A-IPLA: A symmetric, non-monotonic activation function for convolutional neural network

Nguyen Van Truong
Military Industrial College, Vietnam

ABSTRACT

In this paper, a new symmetric origin, non-saturated, and non-monotonic activation function called adaptive inverse proportional linear activation (A-IPLA) is proposed to avoid the output deviation problem, improve the neural network recognition performance and convergence speed. The function uses the piecewise activation method which each segment has different activation behaviors including linear activation and inverse proportional activation according to the change of the piecewise point and the slope coefficient. Firstly, A-IPLA is the origin symmetry function that avoids the non-zero output mean, solving the output deviation problem. Secondly, the slope coefficient of A-IPLA can be learned in the training process. Experiments using the Lenet-5 models performed on various benchmark datasets show that the proposed activation functions outperform the state-of-the-art activation functions in all tests with recognition accuracy improvements of 0.20% on MNIST and 5.34% on CIFAR-10. Meanwhile, under the same accuracy requirement, the convergence rate of the proposed function is 3.55x faster on MNIST and 2.69x faster on CIFAR-10, respectively.

***Keywords**: Activation function, inverse proportional linear function, non-monotonic activation, convolutional neural network*

1. INTRODUCTION

In recent years, artificial neural network (ANN) has been successfully applied in many fields such as image recognition [1], speech recognition [2], object detection [3] and computer vision [4]. One of the important factors that give neural networks such success is the activation function, which is used to activate neurons in the feedforward process and update weights, bias in the back-propagation process [5 - 7]. Sigmoid [8] is a common activation function, however, the sigmoid function takes the point of (0, 0.5) as the center of symmetry, and the output of the function is greater than zero, which causes the convergence speed to be slowed. The hyperbolic tangent (tanh) function with the symmetry center is the origin point is introduced to effectively accelerate the convergence rate [9]. But like the sigmoid function, the hyperbolic tangent function also has saturation zones, which leads to the vanishing gradient problem in the training process [10]. Some improvement functions such as Elliot [11], Algebraic sigmoid [12], Parametric Algebraic [13] are presented, but the problem is not completely solved.

After that, a non-saturation activation function called Rectified Linear Unit (ReLU) is proposed by Nair & Hinton in 2010 to solve the problem of vanishing gradient and becomes one of the widely used activation functions of ANN [14]. ReLU is a linear function with an identity function in the positive X-axis, and zero in the negative X-axis that guarantees fast

Corresponding author: PhD. Nguyen Van Truong
Email: nvtruong.25890@gmail.com

computation since it avoids the exponentials and divisions computations. However, since the gradient of ReLu in the negative part is zero, the weights are not updated, leading to the formation of "dead neurons". Therefore, many activation functions are proposed to avoid the ReLU "dead neurons" [15 - 19], but the result is still not really ideal. Linear sigmoidal activation proposed by Vivek Singh Bawa is another new non-saturated linear activation function, in which the functions are defined as the first-order functions in all intervals to avoid the generation of dead neurons and compress the data amplitude [20].

All of the above activation functions are monotonic. However, the papers [21 - 23] introduce serial non-monotonic activation functions called SWish family indicating that the SWish function performs as well as the other monotonic activation functions. In addition, these functions all have the non-zero output mean that leads to the output displacement problem.

This paper proposes a new activation function called inverse proportional linear activation (IPLA) function, which holds the advantages of the above functions such as the original symmetry of tanh function, piecewise linear behavior of non-saturation functions, and non-monotonic property of Swish function, prevents the output displacement problem, improves the network recognition accuracies and convergence performance effectively. The main contributions of this work include:

1. A parametric inverse proportional linear activation (PIPLA) function including the properties of origin symmetric, non-monotonic is proposed. The functions are defined according to the change of the input data range, and slope coefficient, improving the recognition accuracy and convergence.
2. Adaptive inverse proportional linear activation (A-IPLA) function is proposed, in which the slope coefficients are learned in the training process.
3. The recognition accuracy and convergence performance of proposed activation functions are analyzed and compared with the state-of-the-art functions on bench-mark datasets MNIST and CIFAR-10 using the Lenet-5 CNN model.

The rest of the paper is organized as follows. Section 2 describes the state-of-the-art activation functions. Section 3 presents the proposed activation functions. The experimental results and comparisons are shown in Section 4. The conclusion is presented in Section 5.

