Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

A Novel Fixed-Parameter Activation Function for Neural Networks: Enhanced Accuracy and Convergence on MNIST

Najmeh Hosseinipour-Mahani[✉], **Code Orcide:** 0000-0003-2311-2040
Department of Applied Mathematics, Graduate University of Advanced Technology, Kerman, Iran, nhoseinipour@gmail.com
Amirreza Jahantab, **Code Orcide: 0009-0005-1946-8353**
Department of Computer science, Shahid Bahonar University of Kerman, Kerman, Iran, Jahantab83@gmail.com

Abstract — Activation functions are essential for extracting meaningful relationships from real-world data in deep learning models. The design of activation functions is critical, as they directly influence the performance of these models. Nonlinear activation functions are commonly preferred since linear functions can limit a model's learning capacity. Nonlinear activation functions can either have fixed parameters, which are predefined before training, or adjustable ones that modify during training. Fixed-parameter activation functions require the user to set the parameter values prior to model training. However, finding suitable parameters can be time-consuming and may slow down the convergence of the model. In this study, a novel fixed-parameter activation function is proposed and its performance is evaluated using benchmark MNIST datasets, demonstrating improvements in both accuracy and convergence speed.

Keywords —Activation Function, Deep Learning, Fixed-Parameter, Neural Networks, MNIST Dataset, Nonlinear Function, Gradient Optimization, Vanishing Gradient Problem

I. Introduction

Deep learning has become a cornerstone of modern artificial intelligence, powering breakthroughs in fields such as computer vision, natural language processing, and autonomous systems. Central to the success of deep learning models are activation functions, which introduce non-linearity to neural networks, allowing them to capture complex patterns and relationships in data. Without appropriate activation functions, neural networks would behave as linear models, limiting their capacity to solve real-world, non-linear problems [15, 16]. Activation functions can be broadly categorized into linear and nonlinear types. Linear activation functions, although simple, restrict the representational power of deep neural networks. For this reason, nonlinear activation functions such as ReLU, Sigmoid, and ELU are commonly used to enable the network to learn more complex data representations.

A key characteristic of these nonlinear functions is the ability to determine how neurons should fire and adjust their responses based on the input values. Most nonlinear activation functions in the literature are designed with fixed parameters. These parameters are constants defined before training, and they remain unchanged throughout the learning process. While fixed-parameter activation functions like ReLU and ELU have proven to be effective, they often require careful selection of parameters to optimize the model's learning capabilities.

In this study, we propose a novel fixed-parameter activation function that operates across three distinct regions: negative, linear, and positive. Unlike some existing functions, the parameters in our activation function are predefined before training and remain constant throughout the learning process. The aim is to provide an efficient activation mechanism that balances flexibility in capturing nonlinear patterns with computational efficiency. We evaluate the performance of this new activation function on the MNIST benchmark dataset to demonstrate its effectiveness in improving both accuracy and convergence speed.

II. Related work

Over the past two decades, numerous fixed-parameter and trainable activation functions have been developed to enhance the training performance of deep learning models. Some of these include ReLU, LReLU, ELU, RSigELU, GeLU, FeLU, PReLU, DReLU, and PELU [3, 5, 7, 10, 13, 14]. The ReLU activation function was introduced to address the vanishing gradient problem by allowing negative values to be transformed into positive ones [3]. The LReLU activation function incorporates negative weights during the training process [4]. Subsequently, the ELU activation function was introduced as an alternative to ReLU to address issues related to vanishing gradients and negative weights [9]. Building on this, the SELU activation function was developed to enhance the training performance of ELU [5]. Fixed-parameter activation functions require predetermined parameters prior to the training of deep learning models, which can result in slower convergence rates. Additionally, identifying

Submit Date: 2025-09-28
Accept Date: 2025-10-05
✉ Corresponding author

appropriate parameter values for activation functions often necessitates extensive experimentation, making it a time-consuming endeavor.

After that, a variety of trainable activation functions, including ALISA, PELU, PRELU, P+FELU, PSigmoid and GELU, have been proposed [9]. These functions allow the model to fine-tune parameters during training, which is done through backpropagation. This technique, commonly used in deep learning, adjusts the weights, biases, and activation function parameters across neurons in the model. As training progresses, the parameters of these activation functions are updated in each iteration, allowing the model to find the most suitable weights. This adaptive approach enables trainable activation functions to achieve more efficient and accurate convergence by optimizing parameter values for different datasets and neural network architectures [11].

