Journal of New Results in Science
https://dergipark.org.tr/en/pub/jnrs
Review Article
Open Access

E-ISSN:1304-7981 https://doi.org/10.54187/jnrs.1011739

An overview of the activation functions used in deep learning algorithms

Serhat Kılıçarslan[1], Kemal Adem[2], Mete Çelik[3]

Keywords:
Activation function,
Neural network,
Deep learning

Abstract — In deep learning models, the inputs to the network are processed using activation functions to generate the output corresponding to these inputs. Deep learning models are of particular importance in analyzing big data with numerous parameters and forecasting and are useful for image processing, natural language processing, object recognition, and financial forecasting. Also, in deep learning algorithms, activation functions have been developed by taking into account features such as performing the learning process in a healthy way, preventing excessive learning, increasing the accuracy performance, and reducing the computational cost. In this study, we present an overview of common and current activation functions used in deep learning algorithms. In the study, fixed and trainable activation functions are introduced. As fixed activation functions, sigmoid, hyperbolic tangent, ReLU, softplus and swish, and as trainable activation functions, LReLU, ELU, SELU and RSigELU are introduced.

Subject Classification (2020):

1. Introduction

Deep learning is used to produce solutions to real-world problems, inspired by artificial neural networks and the human brain. Today, deep learning architectures are actively preferred by researchers in many areas such as autonomous vehicles, image processing, signal processing, and prediction [1-3]. Deep learning architectures are consisted of such as the layers convolution, max-pooling, activation, dropout, normalization, flatten, full connection and softmax [4]. Activation functions have a significant contribution to the development of activation functions by scientists due to their positive contribution to learning on deep neural networks [5].

Activation functions are used in deep neural network architectures to decide whether to transfer information to the next neuron. In deep learning algorithms, activation functions have been developed by taking into account features such as performing the learning process in a healthy way, preventing overfitting, increasing the accuracy performance, and reducing the computational cost. Activation functions enable to work and learn better the deep learning models by revealing the hidden features of real-world problems. However, if deep learning models are run without choosing activation functions,

[1]serhat.kilicarslan@gop.edu.tr (Corresponding Author); [2]kemaladem@aksaray.edu.tr; [3]mcelik@erciyes.edu.tr
[1]Department of Informatics, Tokat Gaziosmanpaşa University, Tokat, Turkey
[2]Department of Software Engineering, Engineering Faculty, Aksaray University, Aksaray, Turkey
[3]Department of Computer Engineering, Engineering Faculty, Erciyes University, Kayseri, Turkey
Article History: Received: 18 Oct 2021 — Accepted: 08 Dec 2021 — Published: 31 Dec 2021

the desired success cannot be achieved with limited learning. Therefore, the deep learning architectures used cause it to behave like linear regression. For this reason, non-linear activation functions are preferred in deep neural network architectures. The first is the fixed activation functions, the second is the trainable activation functions [6]. In deep learning algorithms, back propagation algorithm is used to update the parameters. At the end of the update process, the derivative value of the functions is returned. For this reason, the activation functions used in deep learning architectures should have the ability to receive derivative continuously. Deep learning architectures have started to attract the attention of users with the use of activation functions. In the literature, deep neural networks were started to be trained by using ReLU, LReLU, ELU, PReLU, swish and similar activation functions, and success results were achieved [7-11]. Activation functions in deep learning architectures are expected to have features such as being derivative, non-linear, reaching the global optimum without being stuck in the local optimum.

In this study, general research is carried out on the proposed activation functions in the literature. However, non-linear activation functions used on deep learning architectures are examined. In the literature, many fixed and trainable activation functions have been proposed, and the number of studies conducted in this area by years is shown in Figure 1.

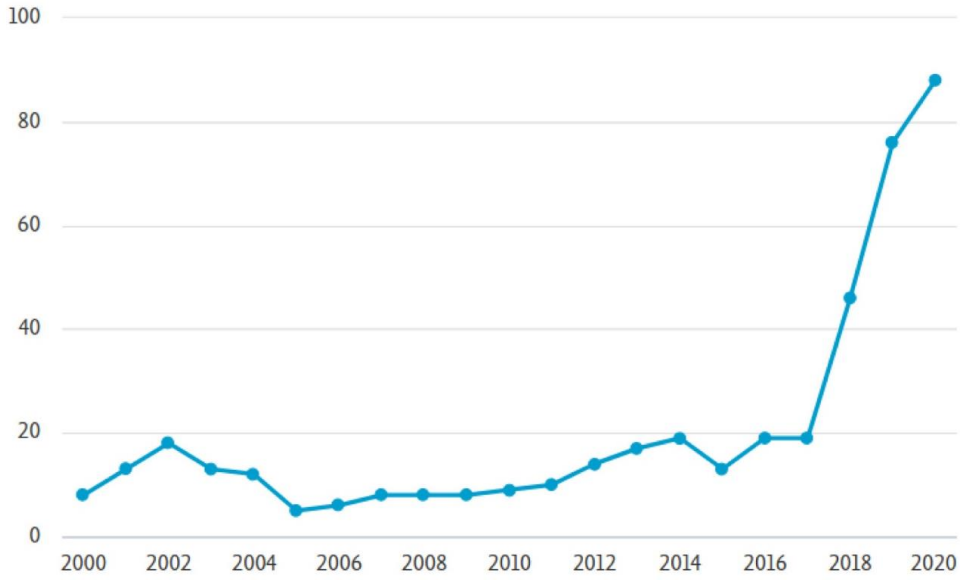


Figure 1. Number of studies conducted in the literature on activation functions [5]

In the literature, with the developing technology is increasing the number of studies on deep learning algorithms. In deep learning algorithms, features such as the ability to perform the learning process properly and increase the accuracy performance are important. Therefore, many studies on the development of the activation function are appear in the literature.