2. ACTIVATION FUNCTION FAMILY

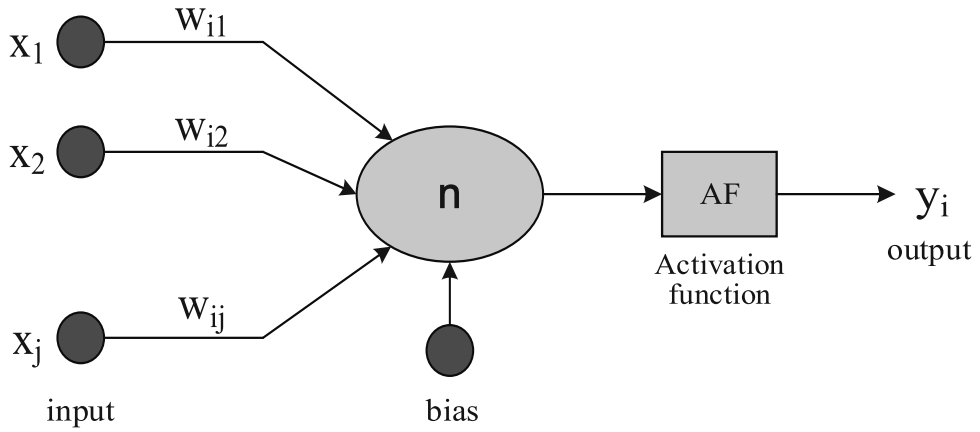


Figure 1. The computational principle of the ith neuron

An artificial Neural Network (ANN) is a multi-layer neural network structure in which each layer has over hundreds of neurons. Each neuron in the hidden layers and output layer shown in Figure 1 is calculated by multiplication, addition, and activation operation. The formula is defined as follows:

$$f(x) = g(w * x + b) \quad (1)$$

Where *x* is the input of the activation function, *w* is the weight connected between two layers, *b* is the bias, and *g* is the activation function. The training performance of a neural network has relied heavily on the activation function selection. Currently, commonly used activation functions include:

2.1. Sigmoid function

The sigmoid function is the most standard form of activation function that is widely used in ANNs. It is an S-shaped function with an output range of [0, 1] and can be defined by Eq. 2:

$$sigmoid(z) = \frac{1}{1 + e^{-z}} \quad (2)$$

where *z* is the input of the activation function. The gradient of the sigmoid is:

$$\frac{d}{dz} sigmoid(z) = sigmoid(z).(1 - sigmoid(z)) \quad (3)$$

The sigmoid function is complicated since it requires a lot of exponential and division operations.

2.2. Hyperbolic tangent function

Similar to the sigmoid function, the tanh function is also an S-shaped curve. However, this function takes the origin of coordinates as the symmetry center, and the output ranges from -1 to 1, which effectively improves the network convergence. The tanh function is given by:

$$tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \quad (4)$$

And the gradient is given by:

$$\frac{d}{dz} tanh(z) = 1 - tanh^{2}(z) \quad (5)$$

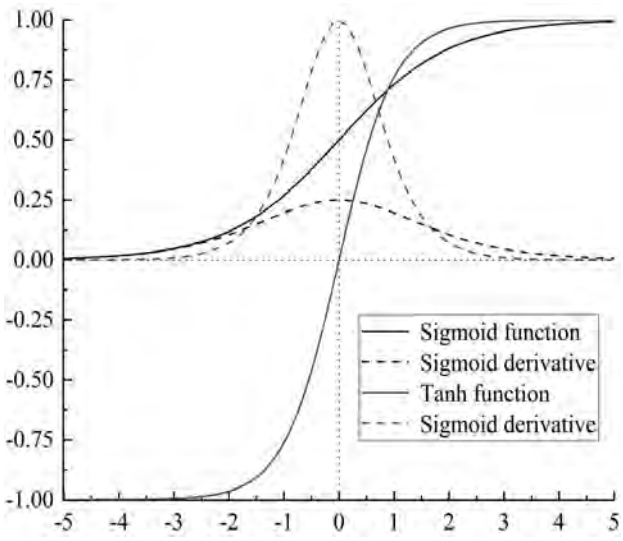


Figure 2. Function and derivative curves of sigmoid and tanh

Figure 2 shows the curves of sigmoid, tanh functions, and their derivatives. They both have the vanishing gradient problem due to the gradient is close to zero in the saturation intervals.