The ALISA activation function is indeed designed to mitigate issues related to the vanishing gradient problem, which can occur in deep neural networks [9]. The PELU activation function enhances the ELU by incorporating a trainable scalar parameter. This addition aims to address the bias shift problem, allowing for more flexibility and improved performance in neural network training [2]. The PReLU activation function was proposed as an alternative to the standard ReLU activation function to address some of its limitations, particularly the issue of negative weights [7]. The P+FELU activationfunction has been proposed to address the issues of vanishing gradient and negative weights [8]. The PSigmoid activation function was introduced as an enhancement to the Squeeze and Excitation (SE) Networks block [1]. The GELU activation function serves as an alternative to ReLU and ELU, showing performance improvements across various tasks in computer vision, natural language processing, and speech [12]. Additionally, the parametric RSigELU (P+RSigELU) includes trainable activation functions such as P+RSigELU Single (P+RSigELUS) and P+RSigELU Double (P+RSigELUD), which were developed by extending the RSigELU activation function [11].

The vanishing gradient problem is indeed a challenge associated with the Sigmoid and $Tanh$ activation functions, particularly in deep neural networks, as these functions can lead to very small gradients during backpropagation. The ReLU, derivatives, and the most suitable activation function help mitigate this issue. However, it is important to note that ReLU does not activate for negative inputs, effectively ignoring them, which can lead to the "dying ReLU" problem where neurons can become inactive. Behaviors of activation functions in the literature are shown in Fig.1.

As can be seen in Fig. 1, P+RSigELUS is not continuous at $x = 1$, and consequently, it is not differentiable either.

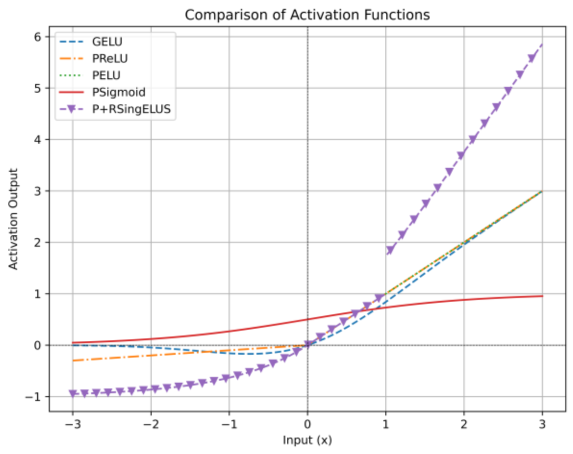

Fig. 1. Behaviors of activation functions

In this study, we propose a new activation function designed to enhance performance across both negative and positive input regions. This function addresses issues associated with the negative region, bias shift, and vanishing gradients. By actively engaging with inputs from both sides, and by incorporating a constant gradient parameter, our activation function offers greater flexibility. This approach helps reduce parameter errors for each neuron and supports a more effective learning process.

III. PROPOSED ACTIVATION FUNCTION

A. Introduction to Activation Function

In this section, a novel activation function is proposed that operates across three distinct regions: negative, linear, and positive. The function's behavior is governed by the input value x, with distinct formulations for each region. To enhance flexibility and performance, the function introduces adjustable parameters, which are carefully tuned prior to training based on the characteristics of the dataset or task. These parameters remain constant during training, ensuring a stable and computationally efficient implementation. The proposed activation function is defined as follows:

$$f(x) = \begin{cases} \frac{ax}{1+e^{x}} & -\infty < x < 0 \\ x & 0 \le x \le 1 \\ ae^{-x}(x-1)+1 & 1 < x < \infty \end{cases} \qquad (1)$$