2. Fixed Parameter Activation Functions

Non-linear fixed parameter activation functions has work actively on deep learning architectures without taking any external parameters. Fixed parameter activation functions used in the literature are expressed as sigmoid, hyperbolic tangent, ReLU, softplus, swish [7-11].

2.1. Sigmoid Activation Function

In deep learning architectures, back propagation algorithm has prefer in order to realize the learning process between neurons. The back propagation algorithm are performs the updating process by taking

derivatives of the parameters in the architectures. Therefore, it is important that in the activation functions are derivation. In linear activation functions, with the help of back propagation algorithm is returned a fixed value when the derivative is carried out. Therefore, it cannot perform the learning process on deep learning architectures. In order to overcome this problem, a non-linear sigmoid activation function, which can derivative in figure 2, is proposed [12, 13]. In Equations 2.1 and 2.2, the normal and derivatives of the sigmoid activation function are given.

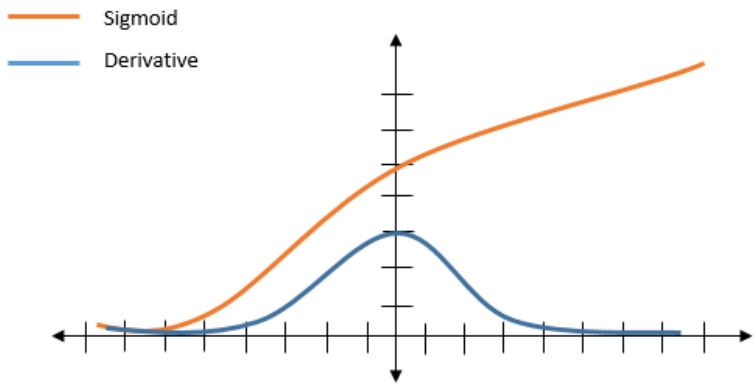


Figure 2. Sigmoid activation function

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

$$\frac{df(x)}{dx} = \frac{e^{x}}{(1 + e^{x})^{2}} \tag{2.2}$$

In Equation 2.1, parameter x represents the input data in the sigmoid activation function. In figure 2 shows the normal and derivative forms of the sigmoid activation function. Sigmoid is the most commonly preferred activation function among non-linear functions. In Figure 2, the termination of the derivative operation after the interval [-5.5] causes the learning process to stop and reveals the vanishing gradient problem [14 ,15]. In addition, back propagation algorithm is used to be the derivative result small of the sigmoid activation function and to update the parameters in the neural network structure. Therefore, the weights will not be updated at the desired level and the learning process in neural networks will be interrupted. Because of this problem, the sigmoid activation function is not as popularly used as it used to be.

2.2. Hyperbolic Tangent Activation Function

The hyperbolic tangent activation function is a non-linear activation function that and can perform the derivative operation. The hyperbolic tangent activation function is in structure similar to the sigmoid activation function. In Figure 3 and Equation 2.3, 2.4, the hyperbolic tangent activation function is seen its normal and derivative states [16].

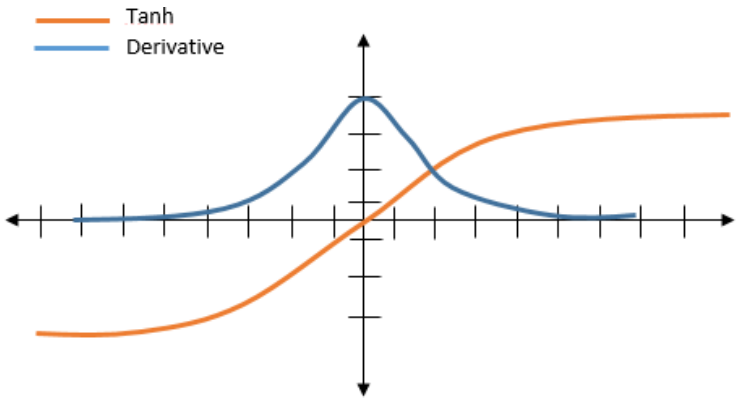


Figure 3. Hyperbolic tangent activation function

$$f(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.3}$$

$$\frac{df(x)}{dx} = 1 - f(x)^{2} \tag{2.4}$$

When Figure 3 is examined, the hyperbolic tangent activation function are seen in the sigmoidal curve and the S-shaped curve. While the sigmoid activation function can produce output between [0,1], the hyperbolic tangent activation function can produce values between [-1,1]. It is observed that the learning process continues on negative values due to the hyperbolic tangent activation function value range. However, it faces the vanishing gradient problem due to the derivative values approaching zero after [-1,1] values [14].

2.3. ReLU Activation Function

In sigmoid and hyperbolic tangent activation functions, the vanishing gradient problem arises because it cannot be derivative after a certain threshold value. Therefore, the ReLU activation function has been developed in order to find a solution to the vanishing gradient problem. The ReLU activation function is a non-linear activation function that and can perform the derivative operation. Equations 2.5 and 2.6 show the ReLU activation function [7].