2.3. ReLU function

ReLU is a non-saturation, piecewise linear activation function with the following form:

$$ReLU(z) = \begin{cases} z, z \geq 0 \\ 0, other \end{cases} \quad (6)$$

Which has the gradient:

$$\frac{d}{dz} ReLU(z) = \begin{cases} 1, z \geq 0 \\ 0, other \end{cases} \quad (7)$$

ReLU provides an unbounded linear curve with the fixed gradient of 1 in the positive interval, effectively solving the vanishing gradient problem. Besides, simple calculation and fast convergence are the other advantages of ReLU. However, ReLU also has the disadvantage that the ReLU output and gradient in the negative interval are equal to zero, resulting in some neurons never being activated. To solve this problem, some improved activation functions are proposed, such as LReLU, PLReLU, ELU, and PELU [23, 24, 25].

2.4. Swish function

The Swish function proposed by Ramachandran et al. in 2017 is one of the first non-monotonic activation functions. It is a hybrid function of sigmoid and ReLU, which is bounded, non-monotonic in the negative region, and unbounded, monotonic in the positive region. The Swish function and its derivative are defined as follows:

$$Swish(z) = z.\frac{1}{1+e^{-z}} \quad (8)$$

$$\frac{d}{dz}Swish(z) = Swish(z) + sigmoid(z).(1 - Swish(z)) \quad (9)$$

2.5. Linearized sigmoid activation function

$$LiSA(z) = \begin{cases} \alpha_1 z - \alpha_1 + 1, if 1 < z < \infty \\ z, if 0 \le z \le 1 \\ \alpha_2 z, if -\infty < z < 0 \end{cases} \quad (10)$$

The linearized sigmoid activation (LiSA) function that the formula and its derivative are shown in Eq. 10, and Eq. 11, respectively is the linear function with three linear intervals.

$$\frac{d}{dz}LiSA(z) = \begin{cases} \alpha_1, if 1 < z < \infty \\ 1, if 0 \le z \le 1 \\ \alpha_2, if -\infty < z < 0 \end{cases} \quad (11)$$

The function outputs in the negative and positive regions are controlled by the slope coefficient field α_1, α_2 to compress the function amplitude. The curves and derivatives of ReLU, Swish, and LiSA function are shown in Figure3.

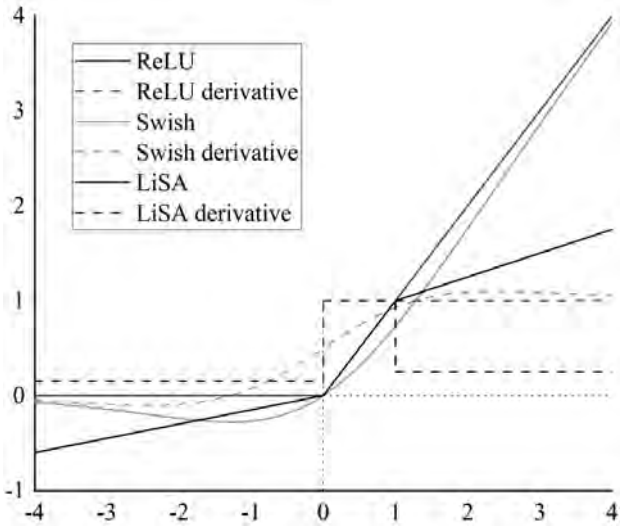


Figure 3. Function and derivative curves of ReLU, Swish, and LiSA

3. THE PROPOSED ACTIVATION FUNCTION

3.1. Parametric inverse proportional linear activation function (PIPLA)

In this paper, a new origin symmetric, piecewise non-monotonic function called parametric inverse proportional linear activation (PIPLA) is proposed to solve the problems of vanishing gradient, dead neurons, output displacement, and improve the overall performance of the ANN model. The PIPLA function is defined as follows:

$$f(z) = \begin{cases} \frac{\alpha A}{z} - A + \alpha, z \le -A \\ z, -A < z < A \\ \frac{\alpha A}{z} + A - \alpha, z \ge A \end{cases} \quad (12)$$

the derivative of PIPLA is:

$$\frac{df}{dz} = \begin{cases} -\frac{\alpha A}{z^2}, z \le -A \\ 1, -A < z < A \\ -\frac{\alpha A}{z^2}, z \ge A \end{cases} \quad (13)$$

$$= \begin{cases} 1, -A < z < A \\ -\frac{\alpha A}{z^2}, other \end{cases}$$

where the parameter A is the demarcation point segmenting the PIPLA function into three intervals and also controls the amplitude of PIPLA; α is the slope coefficient.

