Contents lists available at ScienceDirect

AI Open

journal homepage: www.keaipublishing.com/en/journals/ai-open

Full length article

Enhancing neural network classification using fractional-order activation functions

Meshach Kumar[a,*], Utkal Mehta[a], Giansalvo Cirrincione[b]

[a] *Discipline of Electrical and Electronic Engineering, The University of the South Pacific, Laucala Campus, Fiji*
[b] *Lab. LTI, University of Picardie Jules Verne, Amiens, France*

ARTICLE INFO

Keywords:
Fractional calculus
Neural networks
Classification
Multilayer perceptron
Activation functions
Accuracy

ABSTRACT

In this paper, a series of novel activation functions is presented, which is derived using the improved Riemann–Liouville conformable fractional derivative (^{RL}CFD). This study investigates the use of fractional activation functions in Multilayer Perceptron (MLP) models and their impact on the performance of classification tasks, verified using the IRIS, MNIST and FMNIST datasets. Fractional activation functions introduce a non-integer power exponent, allowing for improved capturing of complex patterns and representations. The experiment compares MLP models employing fractional activation functions, such as fractional sigmoid, hyperbolic tangent and rectified linear units, against traditional models using standard activation functions, their improved versions and existing fractional functions. The numerical studies have confirmed the theoretical observations mentioned in the paper. The findings highlight the potential usage of new functions as a valuable tool in deep learning in classification. The study suggests incorporating fractional activation functions in MLP architectures can lead to superior accuracy and robustness.

1. Introduction

Machine learning has become a cornerstone for addressing intricate problems across many domains. One of the primary tasks within this field is classification, a process that categorizes input data (Zhang, 2000). The significance of neural networks (NNs) in modern data-driven scenarios cannot be overstated, requiring continuous investigation for broader and more efficient applications. Different methods, such as binary, multi-class, and hierarchical classification, can serve many purposes. Due to their capability to understand complex functions and relationships between inputs and outputs, NNs are a predominant tool for classification. It has diverse applications ranging from finance and marketing, medical diagnosis, speech recognition and image recognition (Yıldız et al., 2019).

Several NN architectures cater to different classification tasks. These include feed-forward neural networks (FNNs), recurrent neural networks (RNNs), suitable for sequential data like speech and text (Gill and Khehra, 2022), and convolutional neural networks (CNNs), optimized for grid data such as images (Chen and Shi, 2021). While these networks offer versatility, there are still several challenges that need to be overcome. NNs black-box nature raises interpretability concerns (Yıldız et al., 2019). Furthermore, NNs demand large datasets, involve resource-intensive training, and are prone to overfitting, which can degrade their predictive accuracy (Chen and Shi, 2021). Moving on, fractional calculus extends the conventional understanding of derivatives and integrals by considering non-integer orders (Wang et al., 2017). Its application in areas like physics and engineering has been transformative, and recently, its integration into neural networks has gained traction (Zou et al., 2014). By bringing the unique properties of fractional calculus into play, NNs can better account for long-range dependencies inherent in many real-world datasets, boosting performance in various tasks (Zhang et al., 2017; Viera-Martin et al., 2022; Yu et al., 2012; Aguilar et al., 2020). The understanding and implementation of fractional calculus have become critical for solving increasingly complex and challenging problems in various applications, and it holds significant potential for future research and innovation.

Furthermore, activation functions play a crucial role in analyzing the stability of NNs and greatly influence their performance in interpreting physical phenomena. It evaluates the importance of neuron inputs and decides if a neuron should be activated based on mathematical operations (Karlik and Olgac, 2011). Research on fractional activation functions for neural networks is an emerging field. In 2018, Ivanov (2018) introduced a method to develop fractional activation functions by extending functions like sine, cosine, and the logistic function using the ML function principle. Experiments were conducted to assess the effects of these activation functions on learning and prediction accuracy in NNs. Zamora Esquivel et al. (2019) later proposed

[*] Corresponding author.
E-mail addresses: meshachkumar19@gmail.com (M. Kumar), utkal.mehta@usp.ac.fj (U. Mehta), nimzoexin59@gmail.com (G. Cirrincione).

https://doi.org/10.1016/j.aiopen.2023.12.003
Received 20 July 2023; Received in revised form 25 October 2023; Accepted 20 December 2023
Available online 27 December 2023

a unique methodology that allowed NNs to autonomously optimize the activation functions using concepts from fractional calculus. This method, when applied to a ResNet18 architecture, yielded superior accuracy compared to a ResNet100 on the CIFAR10 dataset. In a subsequent study, Zamora et al. (2022) presented Fractional Adaptive Linear Units (FALUs), which merges principles from fractional calculus to create a wide array of activation functions, including Sigmoid, Gaussian, ReLU, and more. FALUs are advantageous because it introduces only a few additional trainable parameters without the need for special optimization or initialization techniques. It presents an automated, efficient solution to the problem of optimizing activation functions and have shown promising results against traditional and modern methods.

Additionally, Job et al. (2022) studied the fractional-order versions of the rectified linear unit activation function and its numerous variants, both linear and nonlinear. These variants were designed using methods like the Maclaurin and Taylor expansion series. To assess the efficiency, a simulation was carried out using MLP models to predict power from a Texas wind turbine. The performance was gauged by tweaking the activation function in both hidden and output layers. In another study (Altan et al., 2020), the paper delved into the conformable derivative concept, which in the realm of NNs, has notable benefits. Using this derivative method, researchers effectively employed the sigmoid activation function. Similarly, Solís-Pérez et al. (2022) introduced a novel activation function for NNs, combining the hyperbolic tangent and Khalil conformable exponential function. This new function showcased adaptability and was tested in three scenarios: estimating the Nusselt number in a helical double-pipe evaporator, finding the volumetric mass transfer coefficient in an electro-chemical reaction, and gauging the thermal efficiency of a solar parabolic trough collector. Model performance metrics revealed correlations of 99%, 97%, and 95% with experimental data in each scenario, respectively. Notably, the network achieved satisfactory learning with fewer neurons in the hidden layer and effectively trained on limited experimental data. This function was termed conformable activation function (CAF). Another study by Altan et al. (2023) combined the sigmoid activation function with a fractional derivative approach, using the proportional Caputo definition. The aim was to reduce backpropagation convergence errors and boost generalization. The study's results confirmed that this combination surpassed traditional derivative models in classification accuracy for neural networks.

Fractional activation functions have gained attention in the field of neural networks and machine learning due to their potential benefits. These functions can enhance accuracy by leveraging their unique properties, leading to improved predictions and overall performance. Furthermore, they contribute to better generalization capabilities, allowing models to handle unseen data effectively and reduce overfitting issues. This increased reliability makes fractional activation functions valuable for real-world applications. These activation functions offer enhanced flexibility in model configurations for various tasks. One more important benefit is achieving higher accuracy while using fewer parameters. This enables the creation of smaller and more efficient models without compromising performance, which is crucial in resource-limited computational environments. On the other hand, various limitations are associated with using existing fractional activation functions. These functions can introduce computational complexity, leading to increased computational costs compared to conventional activation functions (Ivanov, 2018; Zamora Esquivel et al., 2019; Zamora et al., 2022; Altan et al., 2023). Implementing these complex functions can also pose challenges due to design issues (Altan et al., 2020). Moreover, the output not following a zero-centric nature leads to poor convergence (Solís-Pérez et al., 2022). Additionally, some of these fractional activation functions may not perform well when used in the outer layers of neural networks (Job et al., 2022). Addressing these challenges requires a comprehensive evaluation and improvement of such fractional activation functions within the field of study.

Finally, the primary objective of this paper is to propose innovative fractional activation functions that not only leverage the benefits offered by existing functions but also overcome their limitations. Considering the challenges associated with current fractional activation functions, such as computational complexities and feasibility concerns, this paper aims to introduce a simpler and more efficient version. Through meticulous analysis of the advantages and drawbacks of established functions, this paper strives to develop a new class of fractional activation functions that maximize performance, ease of implementation, and achieve faster convergence.

This paper is divided into the following sections: Section 2 presents a brief overview of various activation functions, Section 3 presents the proposed methodology, Section 4 discusses the mathematical preliminaries, Section 5 presents the proposed activation functions and their characteristics, Section 6 summarizes the dataset and network architecture used. A comparison of the proposed activation function with other functions is shown in Section 7, followed by the conclusions in Section 8 and an appendix.

2. Activation functions in deep learning: A brief overview

The advancement of activation functions in neural networks presents an intriguing development. These functions serve a critical role in determining the output of neurons, and throughout the course of time, several activation functions have been suggested to enhance training efficiency and optimize model performance. A linear function serves as a basic activation function, producing an output of $c \times x$ for an input x, where c is a constant. While it does not introduce non-linearity, neural networks require non-linearity to be effective. Without it, multiple layers in a network still produce a linear output. Since real-world data is typically not linear, non-linear layers transform data in a more complex feature space, enhancing the network's capabilities. This section provides an overview of the evolution of activation functions for deep learning.

Logistic sigmoid/tanh Unit Based Activation Functions: To introduce non-linearity into neural networks, early approaches utilized the logistic sigmoid and tanh activation functions. The logistic sigmoid is a widely used traditional non-linear function that restricts outputs to the range of [0, 1]. However, this function exhibits saturation for extreme inputs, leading to a vanishing gradient problem. Another commonly employed activation function is the tanh, which shares similarities with the logistic sigmoid while possessing zero centricity. With its output in the range of [−1, 1], this squashing function also experiences challenges related to vanishing gradients.

Rectified Linear Unit Based Activation Functions: The limitations of the previously mentioned activation functions, such as logistic sigmoid and tanh, include saturated output and increased complexity. The Rectified Linear Unit (ReLU) (Nair and Hinton, 2010) is considered the best activation function due to its simplicity and improved performance. In fact, it was utilized in the renowned AlexNet model (Krizhevsky et al., 2017). Researchers have also explored different variations of ReLU to address its drawbacks, including limited non-linearity, unbounded output, and failure to utilize negative values.