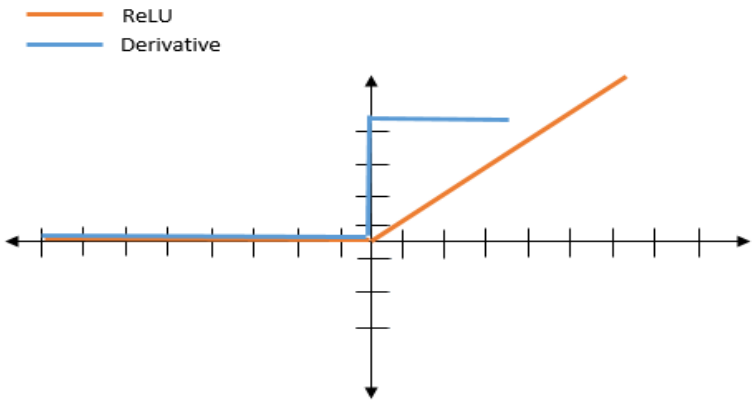


Figure 3. ReLU activation function

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{2.5}$$

$$\frac{df(x)}{dx} = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases} \tag{2.6}$$

ReLU is widely used because it overcomes the vanishing gradient problem. However, the ReLU activation function is faces the problem of negative region. Since negative values are set to zero, the derivative of the values cannot be taken and the learning process slows down [4, 18]. One of the biggest advantages of the ReLU activation function is that the computational load is low compared to other functions and it can be widely used in multi-layered architectures. It is one of the most widely used activation functions in deep neural networks.

2.4. Swish Activation Function

The swish activation function was developed by Google researchers, similar to the sigmoid activation function in Equations 2.7 and 2.8 [12]. The Swish activation function does not always have a single and continuous positive or negative derivative throughout the entire architecture. In addition, instead of taking only positive derivatives over all points, like the sigmoid function, it has negative derivatives over certain points. The developers have demonstrated that the swish function outperforms the commonly used ReLU activation function by testing it on challenging datasets. Figure 4 shows the normal and derivatized version of the swish activation function.

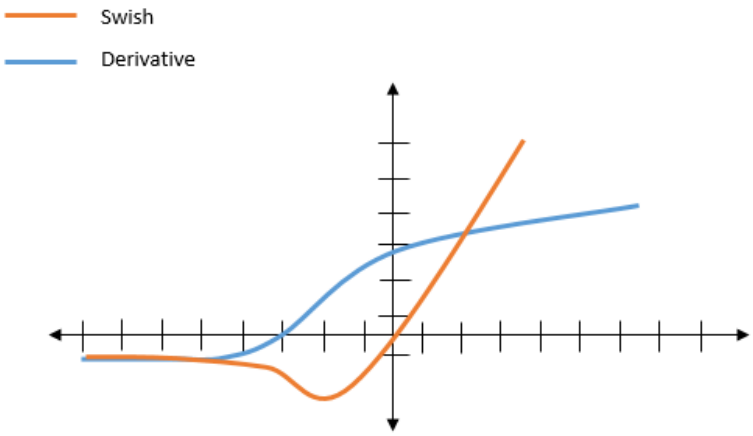


Figure 4. Swish activation function

$$f(x) = x\text{sigmoid}(x) = \frac{x}{1 + e^{-x}} \tag{2.7}$$

$$\frac{df(x)}{dx} = f(x) + \sigma(x)(1 - f(x)) \tag{2.8}$$

In Equation 2.8, the parameter σ represents the sigmoid function. Since the sigmoid function faces the vanishing gradient problem, the learning process is not at the desired level. Due to this problem, the sigmoid function has begun to lose its former popularity. In order to regain its former popularity, it defines the swish function with the help of the multiplication of the current inputs and the sigmoid activation function, as seen in Equation 2.7. In addition, it is seen that the swish function overcomes the negative region problem seen in the ReLU function, as in Figure 4 [12]. It is widely used mainly in image processing studies [19].

3. Parametric Activation Functions

Among the non-linear functions, the parameterized activation function is the functions that allow us to work actively on deep learning architectures thanks to the parameter value. Parametric activation functions used in the literature are expressed as LReLU, ELU, SELU, RSigELU [4, 8- 12].

3.1. LReLU Activation Function

The LReLU activation function was developed to cope with the negative region problem that occurs in the ReLU activation function [8]. The negative zone problem is that negative values are set to zero during processing. As a result of this adjustment, the learning process does not occur as a result of the death of some neurons due to the fact that negative outputs cannot be differentiated. The LReLU activation function, in other words, is an advanced variant of the ReLU activation function. In order to overcome the current problem, LReLU sets the activation function like Equations 3.1 and 3.2 instead of setting zero when the input value x is less than zero as in Equation 2.3. Figure 5 shows the normal and derivatized state of the LReLU activation function.