For $x < 0$, the negative region is controlled by the hyperparameter a. This allows the function to smoothly handle negative inputs while maintaining a balance between linearity and nonlinearity. For $0 \leq x \leq 1$, the function behaves linearly. In this region, the function passes the input directly, which helps maintain the gradient flow during backpropagation, reducing the risk of vanishing gradients. For $x > 1$, the positive region allows the function to grow exponentially as $x$increases, while the hyperparameter a adjusts the steepness of the curve, enhancing the model's ability to capture complex patterns in the data. By incorporating parameter a, the activation function adapts to the input distribution, improving the model's capacity to learn from data efficiently. The proposed function is designed to overcome common issues such as vanishing gradients and inefficient learning in deep networks. Additionally, it retains the benefits of hybrid activation functions, combining aspects of ReLU, ELU, and sigmoid functions, which ensures stable training and faster convergence. The introduction of parameterized slopes in both the negative and positive regions allows the network to better capture subtle variations in data, ultimately improving accuracy and convergence speed. In summary, this new activation function offers flexibility through parameterized control of its behavior in different regions, making it a suitable choice for deep learning architectures.
The proposed activation function behaves as shown in Fig. 2.

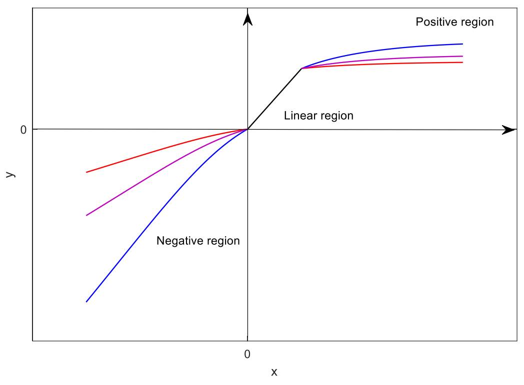

Fig. 2. Behavior of the proposed activation function

The derivative of the proposed activation function is given in the following:

$$\frac{df(x)}{dx} = \begin{cases} \dfrac{ae^{-x}(e^{-x} + x + 1)}{(1 + e^{-x})^2} & -\infty < x < 0 \\ 1 & 0 \leq x \leq 1 \\ ae^{-x}(2 - x) & 1 < x < \infty \end{cases}$$

In deep learning architectures, backpropagation is a key method used to update parameters during the learning process [6]. An important aspect of this is that the proposed activation functions must be differentiable. Notably, the activation function remains active in both the negative and positive regions after differentiation. Furthermore, the weights are initialized by randomly sampling from a normal distribution, which helps maintain appropriate variance across all layers. Consequently, the range of value generation and the steepness of the function are not fixed, as seen in logistic sigmoid and hyperbolic tangent functions, but instead fluctuate. While designing the network architecture, there are typically two approaches to utilizing free parameter functions. The first approach involves structures where each neuron in the network, with the exception of those in the final layer, possesses its own set of training parameters. The second approach consists of structures in which the neurons within each hidden layer share common fixed parameters. In this study, the first method was preferred.

The proposed activation function offers several advantages: it adjusts the slope parameters during training to optimize the values for each neuron in every iteration. This function effectively tackles issues like vanishing gradients and negative regions, thus enhancing the overall learning process. Additionally, it provides improved accuracy and faster convergence compared to previous activation functions. Notably, the proposed activation function is both parametric and monotonic.

B. Region-Wise Analysis of the Proposed Activation Function and the Role of the Hyperparameter a

In our proposed activation function, the hyperparameter a plays a crucial role in shaping the output response to different input values. This function consists of three distinct regions:

1. **For negative values of x:**

$f(x) = \frac{ax}{1+e^{x}}$, for $-\infty < x < 0$.

In this region, a controls the slope of the function, influencing how sharply negative inputs are transformed. A higher a results in a steeper response, whereas a lower a leads to a more gradual transition.

2. **For values between 0 and 1**:

   $f(x) = x, \qquad \text{for} \qquad 0 \leq x \leq 1.$

   In this range, the function behaves linearly, maintaining the identity mapping. The hyperparameter a does not have an impact in this region.

3. **For positive values of x:**

   $f(x) = ae^{-x}(x - 1) + 1, \qquad \text{for} \quad 1 < x < \infty.$

   Here, a determines the rate at which the function decays as x increases. A larger a slows down the decay, allowing the function to remain closer to 1 for a longer range of x, while a smaller a leads to faster convergence towards 1.

