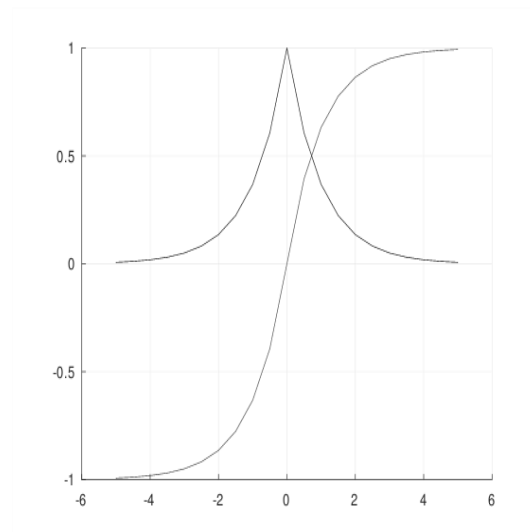


Figure 2: Bi-lateral sigmoid and its derivative

2. Hyperbolic tangent function

$$\sigma_2(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (8)$$

$$\frac{\partial \sigma_2(x)}{\partial x} = 1 - \tanh^2(x) = 1 - \sigma_2^2(x) = 1 - \left(\frac{e^x - e^{-x}}{e^x + e^{-x}} \right)^2 \quad (9)$$



$$\begin{aligned} \frac{\partial \sigma_2^2(x)}{\partial x^2} &= -2\sigma_2(x) \frac{\partial \sigma_2(x)}{\partial x} = -2\sigma_2(x)(1 - \sigma_2^2(x)) = \\ &= 2\tanh^3(x) - 2\tanh(x) = 2 \left(\frac{e^x - e^{-x}}{e^x + e^{-x}} \right)^3 - 2 \left(\frac{e^x - e^{-x}}{e^x + e^{-x}} \right) \end{aligned} \quad (10)$$

$$\sigma_2(x) = 0 \quad \lim_{x \rightarrow +\infty} \sigma_2(x) = +1 \quad \lim_{x \rightarrow -\infty} \sigma_2(x) = -1 \quad (11)$$

$$\frac{\partial \sigma_2(x)}{\partial x} > 0 \quad \lim_{x \rightarrow +\infty} \frac{\partial \sigma_2(x)}{\partial x} = 0 \quad \lim_{x \rightarrow -\infty} \frac{\partial \sigma_2(x)}{\partial x} = 0 \quad (12)$$

3. Arc tangent function

$$\sigma_3(x) = \frac{2}{\pi} \operatorname{atan} \left(\frac{\pi}{2} x \right) \quad (13)$$

$$\frac{\partial \sigma_3(x)}{\partial x} = \frac{1}{1 + \left(\frac{\pi}{2} x \right)^2} \quad (14)$$

$$\frac{\partial \sigma_3^2(x)}{\partial x^2} = \frac{-2 \left(\frac{\pi}{2} \right)^2 x}{\left(1 + \left(\frac{\pi}{2} \right)^2 x^2 \right)^2} \quad (15)$$

$$\sigma_3(x) = 0 \quad \lim_{x \rightarrow +\infty} \sigma_3(x) = +1 \quad \lim_{x \rightarrow -\infty} \sigma_3(x) = -1 \quad (16)$$

$$\frac{\partial \sigma_3(x)}{\partial x} > 0 \quad \lim_{x \rightarrow +\infty} \frac{\partial \sigma_3(x)}{\partial x} = 0 \quad \lim_{x \rightarrow -\infty} \frac{\partial \sigma_3(x)}{\partial x} = 0 \quad (17)$$