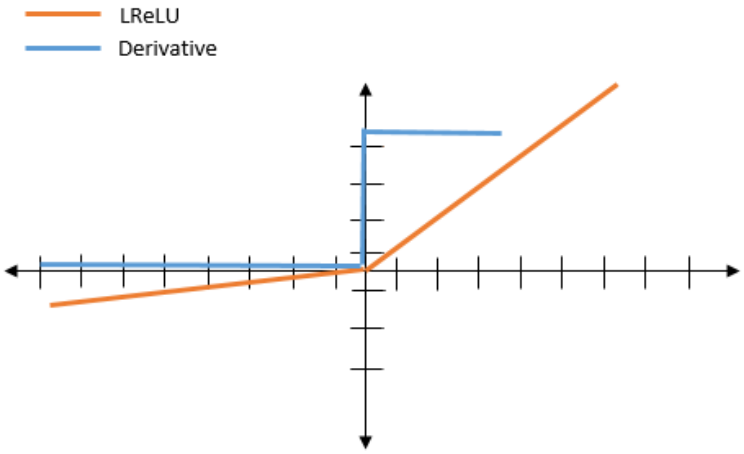


Figure 5. LReLU activation function

$$f(x) = \begin{cases} \alpha x, & x < 0 \text{ and } \alpha = 0.01 \\ x, & x \geq 0 \end{cases} \tag{3.1}$$

$$\frac{df(x)}{dx} = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases} \tag{3.2}$$

By adding a slope parameter α to Equation 3.1 instead of setting negative values to zero as in Equation 2.3, the LReLU activation function that can overcome the negative region problem has emerged. Thus, it is observed in the literature that it is widely used in deep learning studies as an alternative to the ReLU activation function.

3.2. ELU Activation Function

The ELU activation function was developed to cope with the negative region problem arising from the ReLU activation function and inspired by natural gradients [9]. The ELU activation function is adopts the positive region like the ReLU activation function, which effectively avoids the vanishing gradient problem. It is also used like Equations 3.3 and 3.4 as an exponential function in the negative part like the LReLU activation function. Thanks to the ELU activation function, neuron deaths are prevented

within the deep learning architecture. Thanks to this feature, it stands out with its faster convergence and continuous learning feature [19]. Figure 6 shows the normal state and the derivatized state of the ELU activation function.

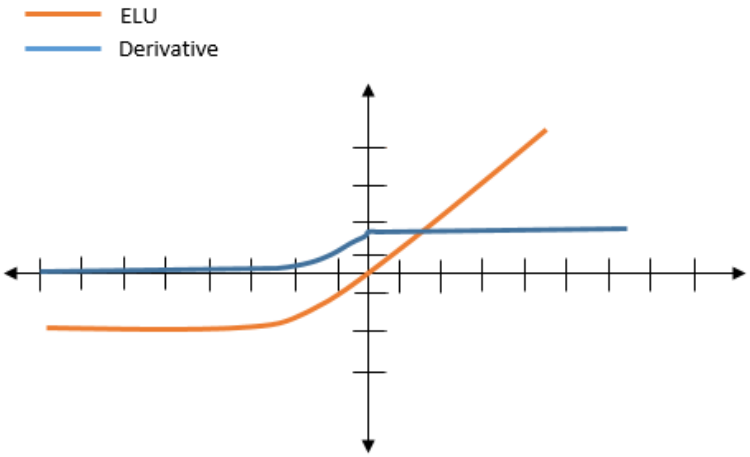


Figure 6. ELU activation function

$$f(x) = \begin{cases} x, & x > 0 \\ \alpha(e^x - 1), & x \leq 0 \end{cases} \tag{3.3}$$

$$\frac{df(x)}{dx} = \begin{cases} 1, & x > 0 \\ ae^x, & x \leq 0 \end{cases} \tag{3.4}$$

In summary, the ELU activation function works slower than the ReLU activation function due to the exponential expression [9]. Time is an important factor when convolutional neural network architectures are applied on real-time data. Therefore, it is used as an alternative to the ReLU activation function. ELU activation function gives better performance in classification and training speed for α=0.3 [20].

3.3. SELU Activation Function

The SELU activation function developed by Klambauer et al. and was developed as in Equations 3.5 and 3.6 in order to overcome the slow work in the ELU activation function [21]. In addition, the SELU activation function can overcome the negative region problem found in the ReLU activation function, which effectively avoids the vanishing gradient problem. In this way, SELU works actively in both positive and negative regions. Figure 7 shows the normal and derivatized state of the SELU activation function.