Exponential Unit Based Activation Functions: One major issue with the rectified linear unit activation function is the inadequate utilization of negative values, which can result in a vanishing gradient. To address these limitations, researchers have explored exponential function-based activation functions. The exponential linear unit (ELU) (Clevert et al., 2015) incorporates negative values by employing the exponential function. Various variants of ELU have been proposed in academic literature as alternative activation functions (Dubey et al., 2022).

Learning/Adaptive Activation Functions: Many commonly used activation functions such as sigmoid, tanh, ReLU, and ELU are designed manually and may not fully utilize the complexity of the data. An emerging trend in this field is the use of learning-based adaptive activation functions. These types of activation functions include adjustable

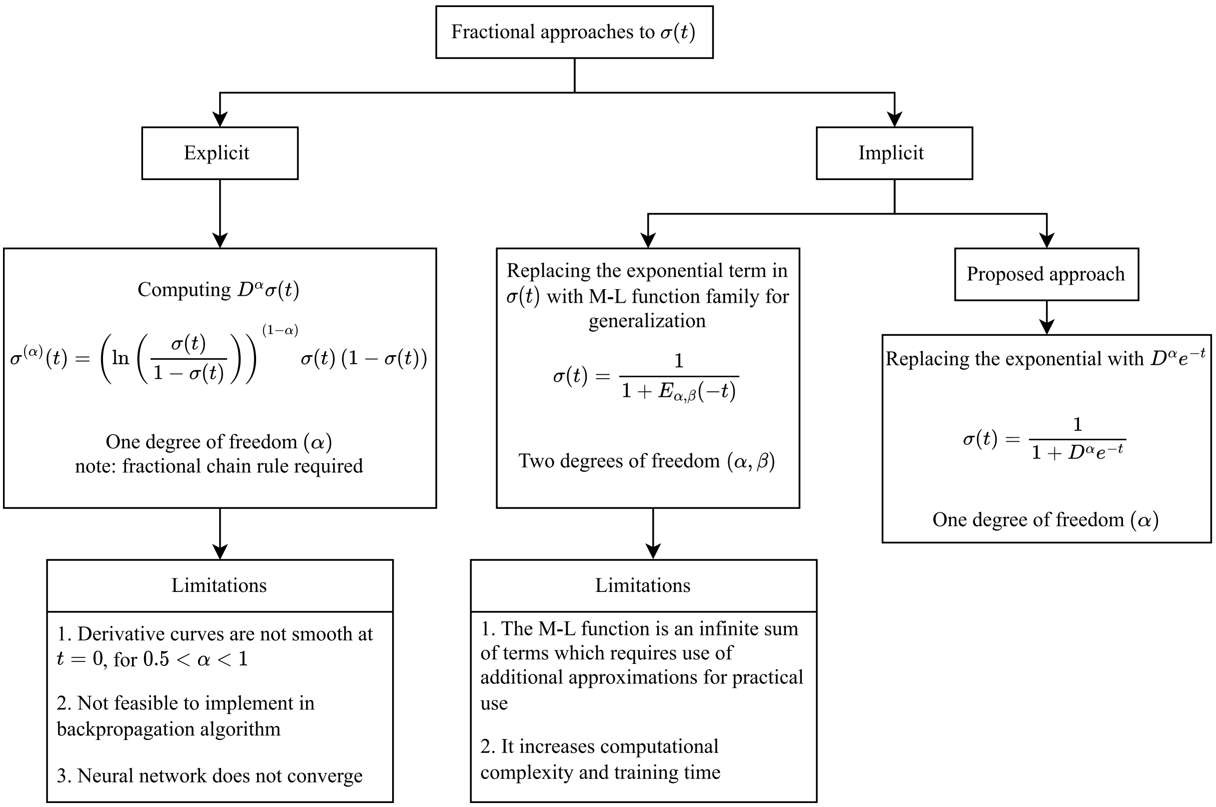

Fractional approaches to $\sigma(t)$
Explicit
Implicit
Computing $D^{\alpha}\sigma(t)$
$\sigma^{(\alpha)}(t) = \left(\ln\left(\frac{\sigma(t)}{1-\sigma(t)}\right)\right)^{(1-\alpha)} \sigma(t)\,(1-\sigma(t))$
One degree of freedom (α)
note: fractional chain rule required
Replacing the exponential term in $\sigma(t)$ with M-L function family for generalization
$\sigma(t) = \frac{1}{1+E_{\alpha,\beta}(-t)}$
Two degrees of freedom (α, β)
Proposed approach
Replacing the exponential with $D^{\alpha}e^{-t}$
$\sigma(t) = \frac{1}{1+D^{\alpha}e^{-t}}$
One degree of freedom (α)
Limitations
1. Derivative curves are not smooth at $t = 0$, for $0.5 < \alpha < 1$
2. Not feasible to implement in backpropagation algorithm
3. Neural network does not converge
Limitations
1. The M-L function is an infinite sum of terms which requires use of additional approximations for practical use
2. It increases computational complexity and training time

Fig. 1. Formulation of the proposed approach.

parameters that can be learned during training. For instance, Adaptive Piecewise Linear (APL) (Agostinelli et al., 2014) and Swish (Ramachandran et al., 2017) are examples of adaptive activation functions with two and one learnable parameters respectively.

Miscellaneous Activation Functions: In recent years, many other activation functions have also been investigated. These activation include softplus units (Sun et al., 2020), probabilistic functions (Su et al., 2017), polynomial functions (Li et al., 2019), and kernel functions (Scardapane et al., 2019).

3. Proposed methodology

There are two methods for incorporating fractional calculus into activation functions: explicit and implicit. In the explicit method, the activation function remains unchanged during the forward pass, but in the backward pass, instead of taking its first derivative, we take its fractional derivative (Job et al., 2022; Altan et al., 2020, 2023). Conversely, in the implicit method, we replace or generalize the exponential function with a family of functions that possess similar properties to the exponential. This results in a new function that incorporates fractional calculus properties within the existing activation function (Ivanov, 2018; Solís-Pérez et al., 2022).

The proposed method deviates from the classical approach by substituting a family of functions (M-L) for the exponential in the sigmoid function. From this point of view, the M-L family is replaced with the proposed fractional derivative of the exponential. It must be highlighted that the resulting fractional sigmoid activation function is not the fractional derivative of $\sigma(t)$ (no chain rule is required as in Altan et al., 2020, 2023), but a variant of the M-L approach as shown in Fig. 1. The objective of the proposed approach is to create a versatile activation function that can be seamlessly integrated into various neural models. Additionally, it aims to achieve quicker convergence compared to current fractional activation functions and reduce training time. Lastly, it strives for superior performance in comparison to existing activation functions. This approach aims to address these existing limitations while also building upon and enhancing the advantages offered by current activation functions. The main contributions of this paper are as follows:

1. Proposal of new fractional activation functions that leverage the benefits of existing functions while addressing their limitations.
2. The paper recognizes the computational difficulties posed by current fractional activation functions and proposes a more computationally efficient and straightforward version for implementation.
3. The paper also emphasizes the development of fractional activation functions that not only enhance performance but also achieve faster convergence.
4. The paper provides a meticulous analysis validating the theoretical claims using MLP and three benchmark datasets.

4. Mathematical preliminaries

This section presents all the necessary mathematical definitions, theorems, and proofs used to derive the new fractional-order activation functions. Fractional calculus is a branch of mathematics that studies fractional or non-integer order derivatives and integrals. Various techniques and methods are involved in fractional calculus, such as the Grunwald–Letnikov (G-L) fractional difference operator, the Riemann–Liouville (R-L) fractional integral and derivative, and the Caputo fractional derivative (Mehta et al., 2022). However, this paper uses a new definition of fractional calculus known as CFD. The form of the definition shows that it is the most natural definition and the most fruitful one. The definition for $0 \leq \alpha < 1$ coincides with the classical definitions on polynomials (up to a constant). Further, if $\alpha = 1$, the definition coincides with the classical definition of the first derivative (Khalil et al., 2014).

4.1. The definition

Given a function $f : [0,\infty) \to \mathbb{R}$, the CFD of f of order α is defined by:

$$T_\alpha f(t) = \lim_{\varepsilon\to 0} \frac{f(t+\varepsilon t^{1-\alpha}) - f(t)}{\varepsilon} \tag{1}$$

for all $t > 0, \alpha \in (0,1)$. In addition, if it exists at any order α, then it is simply said f is α-differentiable. If f is α-differentiable $\forall\, t \in (0,a), a > 0$ and $\lim_{t\to 0^+} f^{(\alpha)}(t)$ exists, then define

$$f^{(\alpha)}(0) = \lim_{t\to 0^+} f^{(\alpha)}(t) \tag{2}$$

Both $f^{(\alpha)}(t)$ and $T_\alpha f(t)$ denote the CFD of f with order α.

Furthermore, this definition coincides with the classical definitions of R–L and of Caputo on polynomials (up to a constant multiple).

As a consequence of the above definition, the following useful theorems are obtained from the given analysis.

Theorem 2.1.1. *If a function $f : [0,\infty) \to \mathbb{R}$ is α-differentiable at $t_0 > 0, \alpha \in (0,1]$ then f is continuous at t_0, which implies that $\lim_{h\to 0} f(t_0+h) = f(t_0)$. Hence, α is continuous at t_0.*

Finally, T_α satisfies all the properties in the following theorem.

Theorem 2.1.2. *Let $\alpha \in (0,1]$ and f, g be α-differentiable at a point $t > 0$. Then,*

1. $T_\alpha(af + bg) = aT_\alpha(f) + bT_\alpha(g)$, *for all* $a, b \in \mathbb{R}$.
2. $T_\alpha(t^p) = pt^{p-\alpha}$, *for all* $p \in \mathbb{R}$.
3. $T_\alpha(fg) = fT_\alpha(g) + gT_\alpha(f)$.
4. $T_\alpha(\frac{f}{g}) = \frac{gT_\alpha(f) - fT_\alpha(g)}{g^2}$.
5. $T_\alpha(\lambda) = 0$, *for all constants.*
6. *If, in addition, f is differentiable, then* $T_\alpha f(t) = t^{1-\alpha}\frac{df}{dt}(t)$.