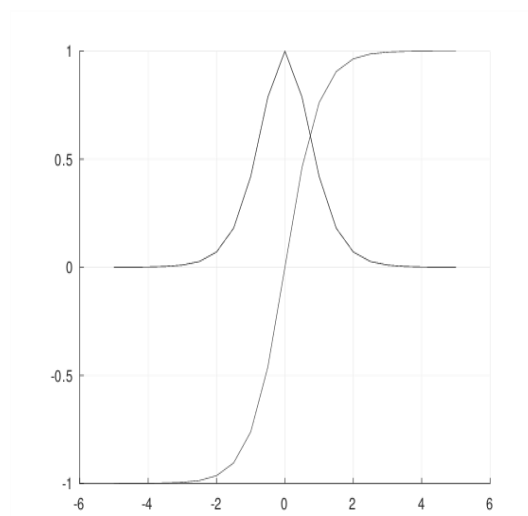


Figure 3: Hyperbolic tangent and its derivative

4. An algebraic function involving square root



$$\sigma_4(x) = \frac{x}{\sqrt{1+x^2}} \quad (18)$$

$$\frac{\partial \sigma_4(x)}{\partial x} = \frac{1}{(1+x^2)\sqrt{1+x^2}} = \frac{1}{(\sqrt{1+x^2})^3} = \frac{1}{(1+x^2)^{\frac{3}{2}}} \quad (19)$$

$$\frac{\partial \sigma_4^2(x)}{\partial x^2} = \frac{-3x}{(1+x^2)^2\sqrt{1+x^2}} = \frac{-3x}{(1+x^2)^{\frac{5}{2}}} \quad (20)$$

$$\sigma_4(x) = 0 \quad \lim_{x \rightarrow +\infty} \sigma_4(x) = +1 \quad \lim_{x \rightarrow -\infty} \sigma_4(x) = -1 \quad (21)$$

$$\frac{\partial \sigma_4(x)}{\partial x} > 0 \quad \lim_{x \rightarrow +\infty} \frac{\partial \sigma_4(x)}{\partial x} = 0 \quad \lim_{x \rightarrow -\infty} \frac{\partial \sigma_4(x)}{\partial x} = 0 \quad (22)$$

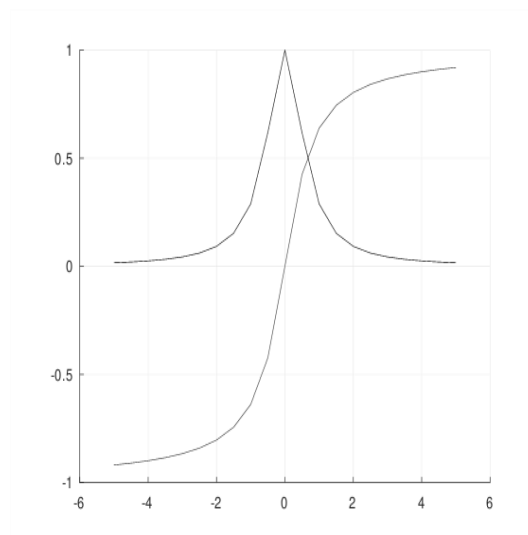


Figure 4: Arc tangent and its derivative

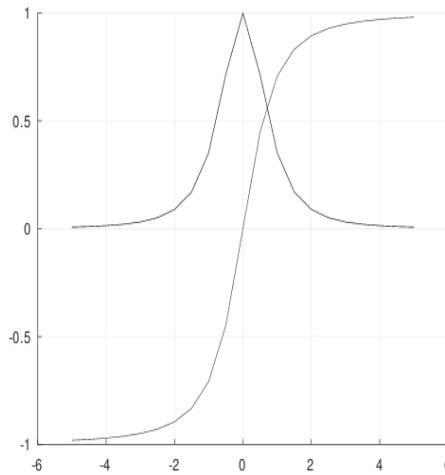


Figure 5: Algebraic function involving square root and its derivative

Numerical experiment with execution times

The execution times have been tested for the four sigmoid functions mentioned above. The numerical experiment has been conducted using the GNU Octave 6.2.0 environment for numerical computation, installed under Debian 11 operating system, on a PC computer having Intel i5 processor and 16 GB of RAM.