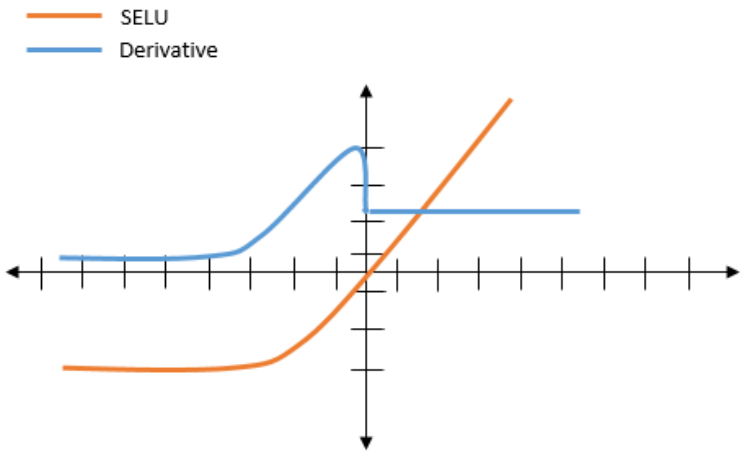


Figure 7. SELU activation function

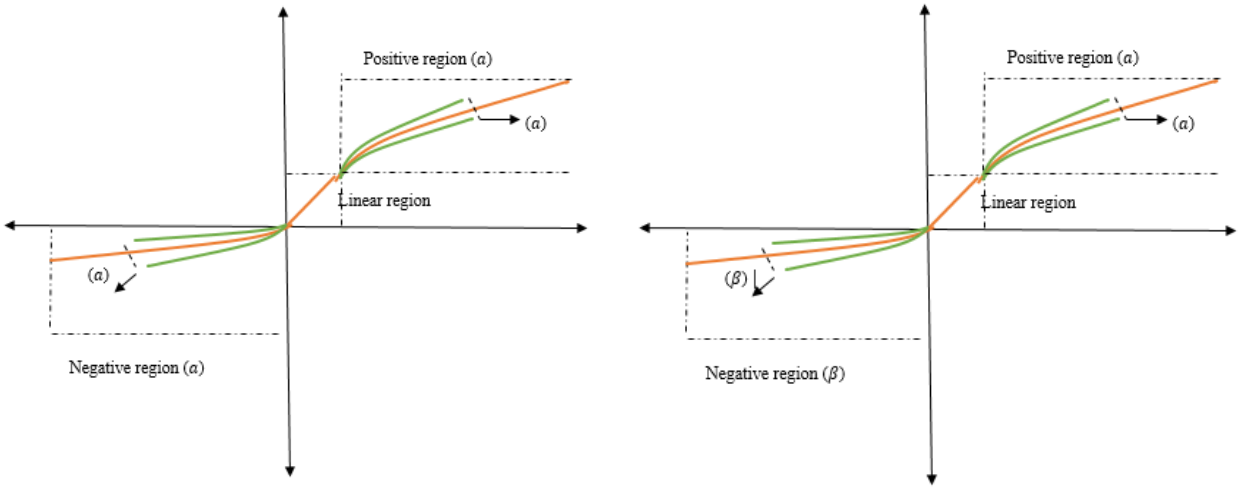
$$f(x) = \lambda \begin{cases} x, & x > 0 \\ \alpha(e^x - 1), & x \leq 0 \end{cases} \tag{3.5}$$

$$\frac{df(x)}{dx} = \begin{cases} \lambda, & x > 0 \\ f(x) + \lambda\alpha, & x \leq 0 \end{cases} \tag{3.6}$$

Equations 3.5 and 3.6 use two constant parameters α and λ. The best results with SELU are obtained with $\alpha = \sim 1.6732$ and $\lambda = \sim 1.0507$ values. When the equation is examined, when $x > 0$ value, ReLU behaves like an activation function. SELU can perform the learning process robustly due to its self-normalization feature, analytically zero mean and unit variance convergence, and allows it to be trained over many layers [22, 23].

3.4. RSigELU Activation Function

It was developed to deal with the problem of the vanishing gradient problem occurring in the sigmoid and tangent activation functions and the negative region problem occurring in the ReLU activation function [4]. The proposed activation function has proposed single and double parameter RSigELUS and RSigELUD activation functions to overcome the existing problems. It is ensured that the proposed activation function works actively in positive, negative and linear regions. Equations 3.7, 3.8, 3.9 and 3.10 of the activation functions of RSigELUS and RSigELUD are given. Figure 8 shows the behavior of the RSigELU activation function.



(a) (b)

Figure 8. Behavior of proposed RSigELUS (a) and RSigELUD (b) function

$$f(x) = \begin{cases} x * \left(\frac{1}{1+e^{-x}}\right) * a + x, & 1 < x \\ x, & 0 \leq x \leq 1 \\ a(e^x - 1), & x < 0 \end{cases} \tag{3.7}$$

$$f(x) = \begin{cases} x\left(\frac{1}{1+e^{-x}}\right)\alpha + x, & 1 < x \\ x, & 0 \leq x \leq 1 \\ \beta(e^x - 1), & x < 0 \end{cases} \tag{3.8}$$

$$\frac{df(x)}{dx} = \begin{cases} \frac{-\alpha x}{(e^x+1)^2} + \frac{\alpha x - \alpha}{(e^x+1)} + \alpha + 1, & 1 < x \\ 1, & 0 \leq x \leq 1 \\ \alpha e^x, & x < 0 \end{cases} \tag{3.9}$$

$$\frac{df(x)}{dx} = \begin{cases} \frac{-\alpha x}{(e^x+1)^2} + \frac{\alpha x - \alpha}{(e^x+1)} + \alpha + 1, & 1 < x \\ 1, & 0 \leq x \leq 1 \\ \beta e^x, & x < 0 \end{cases} \tag{3.10}$$

In Equations 3.7, 3.8, 3.9 and 3.10, α and β represent the defined slope coefficients. While the slope coefficient of a provides the control of the positive region, the control of the negative region is provided with β. As a result, the activation functions should have features that allow the network to converge easily and quickly. In addition, there should be no vanishing gradient problem in the designed activation functions, the outputs should be symmetrical to zero, it should be applicable after each layer and can be calculated millions of times in deep neural networks. When the equations were examined, they reported that the behaviour of the activation function was like ReLU when the parameter a was 0. Also, when a parameter is 0 and β value is 0.2, RSigELUD activation function behaves like ELU activation function [40-42]. When Equations 3.9 and 3.10 are examined, it is seen that RSigELUS and RSigELUD activation functions work on both negative and positive regions when derivatives are taken. Other activation functions defined in the literature are shown in Table 1.