For $\alpha = 1$, the above properties correspond to the integer order derivative.

4.2. Improved conformable fractional derivatives

4.2.1. Preliminaries

Given the R-L and Caputo definitions, for $\alpha \in (n-1,n]$, $n \in \mathbb{N}$, (Khalil et al., 2014)

$$^{RL}D_a^\alpha f(t) = \frac{1}{\Gamma(n-\alpha)}\frac{d^n}{dt^n}\int_a^t \frac{f(x)}{(t-x)^{\alpha-n+1}}dx \tag{3}$$

$$^{C}D_a^\alpha f(t) = \frac{1}{\Gamma(n-\alpha)}\int_a^t \frac{f^{(n)}(x)}{(t-x)^{\alpha-n+1}}dx \tag{4}$$

Considering the R-L and Caputo fractional derivative definitions (Kothari et al., 2019), for $\alpha \in (n-1,n]$, $n \in \mathbb{N}$, it follows

$$\lim_{\alpha\to(n-1)} {}_a^{RL}D^\alpha f(t) = f^{n-1}(t) \tag{5}$$

$$\lim_{\alpha\to(n-1)} {}_a^{C}D^\alpha f(t) = f^{(n-1)}(t) - f^{(n-1)}(a) \tag{6}$$

where $f^{(0)}(t) = f(t)$.

Definition 2.2.1. Given a function $f : R \to R$ the improved Caputo-type CFD of f of order α is defined by Gao and Chi (2020):

$${}_a^{C}T_\alpha(t) = \lim_{\varepsilon\to 0}\left[(1-\alpha)(f(t)-f(a)) + \alpha\frac{f(t+\varepsilon(t-a)^{1-\alpha}) - f(t)}{\varepsilon}\right] \tag{7}$$

where $0 \le a < t < +\infty$, a is a given number.

Definition 2.2.2. Given a function $f : R \to R$, the improved Riemann–Liouville-type conformable fractional derivative of f of order α is defined by Gao and Chi (2020):

$${}_a^{RL}T_\alpha(t) = \lim_{\varepsilon\to 0}\left[(1-\alpha)f(t) + \alpha\frac{f(t+\varepsilon(t-a)^{1-\alpha}) - f(t)}{\varepsilon}\right] \tag{8}$$

where $0 \le a < t < +\infty$ and a is a given number.

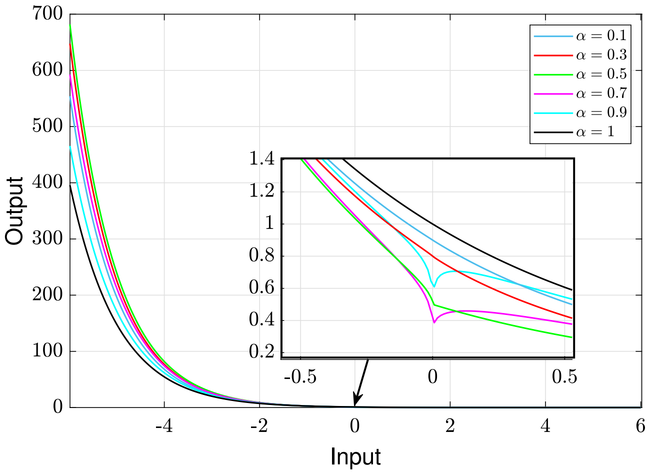


Fig. 2. Fractional derivative of e^{-t} of order α.

Upon simplifying using (5) and (6), one can see

$$\lim_{\alpha\to 0} {}_a^{C}T_\alpha(t) = \lim_{\alpha\to 0}(f(t) - f(a)) = \lim_{\alpha\to 0} {}_a^{C}D^\alpha(t) \tag{9}$$

$$\lim_{\alpha\to 0} {}_a^{RL}T_\alpha(t) = \lim_{\alpha\to 0} f(t) = \lim_{\alpha\to 0} {}_a^{RL}D^\alpha(t) \tag{10}$$

If $\alpha = 1$, both ${}_a^{C}T_\alpha(t)$ and ${}_a^{RL}T_\alpha(t)$ coincide with $\frac{d}{dt}f(t) = f'(t)$.

Remark. (8) was used to derive the proposed functions instead of (7). The term $(f(t) - f(a))$ in (7) causes the functions to lose their symmetry at $t = 0$.

5. Derivation and analysis of fractional-order activation functions

Fractional-order activation functions generalize classical activation functions by including fractional exponents in their equations. In this section, the derivation of the new functions is described mathematically, along with an exploration of their properties at various fractional orders.

5.1. Generalization of exponential function

The RL-type conformable fractional derivative is defined in (8). Given $f(t) = e^{-t}$ it follows (see Appendix A)

$${}_0^{RL}T_\alpha(t) = (1-\alpha)(e^{-t}) - \alpha t^{1-\alpha}e^{-t} \tag{11}$$

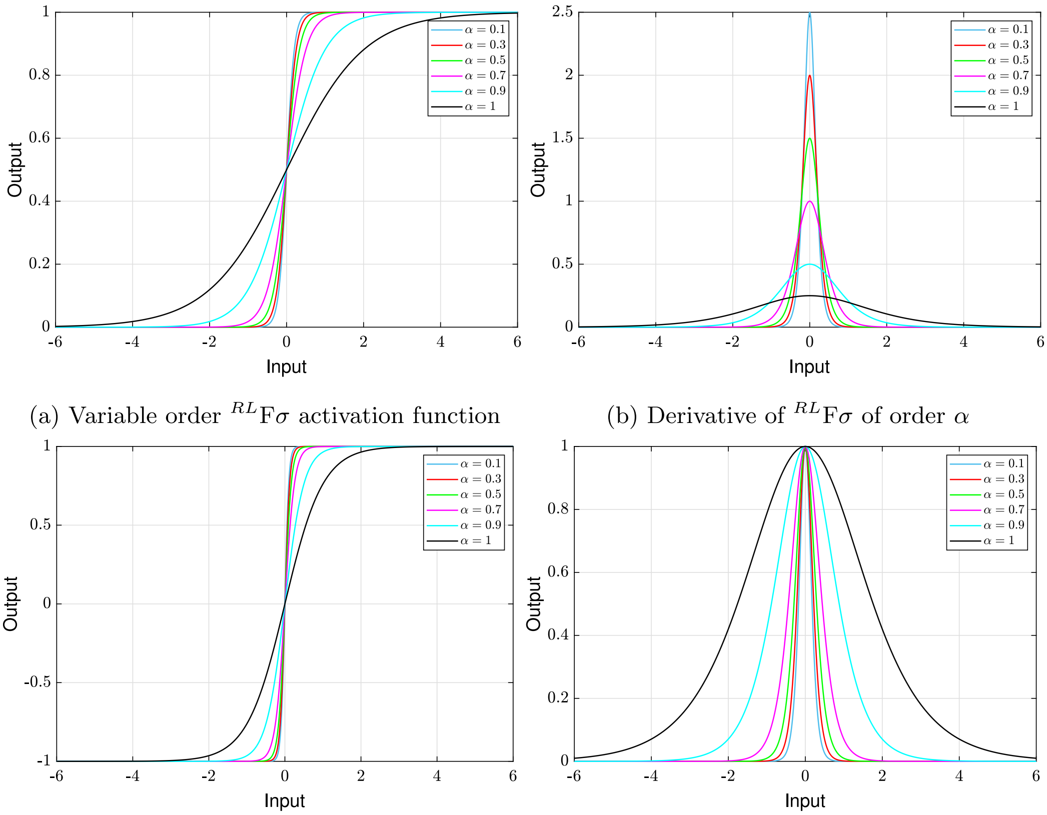
Fig. 2 shows the generalized exponential function of order α, ranging in (0,1]. It can be observed that while changing the α values the y-axis crossing of the generalized exponential term also changes, thus enabling the slope of the exponential function to change accordingly.

5.2. Proposed activation functions

Consider first the classical sigmoid function as

$$\sigma(t) = \frac{1}{1+e^{-t}} \tag{12}$$

To obtain the generalized sigmoid function, the exponential term e^{-t} in (12) is replaced with (11), yielding the new fractional sigmoid activation function. The fractional sigmoid activation function can benefit



(a) Variable order ${}^{RL}\mathrm{F}\sigma$ activation function

(b) Derivative of ${}^{RL}\mathrm{F}\sigma$ of order α

(c) Variable order ${}^{RL}\mathrm{Ftanh}$ activation function

(d) Derivative of ${}^{RL}\mathrm{Ftanh}$ of order α

Fig. 3. Characteristics of new fractional activation functions.

Table 1
Proposed fractional sigmoid and hyperbolic tangent functions and derivatives.

Type	Function	Derivative
${}^{RL}\mathrm{F}\sigma(t)$	$\frac{1}{1+{}_0^{RL}T_\alpha(t)}$	${}^{RL}\mathrm{F}\sigma(t)(1-{}^{RL}\mathrm{F}\sigma(t))$
${}^{RL}\mathrm{Ftanh}(t)$	$2({}^{RL}\mathrm{F}\sigma(2t))-1$	$1-({}^{RL}\mathrm{Ftanh}(t))^2$

Table 2
Fractional rectified linear units.

Type	Function	Derivative
${}^{RL}\mathrm{F}ReLU$	$\text{output} = \begin{cases} t, & t \geqslant 0 \\ 0, & t < 0 \end{cases}$	$\text{output} = \begin{cases} t^{1-\alpha}, & t \geqslant 0 \\ 0, & t < 0 \end{cases}$
${}^{RL}\mathrm{F}LReLU$	$\text{output} = \begin{cases} t, & t \geqslant 0 \\ 0.01t, & t < 0 \end{cases}$	$\text{output} = \begin{cases} t^{1-\alpha}, & t \geqslant 0 \\ 0.01t^{1-\alpha}, & t < 0 \end{cases}$

from this modification, as it increases its generalization, flexibility, and memory efficiency without altering the shape of the original sigmoid function. Saturation can be reached more quickly since the slope of the fractional sigmoid function is controlled by the changes depicted with different α values. The model efficiency may be enhanced as a result of the increased freedom in shaping and regulating the behavior of the function.