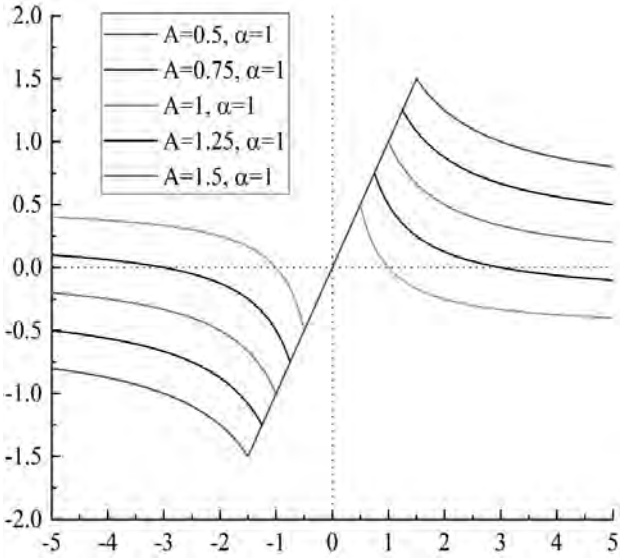
The proposed function combines all the advantages of the commonly used activation function currently. Similar to the ReLU function, PIPLA is segmented into three intervals according to the range of input data, including negative inverse proportional activity region ($-\infty < z \le -A$), linear activity region ($-A < z < A$), and positive inverse proportional activity region ($A \le z < +\infty$). In the linear activity region, PIPLA holds the linear relationship with the symmetric the origin. In the negative and positive inverse proportional activity regions, instead of continuing to use the linear functions, the inverse proportional function of the input data is used, in which the output is

limited to *A* (or -*A*) as the input tends to *A* (or -*A*), and limited to *A* -*α* (or -*A* + *α*) as the input tends to infinity. Similar to the Swish function, PIPLA is a non-monotonic, non-saturation function that effectively avoids the gradient vanishing problem of sigmoid, tanh function, and the dead neurons problem of ReLU function during the training process. Besides, like LiSA, LReLU functions, PIPLA can form various functional models according to the change of demarcation point and the slope coefficient to improve the function flexibility. For example, if the slope coefficient in Eq.12 is set to 1, PIPLA is converted to a function of variable amplitude, as shown in Eq.14. And when the value of the segment point is fixed, such as *A* = 1, PIPLA can also vary with the slope coefficient changed (Eq. 15).

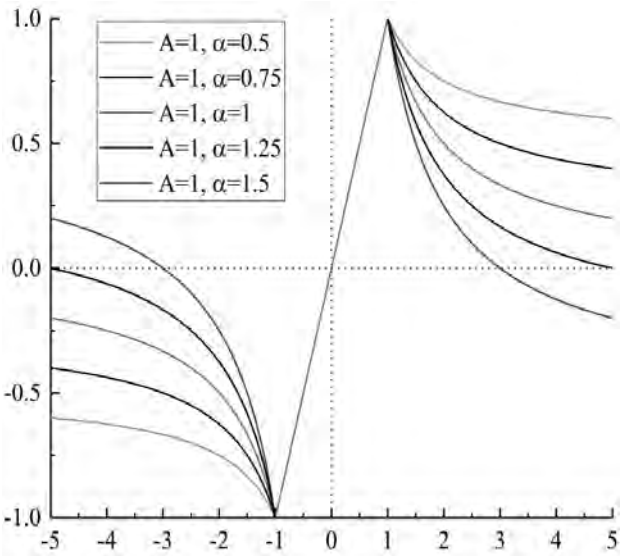
$$f(z) = \begin{cases} \frac{A}{z} - A + 1, z \le -A \\ z, -A < z < A \\ \frac{A}{z} + A - 1, z \ge A \end{cases} \quad (14)$$

$$f(z) = \begin{cases} \frac{\alpha}{z} - 1 + \alpha, z \le -1 \\ z, -1 < z < 1 \\ \frac{\alpha}{z} + 1 - \alpha, z \ge 1 \end{cases} \quad (15)$$

Figure 4 (a, b) shows the curve of the proposed function when changing the input data segment point and slope coefficient, respectively.



(a) α = 1, *A* is changed



(b) *A* =1, α is changed

Figure 4. PIPLA function curves when changing the input data segment point and slope coefficient

On the other hand, Figure 5 shows the derivative curves of the PIPLA function when the slope coefficient is set to 1. For the input is in the range of (-A, A), the derivative is a constant of *A*, and the derivative has a magnitude less than zero for the input is in the other ranges.