Table 1. Activation Functions Used in the Literature

Activation Name	Year	State Function	Derivative State Function
Softplus [26]	2001	$f(x) = \ln(1 + e^x)$	$\frac{df(x)}{dx} = \frac{1}{1 + e^x}$
Softsing [24]	2009	$f(x) = \frac{x}{1 + \lvert x \rvert}$	$\frac{df(x)}{dx} = \frac{1}{(1 + \lvert x \rvert)^2}$
PReLU [28]	2015	$f(x) = \begin{cases} ax & x < 0 \\ x & x \geq 0 \end{cases}$	$\frac{df(x)}{dx} = \begin{cases} a & x < 0 \\ 1 & x \geq 0 \end{cases}$
DReLU [30]	2017	$f(x) = \begin{cases} \begin{cases} 0 & a \leq 0 \text{ and } b \leq 0 \\ a & a > 0 \text{ and } b \leq 0 \end{cases} \\ \begin{cases} -b & a \leq 0 \text{ and } b > 0 \\ a - b & a > 0 \text{ and } b > 0 \end{cases} \end{cases}$	$\frac{df(x)}{dx} = \begin{cases} \begin{cases} 0 & a \leq 0 \\ 1 & a > 0 \end{cases} \\ \begin{cases} 0 & b \leq 0 \\ -1 & b > 0 \end{cases} \end{cases}$
PELU [31]	2017	$f(x) = \begin{cases} \frac{a}{b}x & x \geq 0 \\ a\left(\exp\left(\frac{x}{b}\right) - 1\right) & x < 0 \end{cases}$	$\frac{df(x)}{dx} = \begin{cases} \frac{ax}{b^2} & x \geq 0 \\ -\frac{a}{b^2}\exp\left(\frac{x}{b}\right) & x < 0 \end{cases}$
CELU [32]	2017	$f(x) = \begin{cases} a\left(\exp\left(\frac{x}{a}\right) - 1\right) & x < 0 \\ x & x \geq 0 \end{cases}$	$\frac{df(x)}{dx} = \begin{cases} \exp\left(\frac{x}{a}\right) & x < 0 \\ 1 & x \geq 0 \end{cases}$
Hexpo [34]	2017	$(x) = \begin{cases} -a\left(e^{-\frac{x}{b}} - 1\right) & x \geq 0 \\ c\left(e^{\frac{x}{d}} - 1\right) & x < 0 \end{cases}$	$\frac{df(x)}{dx} = \begin{cases} \frac{a}{b}e^{-\frac{x}{b}} & x \geq 0 \\ \frac{c}{d}e^{\frac{x}{d}} & x < 0 \end{cases}$
SignReLu [25]	2018	$f(x) = \begin{cases} \alpha\frac{x}{1 + \lvert x \rvert} & x < 0 \\ x & x \geq 0 \end{cases}$	$\frac{df(x)}{dx} = \begin{cases} a\frac{1}{(1 + \lvert x \rvert)^2} & x < 0 \\ 1 & x \geq 0 \end{cases}$
LISA [27]	2019	$f(x) = \begin{cases} \alpha_1 x - \alpha_1 + 1 & 1 < x < \infty \\ x & 0 \leq x \leq 1 \\ \alpha_2 x & -\infty < x < 0 \end{cases}$	$\frac{df(x)}{dx} = \begin{cases} \alpha_1 & 1 < x < \infty \\ 1 & 0 \leq x \leq 1 \\ \alpha_2 & -\infty < x < 0 \end{cases}$
FELU [29]	2019	$f(x) = \begin{cases} a\left(2^{x/\ln(2)} - 1\right) & x < 0 \\ x & x \geq 0 \end{cases}$	$\frac{df(x)}{dx} = \begin{cases} a2^{x/\ln 2} & x < 0 \\ 1 & x \geq 0 \end{cases}$
Mish [33]	2019	$f(x) = x\,\text{tangent}(\ln(1 + e^x))$	$\frac{df(x)}{dx} = \frac{e^x\omega}{\delta^2}$
Logish [43]	2021	$f(x) = x\ln[1 + sigmoid(x)]$	$\frac{df(x)}{dx} = \ln\left(1 + \frac{1}{1 + e^{-x}}\right) + \frac{xe^{-x}}{(1 + e^{-x})(2 + e^{-x})}$
SAAF [44]	2021	$f(x) = \frac{x}{\frac{x}{a} + e^{-\frac{x}{\beta}}} \quad 0 < \frac{\beta}{a} < e$	$\frac{df(x)}{dx} = \frac{\left(1 + \frac{x}{\beta}\right)e^{-\frac{x}{\beta}}}{\left(\frac{x}{a} + e^{-\frac{x}{\beta}}\right)^2}$
Soft-Clipping Swish [45]	2021	$f(x) = \frac{1}{a}\log\left(\frac{1 + e^{ax}}{1 + e^{a(x-1)}}\right) \quad a > 0$	

Activation functions enable to work and learn better the deep learning models by revealing the hidden features of real-world problems. However, if deep learning models are run without choosing activation functions, the desired success cannot be achieved with limited learning. Therefore, the deep learning architectures used cause it to behave like linear regression. For this reason, non-linear activation functions are preferred in deep neural network architectures. The first is the fixed activation functions, the second is the trainable activation functions. Also, activation functions commonly evaluated according to error value, success rate, and confidence interval profile.