Using (11), one can obtain

$$ {}^{RL}\mathrm{F}\sigma(t) = \frac{1}{1+{}_0^{RL}T_\alpha(t)} \tag{13}$$

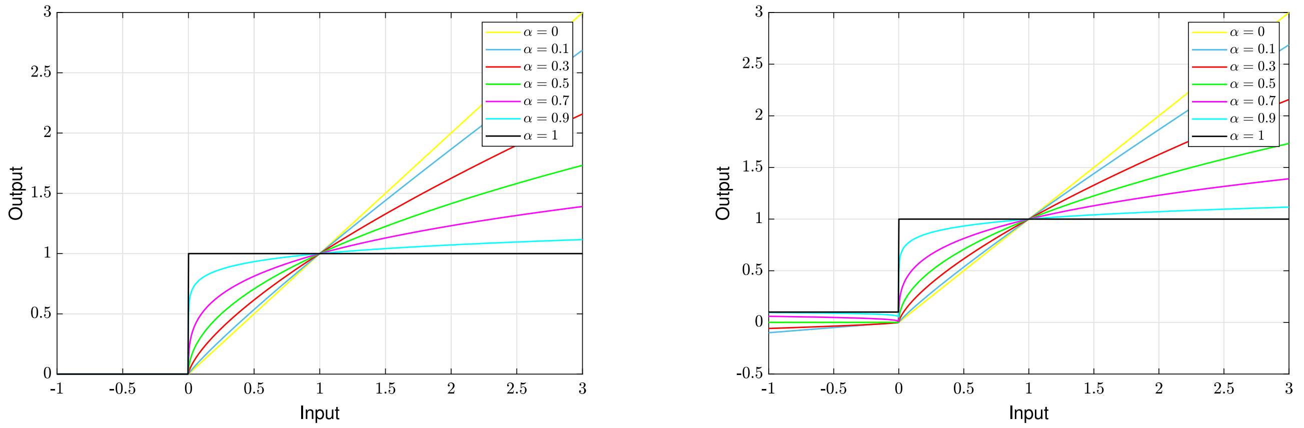
Similarly ${}^{RL}\mathrm{Ftanh}(t)$ can be expressed in terms of (13) as below (see Appendix B)

$$ {}^{RL}\mathrm{Ftanh}(t) = 2({}^{RL}\mathrm{F}\sigma(2t)) - 1. \tag{14}$$

Table 1 portrays the equations of the proposed activation functions and its derivatives. The characteristics of each of the proposed functions are presented in Fig. 3 together with its variable order $\alpha \in (0,1]$ derivatives. Fig. 3(a) shows the characteristics of the ${}^{RL}\mathrm{F}\sigma(t)$ with different values of α. It can be observed, that for smaller values of α, the slope gradually becomes steeper, and for $\alpha = 1$, the traditional sigmoid activation function is achieved. Likewise, the derivative of ${}^{RL}\mathrm{F}\sigma(t)$ in Fig. 3(b) shows that the amplitude of the curve is determined by the value of α. The behavior of the derivative of the sigmoid function as t approaches positive and negative infinity is that it approaches zero in both cases. This means that the slope of the ${}^{RL}\mathrm{F}\sigma(t)$ becomes very small at large positive and negative values of t, indicating that the function becomes almost horizontal. Similarly, Fig. 3(c) shows the characteristics of the ${}^{RL}\mathrm{Ftanh}(t)$ with varying α. It can be observed that for smaller values of α, the slope gradually becomes steeper, and for $\alpha = 1$, the traditional tanh activation function is achieved. The derivative of ${}^{RL}\mathrm{Ftanh}(t)$ in Fig. 3(d) shows that the steepness of the curve is determined by the value of α. The graph has an inflection point at $t = 0$, where it changes sign from positive to negative. The graph of the derivative starts at one at $t = 0$, decreases to zero as t approaches $+\infty$, and likewise as t approaches $-\infty$.

Table 2 presents the proposed fractional rectified linear units. Unlike the implicit approach, these functions have used the explicit approach to incorporate fractional calculus. Therefore, (1) can be directly applied to the classical rectified linear units. During forward propagation, the activation function remains unchanged. However, during backpropagation, instead of computing the first derivative, the derivative has been calculated using $\alpha \in (0,1]$.

Fig. 4 shows the characteristics of the ^{RL}FReLU(left) and ^{RL}FLReLU(right). It can be observed that for $\alpha = 0$, the classical functions are obtained, which are used for forward propagation and



(a) Variable order ^{RL}FReLU (b) Variable order ^{RL}FLReLU

Fig. 4. Characteristics of new fractional rectified linear units.

Table 3
Comparison of activation functions.

Activation function	Diminishing gradients	Better generalization	Lack of adaptability	Computational efficiency	Faster convergence
$^{RL}F\sigma$	No	Yes	No	No	Yes
^{RL}Ftanh	No	Yes	No	No	Yes
$^{RL}FReLU$	No	Yes	No	Yes	Yes
$^{RL}FLReLU$	No	Yes	No	Yes	Yes
σ (Dubey et al., 2022)	Yes	No	No	No	No
tanh (Dubey et al., 2022)	Yes	No	Yes	No	No
ReLU (Dubey et al., 2022)	Yes	No	Yes	Yes	No
LReLU (Dubey et al., 2022)	Partial	No	No	Yes	No
Swish (Ramachandran et al., 2017)	No	No	No	No	No
LiHST (Roy et al., 2022)	No	No	No	No	No
ABReLU (Dubey and Chakraborty, 2021)	No	No	No	Yes	No
$^{PC}\sigma$ (Altan et al., 2023)	No	Yes	Yes	No	No
Ψ^{ζ} (Solís-Pérez et al., 2022)	No	Yes	Yes	No	No
FReLU (Job et al., 2022)	No	Yes	No	No	No
FLReLU (Job et al., 2022)	No	Yes	No	No	No

Table 4
Properties of proposed activation functions.

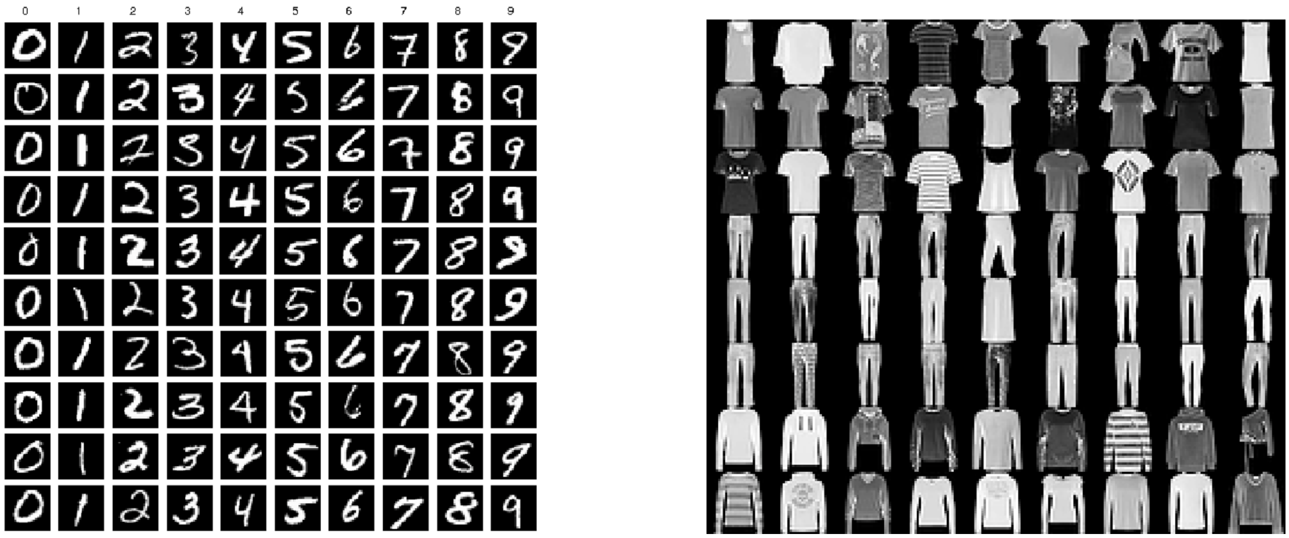
Type	Parametric	Monotonic	Smooth	Differentiable	Bounded
$^{RL}F\sigma$	Yes	Yes	Yes	Yes	Yes [0,1]
^{RL}Ftanh	Yes	Yes	Yes	Yes	Yes [−1,1]
$^{RL}FReLU$	Yes	Yes	No	$t > 0$	For negative values
$^{RL}FLReLU$	Yes	Yes	No	$-t < 0 < t$	No

for $0 < \alpha < 1$ depicts the fractional derivatives, and finally, for $\alpha = 1$, the first derivative is obtained.

Table 3 provides a concise overview of different activation functions commonly used in neural networks giving insight into their key characteristics. These characteristics include the presence of diminishing gradients, the ability to promote better generalization, adaptability to different network configurations, computational efficiency, and the speed of convergence during training. Similarly, Table 4 highlights the properties of proposed fractional activation functions. All functions are parametric, meaning they can be adjusted based on certain parameters, potentially offering more flexibility in modeling. They are monotonic, smooth, and differentiable, which are desirable properties for activation functions as they ensure a smooth gradient flow, aiding in the optimization process. The tables provide a clear comparison and properties of standard and fractional activation functions, highlighting the advantages and limitations of each.