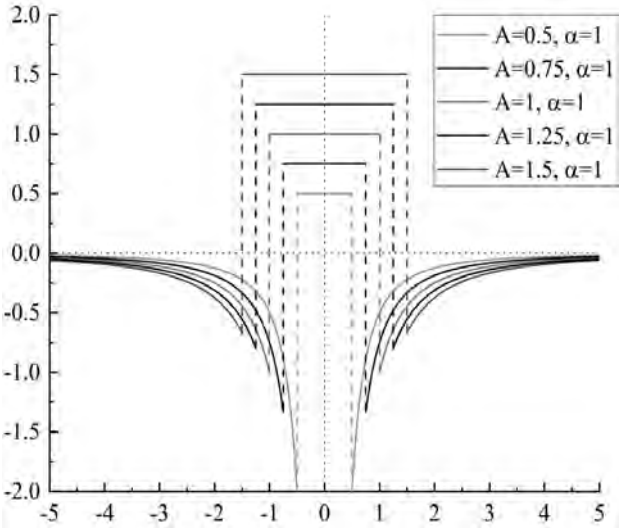


Figure 5. PIPLA derivatives with the slope coefficient are fixed to 1 and the segment point A is changed

In the back-propagation process, the artificial neural network uses the gradient descent method to train the weights and bias parameters of the network. The updating value of the weight is calculated by the following:

$$\frac{dL}{dw_i} = \frac{dL}{df_i} \times \frac{df_i}{dw_i}$$
$$\frac{dL}{df_{i-1}} = \frac{dL}{df_i} \times \frac{df_i}{dz_i} \quad (16)$$

Where w_i denotes the weight matrix, f_i denotes the output after activation at the *i-th* layer, and L denotes the loss function of the neural network. From Eq. 13, the weights updating value is a matrix with positive and negative values. The new weights are calculated by:

$$w_i = w_i - \lambda \frac{dL}{dw_i} \quad (17)$$

Whitefield λ is the learning rate. The new weights can be increased or decreased according to the derivative values of PIPLA, that effectively improves the neural network convergence speed.

3.2. Adaptive inverse proportional linear activation function

In the adaptive inverse balanced linear activation (A-IPLA) function, the slope coefficient field α is trained by using the back-propagation rule for the derivative with respect to. Consequently, the slope coefficient is adjusted during the training process.

The A-IPLA formula is slightly different from the PIPLA function and is defined as:

$$f(z) = \begin{cases} \frac{\alpha^k A}{z^k} - A + \alpha^k, z^k \le -A \\ z^k, -A < z^k < A \\ \frac{\alpha^k A}{z^k} + A - \alpha^k, z^k \ge A \end{cases} \quad (18)$$

Whitefield α^k denotes the channel-wise slope coefficient, z^k denotes the channel-wise activation function input, and k is the channel number of each neural network layer.

Similar to PIPLA, the weights and the slope coefficient field α^k both are trained using the gradient descendent method that they are calculated as follows.

$$\frac{df}{dz} = \begin{cases} -\frac{\alpha^k A}{(z^k)^2}, z^k \le -A \\ 1, -A < z^k < A \\ -\frac{\alpha^k A}{(z^k)^2}, z^k \ge A \end{cases} \quad (19)$$
$$= \begin{cases} 1, -A < z^k < A \\ -\frac{\alpha^k A}{(z^k)^2}, other \end{cases}$$

$$\frac{df}{d\alpha^k} = \begin{cases} \frac{A}{z^k} + 1, z^k \le -A \\ 0, -A < z^k < A \\ \frac{A}{z^k} - 1, z^k \ge A \end{cases} \quad (20)$$

In this paper, the updated slope coefficients of each channel in each layer are set uniformly. Therefore, the average adjustment amount of the slope coefficient is calculated in Eq. 21.

$$d\alpha_{ave}^k = \frac{1}{M} \times \sum_{m=1}^{M} \frac{df}{d\alpha^k} \quad (21)$$

Where M is the neuron number in a layer, and k is the channel number. The updateupdatedu slope coefficient is presented in Eq. 22.

$$\alpha^k = \alpha^k - \beta \times d\alpha_{ave}^k \quad (22)$$

Here, β is the update coefficient and is set to 0.000001 to avoid the data explosion problem.