IV. Materials and method

A. Benchmark datasets

The MNIST dataset is widely recognized as a standard benchmark for evaluating machine learning models, particularly in computer vision tasks. It contains a total of 70,000 grayscale images; each sized 28×28 pixels, representing handwritten digits (0 through 9). The dataset is divided into 60,000 training samples and 10,000 test samples. Each image is accompanied by a corresponding label indicating the digit. Preprocessing steps included normalization of pixel values to a mean of 0.1307 and a standard deviation of 0.3081. This standardization accelerates convergence during training by ensuring consistent input distributions across all samples.

B. Convolutional neural network

To evaluate the performance of the proposed activation function, a convolutional neural network (CNN) inspired by the VGG architecture was implemented. The key modifications to the traditional VGG structure include the use of fewer layers, which is suitable for the MNIST dataset, and the integration of the proposed activation function at every stage of the network.

The architecture comprises four convolutional blocks, each followed by max-pooling and dropout layers to mitigate over fitting. Specifically:

I. **Convolutional Layers:** Four convolutional layers progressively increase the feature depth (32, 48, 64, and 96 channels, respectively) while preserving spatial resolution using a kernel size of 3×3.

II. **Activation Function:** The custom activation function was applied after each convolutional operation to introduce non-linearity and allow the network to model complex patterns in the data.

III. **Pooling and Dropout:** Max-pooling with a 2×2 kernel and a dropout rate of 0.25 were employed to reduce spatial dimensions and prevent over fitting.

IV. **Fully Connected Layers:** After flattening, the feature maps were passed through two fully connected layers (512 and 10 units), where the final layer outputs class probabilities using the softmax function.

C. Optimization Strategy for a

To determine the optimal value of a, we employed a cross-validation strategy combined with a coarse-to-fine search approach:

1. **Coarse Search:** We began with a broad range of possible values for a and conducted a few short training runs (e.g., 5 epochs).
   This step helped us get a rough estimate of suitable values for a that prevent instability and facilitate effective learning.
2. **Fine-tuned Search:** After identifying a promising range, we performed longer training runs with more refined values. The best value for a was selected based on the validation performance of the model. To prevent instability, we stopped training early if the loss exceeded three times its initial value, which signaled potential divergence.

Final Chosen Value

Following the optimization process, we determined that the best value for a in our model was $a = 0.5$. This value provided a balanced trade-off between stability and adaptability, ensuring smooth gradient flow and effective learning dynamics.

V. Results and discussion

A. Experimental evaluation

The experimental results from five training repeats of the proposed activation function on the MNIST dataset are summarized in TABLE I. The table provides metrics for each repeat, as well as the average performance across all runs.

TABLEI. Experimental Results of the Proposed Function for MNIST Dataset

Repeat	EPOCH	Train Loss	Train Acc	Val. Loss	Val. Acc
Repeat 1	40	0.0075	99.78%	0.0601	98.84%
Repeat 2	40	0.0053	99.84%	0.0585	99.01%
Repeat 3	40	0.0056	99.84%	0.0602	99.02%
Repeat 4	40	0.0054	99.83%	0.0590	99.12%
Repeat 5	40	0.0077	99.79%	0.0627	98.96%
Average		0.0063	99.82%	0.0601	98.99%

In addition, the proposed function was benchmarked against other established activation functions. The comparison, shown in TABLE II., highlights its superior performance across key metrics, including training loss, training accuracy, validation loss, and validation accuracy.

TABLE II. Average Success Rates of Activation Functions for MNIST Dataset

Activation	Train_Loss	Train_Acc	Val._Loss	Val._Acc
PReLU	0.2782	0.9139	0.2011	0.9383
PELU	0.2721	0.9187	0.2134	0.9362
ALISA	0.3018	0.9095	0.2275	0.9335
GELU	0.3726	0.8933	0.2203	0.9244
P+FELU	0.3255	0.8941	0.2127	0.9109
PSigmoid	0.2593	0.9009	0.2207	0.9214
P+RSigELUS	0.2501	0.9287	0.1805	0.9471
P+RSigELUD	0.2062	0.9380	0.1458	0.9564
Proposed Function	0.0063	0.9982	0.0601	0.9899

B. Discussion

The results in TABLE I demonstrate the consistency and robustness of the proposed function, with an average training accuracy of 99.82% and validation accuracy of 98.99%. The low variation across the five repeats confirms its stability during training.