6. Dataset description and neural model design

For experimental analysis, three well-known and frequently employed datasets are used to assess the performance and capabilities of the proposed activation functions. These datasets include the Iris dataset, the Fashion-MNIST dataset, and the MNIST dataset. Employing these diverse datasets will showcase the versatility and applicability of the proposed activation functions while highlighting their performance across various domains and problem types. Moving on, the Iris dataset is a well-known dataset in machine learning and statistics. It includes measurements of four attributes (sepal length, sepal width, petal length, and petal width) from three types of iris flowers (setosa, versicolor, and virginica), with 50 samples for each species, totaling 150 data points. It is often used as a starting point for those learning classification techniques. The primary goal is to accurately classify iris flowers into their respective species, it is widely employed for learning and implementing classification algorithms. Fashion-MNIST is another widely used dataset in the field of machine learning and computer vision.



(a) Samples of MNIST dataset.

(b) Samples of FMNIST dataset.

Fig. 5. Dataset samples for training and testing.

It serves as a benchmark for image classification tasks and is often considered a modern replacement for the traditional MNIST dataset. Fashion-MNIST consists of 60,000 training images and 10,000 testing images, each of which is a grayscale image (28 × 28 pixels) of a fashion item or accessory, such as shoes, shirts, dresses, or bags. These images are categorized into ten classes, making it a multi-class classification problem. Finally, the MNIST dataset, short for the Modified National Institute of Standards and Technology database, is a widely recognized and commonly used dataset in the field of machine learning and computer vision. It consists of a collection of 28 × 28 pixel grayscale images of handwritten digits (0 through 9), along with their corresponding labels indicating the digit they represent. The dataset is divided into two main parts: a training set containing 60,000 images and a testing set containing 10,000 images. MNIST is often used as a benchmark for developing and testing various machine learning and deep learning algorithms, especially those related to image classification and digit recognition. Fig. 5 portrays the data samples of MNIST and FMNIST datasets.

6.1. Data preprocessing

Preprocessing was a vital step when working with image datasets like MNIST and Fashion MNIST. It involved transforming and cleaning the data to make it compatible with MLP. This process typically included normalizing pixel values to a range between 0 and 1 to enhance training efficiency, flattening 2D image arrays (28 × 28) into 1D vectors (1 × 784) to accommodate the model's input requirements, and optionally applying data augmentation techniques to increase training data diversity. For these datasets, label encoding was typically not needed as labels were already in the correct format. By implementing these preprocessing steps, the data was prepared in a manner that was conducive to training MLP models, ultimately improving their performance and accuracy.

6.2. MLP architecture

In this classification example, the MLP neural network is comprised of one input layer, one or more hidden layers, and one output layer. The MLP is highly flexible as it supports different types of activation functions, network architectures and optimization strategies, allowing practitioners to tailor their models to suit their needs. Fig. 6 shows the MLP architecture and Table 5 presents the model hyper-parameters used to train and test for the given three separate datasets. The model hyper-parameters are chosen based on thoroughly reviewing literature which uses MLP for MNIST classification (Karlik and Olgac, 2011; Alcantara, 2017; Pedamonti, 2018; Lau and Lim, 2018). Similar hyper-parameter values were used in pre-experiment examples and the best values were chosen based on experimenting baseline values. As for selecting the hyper-parameter fractional order, currently, each α is chosen between 0 and 1, and the network is evaluated. Finally, the α value at which the highest accuracy is achieved is used for all activation functions. Fig. 7 shows a sample of accuracies achieved at different values of α for different datasets.

Table 5
Network hyper-parameters for training and validation.

	IRIS	MNIST	FMNIST
Loss	$-\sum_{i=1}^{N}\sum_{j=1}^{c} y_j \log(\sigma_j)$	$-\sum_{i=1}^{N}\sum_{j=1}^{c} y_j \log(\sigma_j)$	$-\sum_{i=1}^{N}\sum_{j=1}^{c} y_j \log(\sigma_j)$
Hidden layers	$L = 1$	$L = 2$	$L = 2$
Neurons	$n = 32$ per layer	$n = 100$ per layer	$n = 100$ per layer
Learning rate	$\eta = 0.01$	$\eta = 0.1$	$\eta = 0.5$
Batch size	$m = 10$	$m = 50$	$m = 100$
Fractional order	$\alpha = 0.1$	$\alpha = 0.9$	$\alpha = 0.8$

7. Experimental evaluation

This section presents a comparative analysis of using MLP neural networks for classification. The proposed activation functions are compared with their classical counterparts, improved versions of the classical functions, which are the most recent and existing fractional activation functions. For each experiment, only the hidden layer activation functions are changed, and the output layer activation function (softmax) remains the same throughout. The evaluation is performed on the IRIS, MNIST and FMNIST datasets. The proposed analysis is focused on two key metrics: loss and accuracy. The performance of the MLP models is assessed in terms of their maximum accuracy. The effectiveness of the proposed functions is also assessed in terms of their influence on convergence rate, adaptability, and training time required to achieve maximum accuracy. The experiments have been conducted on an Anaconda platform, utilizing the Jupyter Notebook environment,

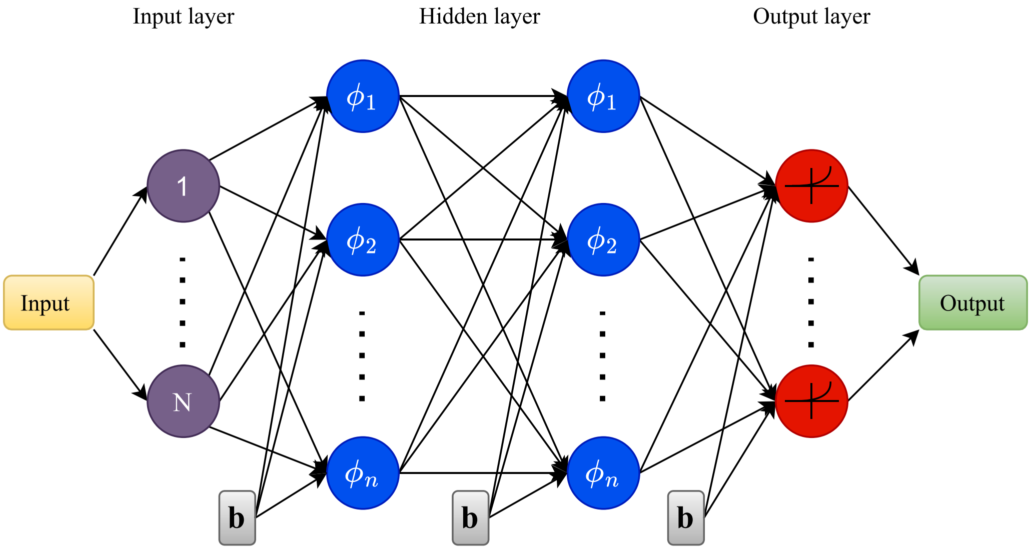


Fig. 6. MLP architecture.

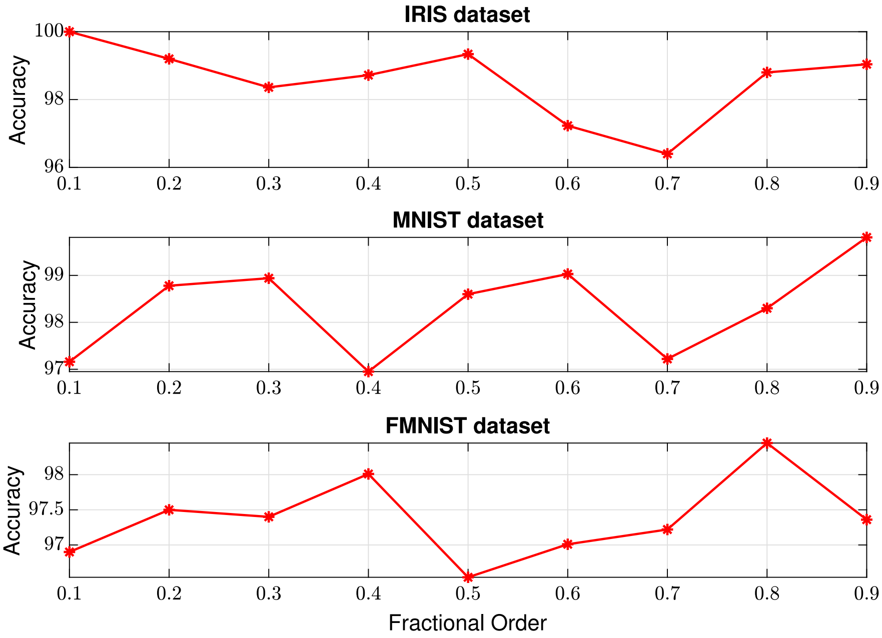


Fig. 7. Fractional order selection for each training dataset.

with a computer system featuring an i7 8th generation 4.0 GHz CPU and 16 GB RAM.

Table 6 summarizes all the training and testing accuracies achieved by each of the functions for the three datasets. Values in bold represent the maximum accuracies of the proposed functions, which are greater than all other functions. After performing these evaluations, some key findings observed are that all proposed functions achieve maximum accuracy in fewer epochs than compared to other functions. The proposed functions are trained for 70 epochs for the IRIS dataset, whereas other functions are trained for 100 epochs. For the MNIST dataset, the proposed functions are trained for 25 epochs and other functions are trained for 100 epochs. Finally, for the FMNIST dataset, the proposed functions are trained for 100 epochs, and other functions are trained for 200 epochs. This shows the superiority of the proposed functions as they can achieve higher accuracies even though they are trained for fewer epochs. Another important observation made was based on existing fractional activation functions, namely ${}^{PC}\sigma$ and Ψ^{ζ}. These functions are not feasible to implement in more than one layer network as it tends to suffer from overflow in gradient values, and the network stops learning. To obtain results for comparative analysis, models with ${}^{PC}\sigma$ and Ψ^{ζ} are trained with one hidden layer only but with 500 neurons. In general, the suggested functions retained the benefits supplied by existing functions. They offered additional benefits, such as superior performance in all circumstances and more generalization capabilities than all other functions. Compared to existing fractional activation functions, the suggested functions were more computationally efficient, and adaptive and eliminated the implementation limitation in deep architectures.