4. Conclusion

Since deep learning architectures work on complex problems, linear activation functions lose their competence. Because activation functions are the basis for learning the complex and continuous relationship between the variables and reaching the global optimum. In addition, there should be no vanishing gradient problem in the designed activation functions, and the outputs should be applicable after each layer in deep neural networks [35-42]. In this article, we have discussed, focusing on those that are the most common fixed and trainable activation functions presented in the literature. Non-linear activation functions are except for the sigmoid and hyperbolic tangent activation functions, the vanishing gradient problem is overcome and it is seen that deep neural networks are trained continuously. In the study, we report the best values obtained without paying attention to the different architectures or experimental setup used for different purposes in the literature.

Author Contributions

All authors contributed equally to this work. They all read and approved the final version of the manuscript.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] K. Adem, S. Kılıçarslan, O. Cömert, *Classification and diagnosis of cervical cancer with stacked autoencoder and softmax classification,* Expert Systems with Applications, 115, (2018) 557– 564.

[2] S. Kılıçarslan, K. Adem, M. Çelik, *Diagnosis and classification of cancer using hybrid model based on ReliefF and convolutional neural network,* medical hypotheses, 137, (2020) 199577.

[3] S. Kılıçarslan, M. Çelik, Ş. Sahin, *Hybrid models based on genetic algorithm and deep learning algorithms for nutritional Anemia disease classification*, *Biomedical Signal Processing and Control*, *63*, (2021) 102231.

[4] S. Kılıçarslan, M. Çelik, RSigELU: A nonlinear activation function for deep neural networks, Expert Systems with Applications, 174, (2021) 114805.

[5] A. Apicella, F. Donnarumma, F. Isgrò, R. Prevete, *A survey on modern trainable activation functions,* Neural Networks, 138 (2021) 14–32.

[6] S. Scardapane, S. Van Vaerenbergh, S. Totaro, A. Uncini, *Kafnets: Kernel-based non-parametric activation functions for neural networks*, Neural Networks, 110, (2019) 19–32.

[7] V. Nair, G. E. Hinton, *Rectified linear units improve restricted Boltzmann machines*, In Proceedings of the 27th international conference on machine learning (ICML-10), 2010, pp. 807–814.

[8] A. L. Maas, A. Y. Hannun, A. Y. Nug, *Rectifier nonlinearities improve neural network acoustic models*, S. Dasgupta, D. McAllester (Eds.), International Conference on Machine Learning Workshop on Deep Learning for Audio, Speech, and Language Processing, Atlanta, USA, 2013, pp. 1–6.

[9] D. A. Clevert, T. Unterthiner, S. Hochreiter, *Fast and accurate deep network learning by exponential linear units (elus)*, arXiv preprint arXiv:1511.07289, (2015).

[10] L. Trottier, P. Gigu, B. Chaib-draa, *Parametric exponential linear unit for deep convolutional neural networks,* in: X. Chen, B. Luo, F. Luo, V. Palade, M. A. Wani (Eds.), 16th IEEE International Conference on Machine Learning and Applications, Cancun, Mexico, 2017, pp. 207–214.

[11] P. Ramachandran, B. Zoph, Q. V. Le, *Searching for activation functions*, arXiv preprint arXiv:1710.05941, (2017).

[12] K. I. Funahashi, *On the approximate realization of continuous mappings by neural networks*, Neural Networks, 2, (1989) 183–192.

[13] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, Mathematics of Control, Signals and Systems, 2, (1989) 303–314.

[14] Y. Bengio, P. Simard, P. Frasconi, *Learning long-term dependencies with gradient descent is difficult,* IEEE Transactions on Neural Networks, 5, (1994) 157–166.

[15] S. Hochreiter, S. Jurgen, *Long Short-Term Memory*, Neural Computation, 9, (1997) 1735–1780.

[16] A. Benjemmaa, I. Klabi, M. S. Masmoudi, J. el Ouni, M. Masmoudi, *Implementations approaches of neural networks lane following system,* O. Faten, B. A. Faouzi (Eds.), in: 16th IEEE Mediterranean Electrotechnical Conference, Yasmine Hammamet, Tunisia, 2012, pp. 515–518.

[17] M. Goyal, R. Goyal, P. Reddy, B. Lall, *Activation Functions*, In Deep Learning: Algorithms and Applications, Springer, Cham, 2020.

[18] N. Jinsakul, C. F. Tsai, C. E. Tsai, P. Wu, *Enhancement of deep learning in image classification performance using exception with the swish activation function for colorectal polyp preliminary screening,* Mathematics, 7, (2019), 1170.

[19] B. Ding, H. Qian, J. Zhou, *Activation functions and their characteristics in deep neural networks*, F. Wang, G. H. Yang (Eds.), in: Chinese Control and Decision Conference, Shenyang, China, 2018, pp. 1836–1841.

[20] D. J. Rumala, E. M. Yuniarno, R. F. Rachmadi, S. M. S. Nugroho, I. K. E. Purnama, *Activation functions evaluation to improve performance of convolutional neural network in brain disease classification based on magnetic resonance images,* S. M. S. Nugroho (Ed.), in: 2020 International Conference on Computer Engineering, Network, and Intelligent Multimedia, Surabaya, Indonesia, 2020, pp. 402–407.