4. EXPERIMENT AND RESULTS

4.1. Neural network architecture

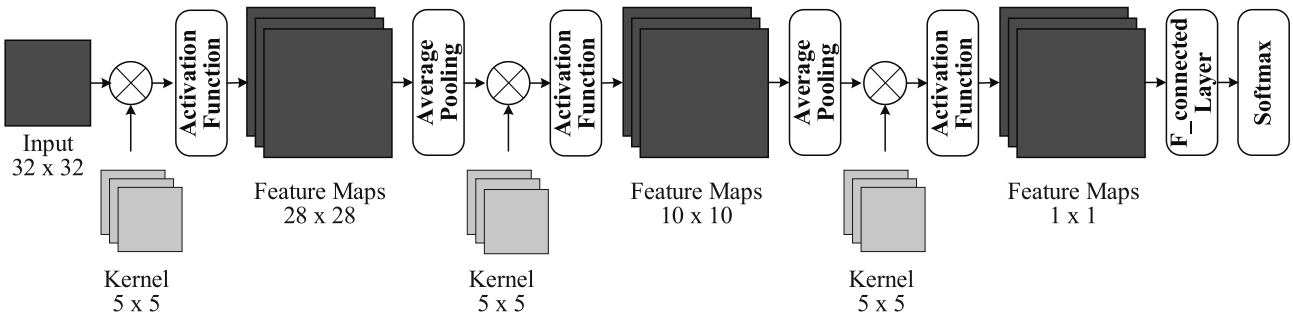


Figure 6. Lenet-5 CNN architecture used in this paper

The convolutional neural network (CNN) used in this paper is similar to Lenet-5 architecture (shown in Figure 6), in which there are three convolutional layers, two pooling layers, and two fully connected layers. The convolutional kernel size is set to 5 x 5 uniformly, and the output maps of the convolutional layers are 6, 16 and 120, respectively. The average pooling method is applied in the pooling layer. Finally, we use the softmax activation function (defined by Eq. 23) in the output layer. Hence, the loss function of CNN is calculated as Eq. 24.

$$Soft\,max(z_i) = \frac{e^{z_i}}{\sum_{n=1}^{N} e^{z_n}} \quad (23)$$

where *N* is the neurons number in the output layer.

$$L_{\text{loss}} = -\sum_{i} y_i \, ln \, f_i \quad (24)$$

Where γ_i is the *i-th* neuron label in the output layer, f_i is the softmax activation function value.

We use this CNN architecture to evaluate the performance of the proposed activation functions, and compared it with other state-of-the-art activation functions such as sigmoid, tanh, Elliot, Rectified Linear Unit (ReLU), Leaky ReLU, Swish, and LiSA on the following benchmark datasets: MNIST and CIFAR-10. Except that the activation function is changed, other hyperparameters such as weight initialization, batch size, training epochs, etc., are the same for all experiments.

4.2. MNIST

MNIST is a grayscale handwritten digital classification dataset with a size of 28 x 28, which contains 60,000 training images and 10,000 testing images, and is divided into ten classes.

Firstly, the PIPLA performance is evaluated with different input data range *A* and slope coefficient. These parameters vary in the range of (0.5 1.5), with a step size is 0.25. All the convolutional layers and fully connected layers use the PIPLA function to activate the neurons. After forty training epochs, the effect of the segment point and slope coefficient selection are shown in tables 1 and 2; in which table 1 shows the effect of changing the segment point value on the network performance when the slope coefficient is fixed to one, and table 2 shows the slope coefficient effect when the segment point is fixed to one.

Table 1. The recognition accuracy with different input data range (*A*) when the slope coefficient (α) fixed to 1 on MNIST

Parameters		Average accuracy (%)	Best accuracy (%)
A	α		
0.5	1	99.12±0.07	99.19
0.75	1	99.17±0.06	99.23
1	1	99.21±0.05	99.26
1.25	1	98.99±0.05	99.04
1.5	1	98.98±0.04	99.02

Table 2. The recognition accuracy with different slope coefficient (α) when the input data range (A) is fixed to 1 on MNIST

Parameters		Average accuracy (%)	Best accuracy (%)
A	α		
1	0.5	99.12±0.06	99.18
1	0.75	99.16±0.03	99.19
1	1	99.21±0.05	99.26
1	1.25	99.19±0.04	99.23
1	1.5	99.13±0.07	99.20

From Tables 1 and 2, we can see that the CNN performance firstly increases with the increase of the input data segment point or slope coefficient value, but after the certain value, the network performance starts to decrease. The Lenet-5 model achieves the highest recognition accuracy of 99.26% after thirty-one training epochs at the value of the input data segment point is 1 and the slope coefficient value is 1.

Secondly, the input data segment point and slope coefficient values that achieving the highest recognition accuracy are used to evaluate the A-IPLA activation function. Table 3 shows the recognition accuracies of the proposed activation function and the state-of-the-art functions after forty training epochs. ReLU function achieves the highest accuracy of 99.12% after thirty-two training epochs among all state-of-the-art functions. While two proposed activation functions have higher recognition accuracies. The PIPLA and A-IPLA functions achieve 99.26% and 99.32% recognition accuracies when the input data range is set to 1, while the slope coefficient values are set to 1 and 1.125, up to 0.15% and 0.20% higher than LiSA function, respectively.