Figs. 8–19 present the comparative convergence curves of all functions in terms of loss and accuracy. Key observations made from these graphs are that for all the evaluations the proposed functions are far superior in terms of convergence rate. It can be seen that for all cases, the models with proposed functions has not only a faster convergence but also achieves higher accuracies right from epoch 1. When compared to different dataset evaluations, for IRIS the training convergence is reduced by 30% when using proposed functions, for MNIST the training

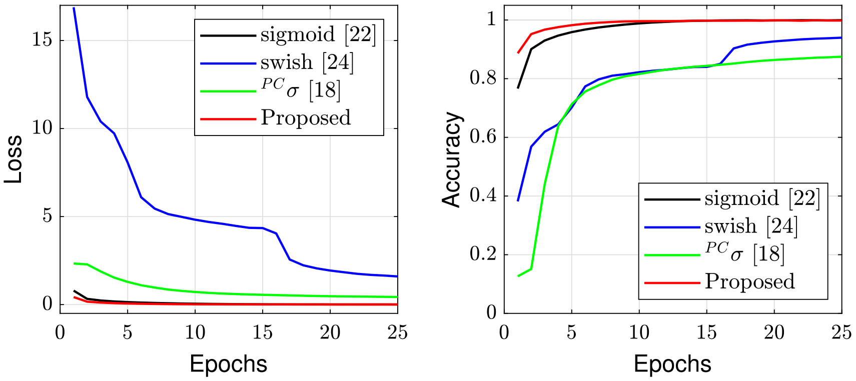


Fig. 8. Convergence curve of logistic sigmoid based activation functions for MNIST.

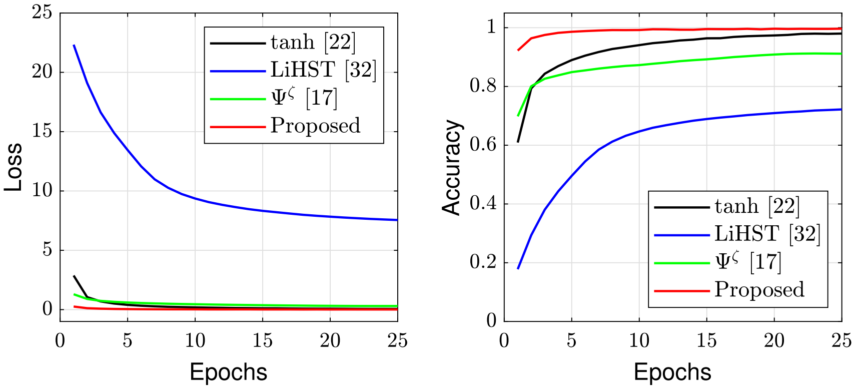


Fig. 9. Convergence curve of tanh based activation functions for MNIST.

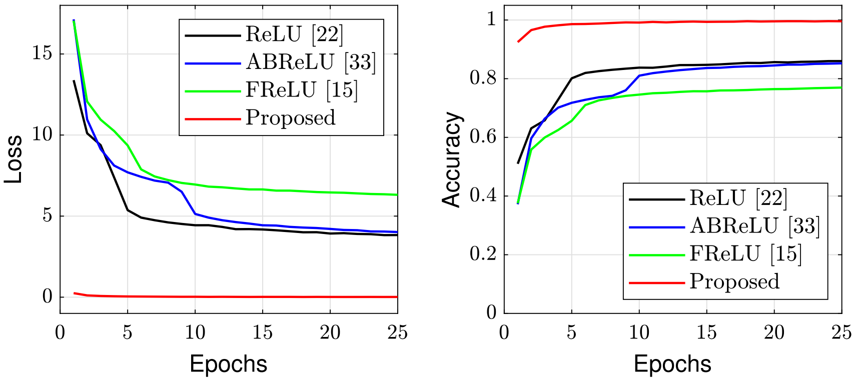


Fig. 10. Convergence curve of rectified linear unit based activation functions for MNIST.

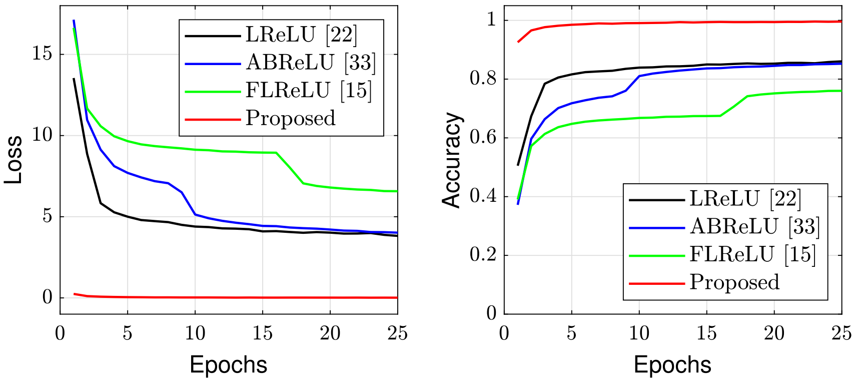


Fig. 11. Convergence curve of rectified linear unit based activation functions for MNIST.

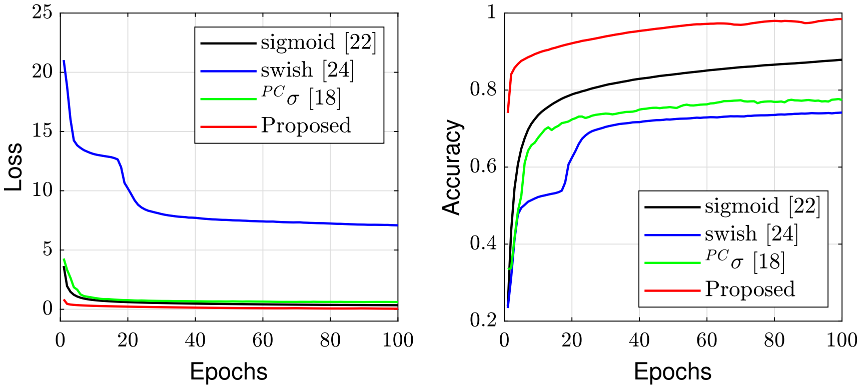


Fig. 12. Convergence curve of logistic sigmoid based activation functions for FMNIST.

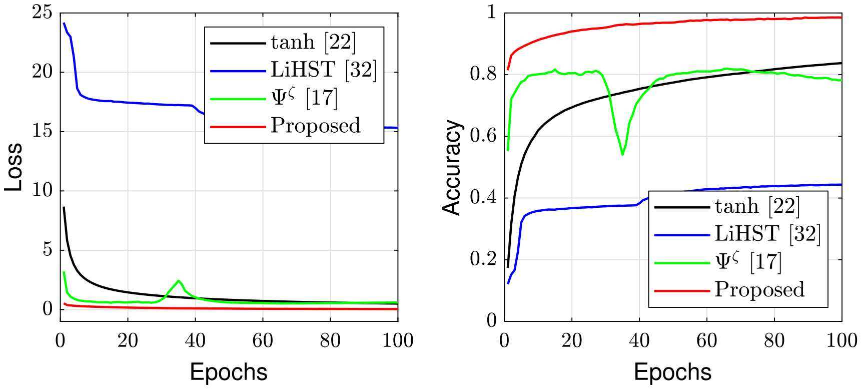


Fig. 13. Convergence curve of tanh based activation functions for FMNIST.

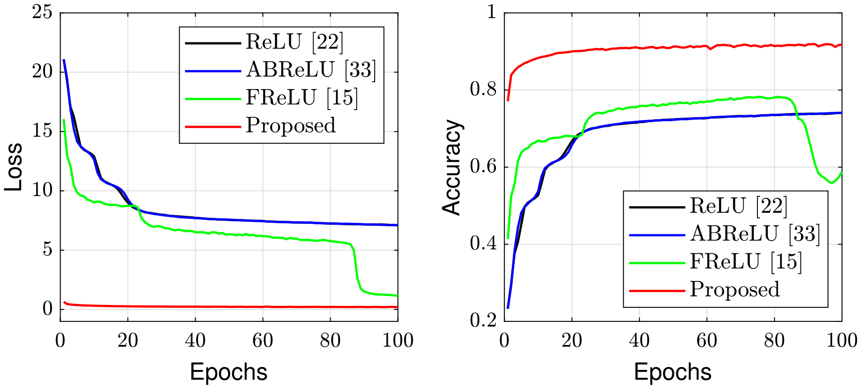


Fig. 14. Convergence curve of rectified linear unit based activation functions for FMNIST.

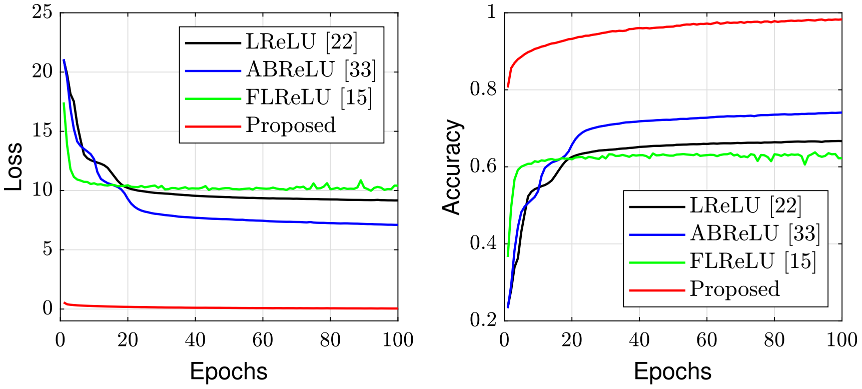


Fig. 15. Convergence curve of rectified linear unit based activation functions for FMNIST.