[21] G. Klambauer, T. Unterthiner, A. Mayr, S. Hochreiter, *Self-normalizing neural networks*, U. V. Luxburg, I. Guyon, S. Bengio, H. Wallach, R. Fergus (Eds.), in: Advances in Neural Information Processing Systems, Long Beach, CA, USA, 2017, pp. 971–980.

[22] T. Yang, Y. Wei, Z. Tu, H. Zeng, P. Ren, *Design space exploration of neural network activation function circuits,* IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 38(10), (2018) 1974–1978.

[23] D. Pedamonti, *Comparison of non-linear activation functions for deep neural networks on MNIST classification task*, arXiv preprint arXiv:1804.02763, (2018).

[24] J. Bergstra, G. Desjardins, P. Lamblin, Y. Bengio, Quadratic polynomials learn better image features, Technical Report, (2009), 1337.

[25] G. Lin, W. Shen, *Research on convolutional neural network based on improved Relu piecewise activation function,* Procedia Computer Science, 131, (2018) 977–984.

[26] C. Dugas, Y. Bengio, F. Belisle, C. Nadeau, R. Garcia, *Incorporating second-order functional knowledge for better option pricing,* Advances in Neural Information Processing Systems, 20, (2001) 472–478.

[27] V. S. Bawa, V. Kumar, *Linearized sigmoidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability,* Expert Systems with Applications, 120, (2019), 346–356.

[28] K. He, X. Zhang, S. Ren, J. Sun, *Delving deep into rectifiers: Surpassing human-level performance on imagenet classification*, R. Bajcsy, G. Hager, Y. Ma (Eds.) in: IEEE International Conference on Computer Vision, Santiago, Chile, 2015, pp. 1026–1034.

[29] Z. Qiumei, T. Dan, W. Fenghua, *Improved convolutional neural network based on fast exponentially linear unit activation function*, IEEE Access, 7, (2019) 151359–151367.

[30] F. Godin, J. Degrave, J. Dambre, W. De Neve, *Dual Rectified Linear Units (DReLUs): a replacement for tangent activation functions in quasi-recurrent neural networks*, Pattern Recognition Letters, 116, (2018), 8–14.

[31] L. Trottier, P. Giguere, B. Chaib-Draa, *Parametric exponential linear unit for deep convolutional neural networks,* In 2017 16th IEEE International Conference on Machine Learning and Applications, 2017, pp. 207–214.

[32] J. T. Barron*, Continuously differentiable exponential linear units,* arXiv preprint arXiv:1704.07483, 2017.

[33] D. Misra, *Mish: A self-regularized non-monotonic activation function*. arXiv preprint arXiv:1908.08681, 2019.

[34] S. Kong, M. Takatsuka, *Hexpo: A vanishing-proof activation function,* Y. Choe (Ed.) in: International Joint Conference on Neural Networks, Anchorage, AK, USA, 201, pp. 2562–2567.

[35] A. L. Hodgkin, A. F. Huxley, *A quantitative description of membrane current and its application to conduction and excitation in nerve,* The Journal of Physiology, 117(4), (1952) 500–544.

[36] N. Jinsakul, C. F. Tsai, C. E. Tsai, P. Wu, *Enhancement of deep learning in image classification performance using xception with the swish activation function for colorectal polyp preliminary screening,* Mathematics, 7(12), (2019) 1170.

[37] H. Ma, Y. Liu, Y. Ren, J. Yu, *Detection of collapsed buildings in post-earthquake remote sensing images based on the improved YOLOv3,* Remote Sensing, 12(1), (2020) 44.

[38] M. A. Bülbül, C. Öztürk, *Optimization, modeling and implementation of plant water consumption control using genetic algorithm and artificial neural network in a hybrid structure*, Arabian Journal for Science and Engineering, (2021) 1–15.

[39] I. Pacal, D. Karaboğa, *A Robust Real-Time Deep Learning Based Automatic Polyp Detection System*, Computers in Biology and Medicine, 134, (2021) 104519.

[40] S. Memiş, S. Enginoğlu, U. Erkan, *A classification method in machine learning based on soft decision-making via fuzzy parameterized fuzzy soft matrices,* Soft Computing (2021). https://doi.org/10.1007/s00500-021-06553-z

[41] S. Memiş, S. Enginoğlu, U. Erkan, *Numerical Data Classification via Distance-Based Similarity Measures of Fuzzy Parameterized Fuzzy Soft Matrices*, IEEE Access, 9, (2021) 88583–88601.

[42] U. Erkan, *A precise and stable machine learning algorithm: Eigenvalue classification (EigenClass),* Neural Computing & Applications, 33, (2021), 5381–5392.

[43] H. Zhu, H. Zeng, J. Liu, X. Zhang, Logish: *A new nonlinear nonmonotonic activation function for convolutional neural network,* Neurocomputing, 458, (2021), 490–499.

[44] Y. Zhou, D. Li, S. Huo, S. Y. Kung, *Shape autotuning activation function*, Expert Systems with Applications, 171, (2021) 114534.

[45] M. A. Mercioni, S. Holban, *Soft-Clipping Swish: A Novel Activation Function for Deep Learning*, L. Kovács, R. E. Precup (Eds.), in: IEEE 15th International Symposium on Applied Computational Intelligence and Informatics, Timisoara, Romania, 2021, pp. 225–230.