Table 3. The recognition accuracy of the proposed activation function and state-of-the-art functions on MNIST

Activation function	Parameters (A, α)	Best Accuracy (%)	Epoch
Sigmoid	Nil, Nil	98.90	29
Tanh	Nil, Nil	98.89	32
Elloitt	Nil, Nil	98.91	24
ReLU	Nil, Nil	99.12	32
LReLU	Nil, 0.01	99.10	18
Swish	Nil, Nil	99.08	23
LiSA	Nil, 0.175,0.25	99.11	23
PIPLA	1.0, 1.0	99.26	31
A-IPLA	1.0, 1.125	99.32	30

The convergence rate is also one of the important factors to evaluate network performance. In addition to the requirement of high recognition accuracy, a network with good performance also needs to achieve the best accuracy in the shortest training time. Fig. 7 shows the convergence performance of the proposed activation functions and state-of-the-art methods on MNIST dataset. The PIPLA and A-IPLA recognition accuracy improvement rates all are higher than other methods. From table 3 and Fig. 7, we can see that ReLU achieves the highest accuracy after thirty-two training epochs, while the proposed activation functions only need the training epochs of 14 and 9 to achieve the same recognition accuracy as ReLU function that 2.28x and 3.55x time faster than ReLU, respectively.

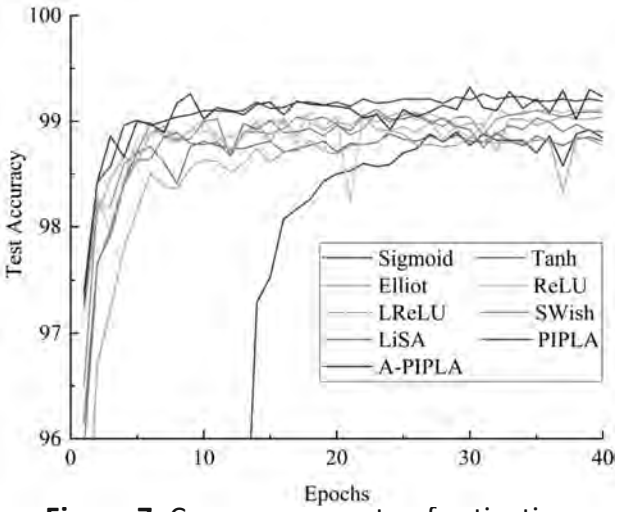


Figure 7. Convergence rate of activation functions on MNIST

4.3. CIFAR-10

CIFAR-10 is a color image dataset with a size of 32 x 32, which contains 50,000 images for training and 10,000 images for validation. These images are divided into ten classes.

Similar to the MNIST dataset, the experiments on the CIFAR-10 dataset are firstly performed to evaluate the performance of the PIPLA function. Then A-IPLA function is tested with the best value of the PIPLA input data segment point and slope coefficient, and compared to the state-of-the-art activation functions. The impacts of the input data segment point and slope coefficient on the CNN performance on CIFAR-10 are shown in tables 4 and 5.

Table 4. The recognition accuracy with different input data range (A) when the slope coefficient (α) is fixed to 1 on CIFAR-10

Parameters		Average accuracy (%)	Best accuracy (%)
A	α		
0.5	1	62.23±1.69	63.92
0.75	1	64.0±1.59	65.59
1	1	61.62±0.62	61.83
1.25	1	59.38±0.84	60.22
1.5	1	59.22±0.79	60.01

Table 5. The recognition accuracy with different slope coefficient (α) when the input data range (A) is fixed to 1 on CIFAR-10

Parameters		Average accuracy (%)	Best accuracy (%)
A	α		
0.75	0.5	63.15±1.53	64.68
0.75	0.75	63.84±1.34	65.18
0.75	1	64.01±1.58	65.59
0.75	1.25	64.41±1.37	65.78
0.75	1.5	63.44±1.53	64.97

From the experimental results in table 6, the proposed function advantages are more obvious compared with the state-of-the-art methods on the CIFAR-10 dataset. After one hundred and twenty training epochs, the proposed activation functions achieve the highest accuracy of 65.78% by using PIPLA and 65.57% by using A-IPLA, improving 5.34% and 5.13% compared with the best result of the state-of-the-art methods (LiSA-60.44%), respectively. The proposed activation functions convergence performance against state-of-the-art functions is shown in Fig. 8. The LiSA function obtains the highest accuracy after one hundred and five epochs. While, the PIPLA obtains the same accuracy after thirty-nine epochs, 2.69x times faster than LiSA. From Fig.8, we can see that although the A-IPLA recognition accuracy is a little worse than the PIPLA function, its convergence is faster than the PIPLA function when it only needs 96 training epochs to obtain the best recognition accuracy.