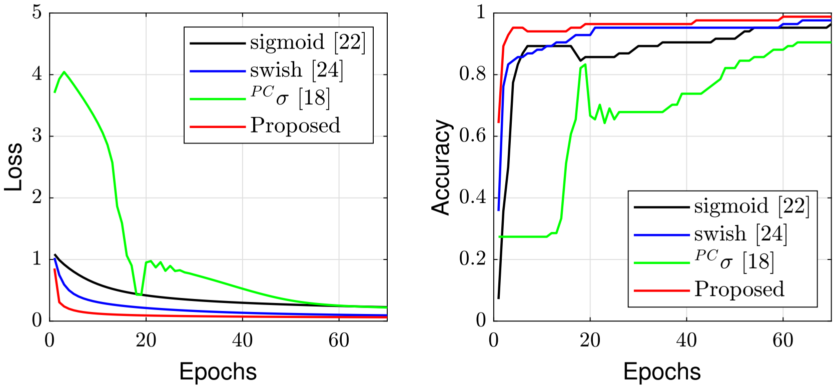


Fig. 16. Convergence curve of logistic sigmoid based activation functions for IRIS.

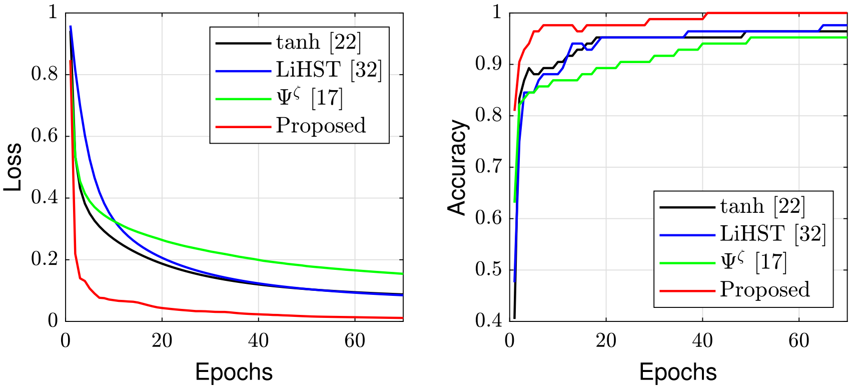


Fig. 17. Convergence curve of tanh based activation functions for IRIS.

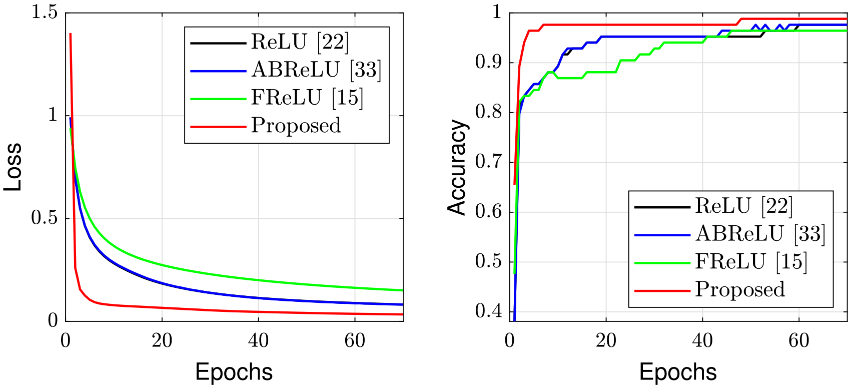


Fig. 18. Convergence curve of rectified linear unit based activation functions for IRIS.

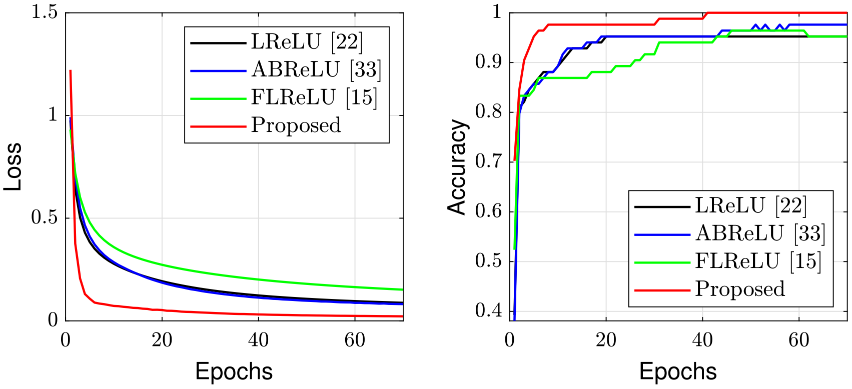


Fig. 19. Convergence curve of rectified linear unit based activation functions for IRIS.

Table 6
Performance comparisons on IRIS, MNIST and Fashion-MNIST datasets.

Activation function	IRIS		MNIST		Fashion-MNIST	
	Training	Testing	Training	Testing	Training	Testing
^{RL}Fσ	**98.81%**	**100.00%**	**99.81%**	**97.53%**	**98.45%**	**96.04%**
sigmoid (Dubey et al., 2022)	96.43%	95.56%	99.32%	95.82%	91.18%	86.03%
Swish (Ramachandran et al., 2017)	97.33%	97.75%	97.99%	96.25%	79.42%	76.23%
$^{PC}\sigma$ (Altan et al., 2023)	91.67%	91.11%	91.92%	92.16%	77.42%	75.87%
^{RL}Ftanh	**100.00%**	**97.78%**	**99.66%**	**97.23%**	**98.43%**	**95.20%**
tanh (Dubey et al., 2022)	96.32%	95.24%	99.57%	94.95%	89.01%	77.92%
LiSHT (Roy et al., 2022)	97.59%	96.88%	85.51%	83.38%	45.42%	43.85%
Ψ^{ζ} (Solís-Pérez et al., 2022)	95.25%	90.48%	90.26%	90.37%	78.24%	78.35%
^{RL}F$ReLU$	**100.00%**	**97.65%**	**99.58%**	**97.53%**	**91.75%**	**87.27%**
ReLU (Dubey et al., 2022)	98.57%	95.78%	88.05%	86.71%	78.85%	76.64%
ABReLU (Dubey and Chakraborty, 2021)	98.11%	97.01%	86.24%	84.77%	79.58%	76.67%
FReLU (Job et al., 2022)	95.14%	95.33%	96.59%	95.70%	80.21%	77.16%
^{RL}F$LReLU$	**100.00%**	**97.62%**	**99.59%**	**97.50%**	**98.30%**	**96.22%**
LReLU (Dubey et al., 2022)	97.62%	95.84%	88.52%	86.86%	67.49%	66.07%
FLReLU (Job et al., 2022)	95.39%	94.92%	86.93%	85.74%	62.75%	59.98%

Table 7
Training time to achieve maximum accuracy.

Activation function	Dataset	
	MNIST	FMNIST
^{RL}Fσ	**00:02:21**	**00:07:24**
sigmoid (Dubey et al., 2022)	00:07:02	00:13:16
Swish (Ramachandran et al., 2017)	00:06:46	00:13:28
$^{PC}\sigma$ (Altan et al., 2023)	00:027:17	00:26:37
^{RL}Ftanh	**00:02:11**	**00:06:15**
tanh (Dubey et al., 2022)	00:07:56	00:12:14
LiSHT (Roy et al., 2022)	00:07:33	00:11:51
Ψ^{ζ} (Solís-Pérez et al., 2022)	00:025:46	00:24:48
^{RL}F$ReLU$	**00:02:09**	**00:06:01**
ReLU (Dubey et al., 2022)	00:08:07	00:11:56
ABReLU (Dubey and Chakraborty, 2021)	00:06:16	00:12:34
FReLU (Job et al., 2022)	00:06:50	00:11:06
^{RL}F$LReLU$	**00:01:58**	**00:05:25**
LReLU (Dubey et al., 2022)	00:07:23	00:12:03
FLReLU (Job et al., 2022)	00:07:03	00:11:30

convergence is reduced by 75% when using proposed functions and for FMNIST the training convergence is reduced by 50% when using proposed functions. Also, it can be seen that the proposed functions obtain much lower loss values right from epoch 1. The functions which suffered the most were the improved versions of the classical functions namely the swish, LiHST, ABReLU and the fractional activation functions which suffered were the FReLU and FLReLU. Overall, the proposed functions outperformed all other functions in terms of faster convergence rate, better generalization and efficiently finding the shortest path to global minimum while training which all the other functions lack in evaluation.

Table 7 presents the training time to reach the highest accuracy utilizing all activation functions for MNIST and FMNIST. The values were not recorded for the IRIS dataset due to being below 20 s training time. The proposed functions, in all cases, reduce training time by a significant amount due to their faster convergence properties. Compared with classical functions, the training time is reduced by 71.61% on average for MNIST and 49.31% on average for FMNIST. When compared with improved versions of classical functions, the training time is reduced by 67.78% on average for MNIST and 50.28% on average for FMNIST. When compared with existing fractional functions, the training time is reduced by 87.08% on average for MNIST and 66.11% on average for FMNIST.

8. Conclusions

The proposed new family of fractional activation functions has several potential benefits. It is a viable choice for rapid training because it can lead to faster convergence and shorter time frames. Another significant advantage is the enhanced adaptability, which is crucial when working with complex data. In addition to boosting generalization during training, the proposed activation functions can reduce training costs, enabling larger and more complex neural networks to be trained. Because of its enhanced ability to pick up on subtleties and nuanced patterns, it can better adapt to complex data sets, which can be challenging for traditional activation functions. Overfitting, where a neural network becomes overly complex while fitting the training data too closely, is also mitigated. Also, the proposed functions show considerable benefits over more complex fractional learning algorithms, making them an attractive candidate for practical use due to their simplicity and ease of implementation. Finally, compared to existing fractional activation functions, the proposed functions are more computationally efficient and can be integrated with many neural architectures. Also, the presented functions can reduce training time drastically, as seen from numerical assessments. Our functions can be feasible with more than one layer network, while previous fractional activation functions would fail to perform. The suggested fractional activation function for neural network training has the potential to improve training performance and efficiency even more.