In TABLE II, the proposed function outperforms all compared activation functions, achieving the lowest training and validation loss, and the highest accuracy.
These findings underscore its effectiveness and potential as a preferred activation function for deep learning tasks.

VI. Conclusion and future work

In this paper, we introduced a novel fixed-parameter activation function for neural networks aimed at enhancing both accuracy and convergence speed, particularly on the MNIST dataset. Our proposed activation function outperforms established methods such as PSigmoid, GELU, and PReLU in terms of accuracy and convergence speed on the MNIST benchmark. However, we are currently testing its performance on more challenging tasks and diverse datasets to evaluate its broader applicability and robustness. We observed that the proposed activation function initially showed limited adaptability to datasets with significantly different characteristics, such as CIFAR-10, making it less versatile compared to trainable activation functions. However, by incorporating the ResNet architecture, we were able to enhance its performance, achieving an accuracy of almost 90% on CIFAR-10. This demonstrates that architectural modifications can significantly improve the function's generalizability. Further exploration of such enhancements will be discussed in the future work.

REFERENCES

[1] Y. Ying, N. Zhang, P. Shan, L. Miao, P. Sun, and S. Peng, "PSigmoid: improving squeeze-and-excitation block with parametric sigmoid," Appl. Intell., vol. 51, no. 10, pp. 7427–7439, 2021.

[2] L. Trottier, P. Gigu, and B. Chaib-draa, "Parametric exponential linear unit for deep convolutional neural networks," in Proc. 16th IEEE Int. Conf. Machine Learning and Applications (ICMLA), pp. 207–214, 2017.

[3] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in Proc. 27th Int. Conf. Machine Learning (ICML-10), pp. 807–814, 2010.

[4] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in Proc. 30th Int. Conf. Machine Learning (ICML), vol. 30, no. 1, p. 3, 2013

[5] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in Proc. Adv. Neural Inf. Process. Syst., vol. 30, pp. 971–980, 2017.

[6] S. Kiliçarslan, K. Adem, and M. Celik, "An overview of the activation functions used in deep learning algorithms," J. New Result Sci., vol. 10, no. 3, pp. 75–88, 2021.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, pp. 1026–1034, 2015.

[8] K. Adem, "P+FELU: flexible and trainable fast exponential linear unit for deep learning architectures," Neural Comput. Appl., vol. 34, no. 24, pp. 21729-21740, 2022.

[9] V. S. Bawa and V. Kumar, "Linearized sigmoidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability," Expert Syst. Appl., vol. 120, pp. 346–356, 2019.

[10] S. Kiliçarslanand M. Celik, "RSigELU: a nonlinear activation function for deep neural networks," Expert Syst. Appl., vol. 174, p. 114805, 2021.

[11] S. Kiliçarslan and M. Celik, "Parametric RSigELU: a new trainable activation function for deep learning," Neural Computing and Applications, vol. 36, pp. 7595–7607, 2024.

[12] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," arXiv preprint arXiv:1606.08415, 2016.

[13] K. Biswas, S. Kumar, S. Banerjee and A. K. Pandey, "TanhSoft—Dynamic Trainable Activation Functions for Faster Learning and Better Performance," in IEEE Access, vol. 9, pp. 120613-120623, 2021.

[14] E. Işık, N. Ademović, E. Harirchian, F. Avcil, A. Büyüksaraç, M. Hadzima-Nyarko and B. Antep, "Determination of natural fundamental period of minarets by using artificial neural network and assess the impact of different materials on their seismic vulnerability," Appl. Sci., vol. 13, no. 2, pp. 2076–3417, 2023.

[15] I. Pacal and S. Kılıcarslan, "Deep learning-based approaches for robust classification of cervical cancer," Neural Computing and Applications, vol. 35 (25), pp. 18813–18828, 2023.

[16] H. Gao, Z. Wang, L. Cai and S. Ji, "ChannelNets: Compact and Efficient Convolutional Neural Networks via Channel-Wise Convolutions," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 8, pp. 2570-2581, 1 Aug. 2021.