The experiment consisted from the computation of the outputs of the four sigmoid functions for 1000 randomly generated values, with normal distribution with zero mean and variance of 3.5. The same input values have been used for all the sigmoid functions. The experiment has been repeated 1000000 times, and the execution times have been averaged over all the repetitions. In order to reduce the influence of the time overhead induced by GNU Octave system, the execution times have also been established, in the same manner, for the identity map, i.e. for function $f(x) = x$, which requires no numerical computation. [6]

The results comprising the estimated computation times of the four sigmoid functions have been presented in Table 1. [7]

Table 2: Execution times

Sigmoid function	Estimated computation times (in $[\mu s]$)
Naively implemented bilateral exponential	6.9
Carefully implemented bilateral exponential	5.4
Naively implemented hyperbolic tangent function	8.5
Built - in hyperbolic tangent function	4.2
Arc - tangent function	12.0
Naively implemented algebraic function involving square root	8.3
Carefully implemented algebraic function involving square root	5.9



The results show that, whilst the hyperbolic tangent function has the shortest computation times when the built-in version is being used, the bi-lateral exponential is not lagging by much, and is effectively the runner-up in the speed contest. Another point to emphasise is that the naive implementation of the bi-lateral exponential function has the shortest computational time. This may be a significant feature in cases when experiments are being conducted with machine learning, where the algorithms have not yet been optimised for the speed of execution.

Conclusion

The sigmoid function in the artificial neuron is an essential part in the process of the transformation of the input values into the corresponding outputs. The properties and the performance of the sigmoid function influence significantly the performance of the machine learning process. In this work, the authors have proposed a new form of sigmoid function : the bi-lateral exponential function, and compare it to other three well known sigmoid functions. It has been shown that the basic properties of this newly proposed sigmoid function are essentially the same those of other, previously proposed sigmoid functions. It has also been shown that the computational speed of the bi-lateral exponential function is comparable, and in some cases even greater, than that of the often used hyperbolic tangent function, whilst surpassing the computational speed of the other sigmoids. The results of using the bi-lateral exponential function in multi layer perceptron neuronal networks are promising, and strongly suggest the need for further investigation.

References

- [1] McCulloch, W. S. , Pitts, W. A., A logical calculus of the ideas immanent in nervous activity, *Bulletin of Mathematical Biophysics, Volume 5* , (1943), pp. 115-133
- [2] Bottcher, W., Machado, P., Lama, N., & McGinnity, T. M. (2021, July). Object recognition for robotics from tactile time series data utilising different neural network architectures. In 2021 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.
- [3] Stefan Popović, Dejan Đukić, Sonja Đukić Popović, Lazar Kopanja, (2022) Preliminary Research on the Application of Neural Networks to the Combustion Control of Boilers with Automatic Firing, Proceedings of the 8 th Virtual International Conference on Science Technology and Management in Energy, 299 - 303, ISBN-978-86-82602-01-9, COBISS.SR-ID 119331337
- [4] Stefan Popović, Lazar Kopanja, Dejan Djukić, Sonja Djukić Popović, (2022) Neural networks and their application in object recognition, Proceedings of the 12th International Conference on Applied Internet and Information Technologies, AIIT 2022, 40 - 48, Zrenjanin, Serbia, ISBN 978-86-7672-361-4 , COBISS.SR-ID 81418505
- [5] Raul Rojas, *Neural Networks - A Systematic Introduction*, Springer-Verlag, Berlin Heidelberg, Germany, 1996
- [6] Blayo, F., Verleysen, M., *Les réseaux de neurones artificiels*, Presses universitaires de France, 1996



ALFA BK UNIVERSITY
FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
FACULTY OF COMPUTER SCIENCE

ALFATECH

Proceedings of Conference



Alfa BK Univerzitet

- [7] Han, J. Morag, C., The influence of the sigmoid function parameters on the speed of backpropagation learning, in *From Natural to Artificial Neural Computation Lecture Notes in Computer Science 930*, (Mira, J.; Sandoval, F. (eds.)), Springer-Verlag, Berlin Heidelberg, Germany, 1995, pp. 195-201