Future work will deal with the automatic selection of the best fractional order in the activation function by considering it as a weight to be estimated by backpropagation during training. In this sense, it can be said that the data determine the fractional order. The next possible study is to verify the functions for deep network architectures.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A

Proof of Eq. (11). The RL-type conformable fractional derivative is defined in (6). Given $f(t) = e^{-t}$

$$
\begin{aligned}
{}^{RL}_{0}T_{\alpha}(f(t)) &= \lim_{\varepsilon\to 0}\left[(1-\alpha)(e^{-t}) + \alpha\frac{e^{-(t+\varepsilon(t-\alpha)^{1-\alpha})} - e^{-t}}{\varepsilon}\right] \\
&= \lim_{\varepsilon\to 0}\left[(1-\alpha)(e^{-t})\right] + \alpha\lim_{\varepsilon\to 0}\left[\frac{e^{-(t+\varepsilon(t-\alpha)^{1-\alpha})} - e^{-t}}{\varepsilon}\right] \\
&= (1-\alpha)(e^{-t}) + \alpha\lim_{\varepsilon\to 0}\left[\frac{e^{-(t+\varepsilon(t-\alpha)^{1-\alpha})} - e^{-t}}{\varepsilon}\right]
\end{aligned}
$$

$$
\begin{aligned}
&= (1-\alpha)(e^{-t}) + \alpha \lim_{\varepsilon \to 0} \frac{\frac{d}{d\varepsilon}\left[e^{-(t+\varepsilon t^{1-\alpha})}\right] - \frac{d}{d\varepsilon}\left[e^{-t}\right]}{\frac{d}{d\varepsilon}\varepsilon} \\
&= (1-\alpha)(e^{-t}) + \alpha \lim_{\varepsilon \to 0} \frac{-t^{1-\alpha} e^{-(t+\varepsilon t^{1-\alpha})} - 0}{1} \\
&= (1-\alpha)(e^{-t}) - \alpha t^{1-\alpha} e^{-t}
\end{aligned}
$$

Appendix B

Proof of Eq. (14). Since the sigmoid function is symmetric around the origin and returns a value in the range [0, 1], the following relationship can be written:

$$
\begin{aligned}
\sigma(-t) &= 1 - \sigma(t) \\
\frac{1}{1+e^{t}} &= \frac{1+e^{t}-e^{t}}{1+e^{t}} \\
&= 1 - \frac{e^{t}}{1+e^{t}} \\
&= 1 - \frac{1}{1+e^{-t}}
\end{aligned}
$$

Now, to see the relationship between tanh and σ, let us rearrange the tanh function into a similar form by

$$
\begin{aligned}
\tanh(t) &= \frac{e^{t}-e^{-t}}{e^{t}+e^{-t}} \\
&= \frac{e^{t}-e^{-t}-2e^{-t}}{e^{t}+e^{-t}} \\
&= 1 + \frac{-2e^{-t}}{e^{t}+e^{-t}} \\
&= 1 - \frac{2}{e^{2t}+1}
\end{aligned}
$$

Now, from the logistic sigmoid's perspective, it follows:

$$
\begin{aligned}
\tanh(t) = 1 - \frac{2}{e^{2t}+1} &= 1 - 2\sigma(-2t) \\
&= 1 - 2(1-\sigma(2t)) \\
&= 1 - 2 + 2\sigma(2t) \\
&= 2\sigma(2t) - 1
\end{aligned}
$$

Hence, it can be concluded that the tanh function is just a rescaled version of the logistic sigmoid function.

References

Agostinelli, F., Hoffman, M., Sadowski, P., Baldi, P., 2014. Learning activation functions to improve deep neural networks. arXiv preprint arXiv:1412.6830.

Aguilar, C.Z., Gómez-Aguilar, J., Alvarado-Martínez, V., Romero-Ugalde, H., 2020. Fractional order neural networks for system identification. Chaos Solitons Fractals 130.

Alcantara, G., 2017. Empirical analysis of non-linear activation functions for deep neural networks in classification tasks. arXiv preprint arXiv:1710.11272.

Altan, G., Alkan, S., Baleanu, D., 2023. A novel fractional operator application for neural networks using proportional Caputo derivative. Neural Comput. Appl. 35 (4), 3101–3114.

Altan, G., Alkan, S., Kutlu, Y., 2020. Fractional-order activation function for feed forward neural networks using conformable derivative. Artificial Intelligence Appl. 1, 11–14.

Chen, W., Shi, K., 2021. Multi-scale attention convolutional neural network for time series classification. Neural Netw. 136, 126–140.

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

Dubey, S.R., Chakraborty, S., 2021. Average biased ReLU based CNN descriptor for improved face retrieval. Multimedia Tools Appl. 80, 23181–23206.

Dubey, S.R., Singh, S.K., Chaudhuri, B.B., 2022. Activation functions in deep learning: A comprehensive survey and benchmark. Neurocomputing 503 (7), 92–108.

Gao, F., Chi, C., 2020. Improvement on conformable fractional derivative and its applications in fractional differential equations. J. Funct. Spaces 2020, 1–10.

Gill, H.S., Khehra, B.S., 2022. An integrated approach using CNN-RNN-LSTM for classification of fruit images. Mater. Today Proc. 51, 591–595.

Ivanov, A., 2018. Fractional activation functions in feedforward artificial neural networks. In: 2018 20th International Symposium on Electrical Apparatus and Technologies. SIELA, IEEE, pp. 1–4.

Job, M.S., Bhateja, P.H., Gupta, M., Bingi, K., Prusty, B.R., 2022. Fractional rectified linear unit activation function and its variants. Math. Probl. Eng. 2022.

Karlik, B., Olgac, A.V., 2011. Performance analysis of various activation functions in generalized mlp architectures of neural networks. Int. J. Artif. Intell. Expert Syst. 1 (4), 111–122.

Khalil, R., Al Horani, M., Yousef, A., Sababheh, M., 2014. A new definition of fractional derivative. J. Comput. Appl. Math. 264, 65–70.

Kothari, K., Mehta, U., Prasad, R., 2019. Fractional-order system modeling and its applications. J. Eng. Sci. Technol. Rev. 12 (6), 1–10.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60 (6), 84–90.

Lau, M.M., Lim, K.H., 2018. Review of adaptive activation function in deep neural network. In: 2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences. IECBES, IEEE, pp. 686–690.

Li, B., Tang, S., Yu, H., 2019. PowerNet: Efficient representations of polynomials and smooth functions by deep neural networks with rectified power units. arXiv preprint arXiv:1909.05136.

Mehta, U., Bingi, K., Saxena, S., 2022. Applied Fractional Calculus in Identification and Control. Springer Nature.

Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning. ICML-10, pp. 807–814.

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

Ramachandran, P., Zoph, B., Le, Q.V., 2017. Searching for activation functions. arXiv preprint arXiv:1710.05941.

Roy, S.K., Manna, S., Dubey, S.R., Chaudhuri, B.B., 2022. LiSHT: Non-parametric linearly scaled hyperbolic tangent activation function for neural networks. In: International Conference on Computer Vision and Image Processing. Springer, pp. 462–476.

Scardapane, S., Van Vaerenbergh, S., Totaro, S., Uncini, A., 2019. Kafnets: Kernel-based non-parametric activation functions for neural networks. Neural Netw. 110, 19–32.

Solís-Pérez, J., Hernández, J., Parrales, A., Gómez-Aguilar, J., Huicochea, A., 2022. Artificial neural networks with conformable transfer function for improving the performance in thermal and environmental processes. Neural Netw. 152, 44–56.

Su, Q., Carin, L., et al., 2017. A probabilistic framework for nonlinearities in stochastic neural networks. Adv. Neural Inf. Process. Syst. 30.

Sun, K., Yu, J., Zhang, L., Dong, Z., 2020. A convolutional neural network model based on improved softplus activation function. In: International Conference on Applications and Techniques in Cyber Intelligence ATCI 2019: Applications and Techniques in Cyber Intelligence 7. Springer, pp. 1326–1335.

Viera-Martin, E., Gómez-Aguilar, J., Solís-Pérez, J., Hernández-Pérez, J., Escobar-Jiménez, R., 2022. Artificial neural networks: A practical review of applications involving fractional calculus. Eur. Phys. J. Spec. Top. 231 (10), 2059–2095.

Wang, J., Wen, Y., Gou, Y., Ye, Z., Chen, H., 2017. Fractional-order gradient descent learning of BP neural networks with Caputo derivative. Neural Netw. 89, 19–30.

Yıldız, İ., Tian, P., Dy, J., Erdoğmuş, D., Brown, J., Kalpathy-Cramer, J., Ostmo, S., Campbell, J.P., Chiang, M.F., Ioannidis, S., 2019. Classification and comparison via neural networks. Neural Netw. 118, 65–80.

Yu, J., Hu, C., Jiang, H., 2012. α-Stability and α-synchronization for fractional-order neural networks. Neural Netw. 35, 82–87.

Zamora, J., Rhodes, A.D., Nachman, L., 2022. Fractional adaptive linear units. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, No. 8. pp. 8988–8996.

Zamora Esquivel, J., Cruz Vargas, A., Camacho Perez, R., Lopez Meyer, P., Cordourier, H., Tickoo, O., 2019. Adaptive activation functions using fractional calculus. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. pp. 2006–2013.

Zhang, G.P., 2000. Neural networks for classification: A survey. IEEE Trans. Syst. Man Cybern. C 30 (4), 451–462.

Zhang, S., Chen, Y., Yu, Y., 2017. A survey of fractional-order neural networks. In: Proceedings of the ASME 2017 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 58233. American Society of Mechanical Engineers, Cleveland, Ohio, USA.

Zou, T., Qu, J., Chen, L., Chai, Y., Yang, Z., 2014. Stability analysis of a class of fractional-order neural networks. TELKOMNIKA Indonesian J. Electr. Eng. 12 (2), 1086–1093.