Table 6. The recognition accuracy of the proposed activation function and state-of-the-art functions on CIFAR-1

Activation function	Parameter (A, α)	Best Accuracy (%)	Epoch
Sigmoid	Nil, Nil	58.40	62
Tanh	Nil, Nil	60.42	110
Elloitt	Nil, Nil	58.79	120
ReLU	Nil, Nil	58.11	92
LReLU	Nil, 0.01	58.23	73
Swish	Nil, Nil	56.68	78
LiSA	Nil, 0.25,0.15	60.44	105
PIPLA	0.75,1.25	65.78	113
A-IPLA	0.75, 1.375	65.57	96

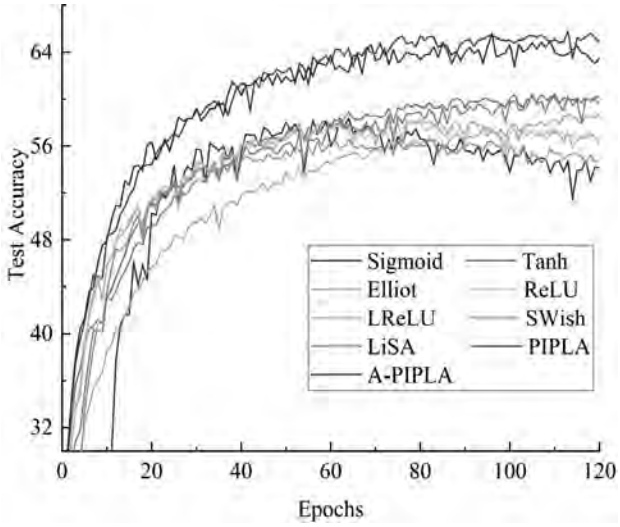


Figure 8. Convergence rate of activation functions on CIFAR-10

5. CONCLUSION

Two variants of activation function called PIPLA, and A-IPLA have been proposed in this paper for internal adjustment in a convolutional neural network to improve the network recognition accuracy and convergence performance. The activation functions are divided into three segments according to the input data, and each segment uses a different activation behavior, including linear activation and inverse proportional activation, in which the P-IPLA function parameters are fixed at the start of design, and the A-IPLA function slope coefficient can be trained by itself. The advantages of these activation functions are that they are symmetric about the origin, bounded, non-saturated, and non-monotonic functions. It helps to avoid output displacement and dead neuron problems, extract better features for the network model, and improve the recognition accuracy and convergence speed. Experiments performed on MNIST, CIFAR-10 datasets using Lenet-5 architecture indicate that proposed activation functions outperform other state-of-the-art activation functions by improving the recognition accuracy of 0.20% and 5.34% on MNIST and CIFAR-10, respectively. While the proposed function convergence rate is also increased significantly with 3.55x faster on MNIST and 2.69x faster on CIFAR-10 when compared to the state-of-the-art activation functions.

REFERENCES

[1] Q. Chen, J. Xu, and V. Koltun, "Fast image processing with fully-convolutional networks," *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2497-2506.

[2] Y.M. Qian, M.X. Bi, T. Tan and K. Yu, "Very Deep Convolutional Neural Networks for Noise Robust Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language*

Processing, vol. 24, pp. 2263-2276, Dec. 2016.

[3] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, pp. 142-158, 2016.

[4] N. Akhtar and A. Mian, "Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey," *IEEE Access*, vol. 6, pp. 14410-14430, 2018.

[5] S.S. Liew, M. Khalil-Hani and R. Bakhteri, "Bounded activation functions for enhanced training stability of deep neural networks on visual pattern recognition problems," *Neurocomputing*, vol. 216, pp. 718-734, 2016.

[6] Ö.F. Ertuğrul, "A novel type of activation function in artificial neural networks: Trained activation function," *Neural Networks*, vol. 99, pp. 148-157, 2018.

[7] C.E. Nwankpa, W. Ijomah, A. Gachagan and S. Marshall, "Activation Functions: Comparison of Trends in Practice and Research for Deep Learning," *ArXiv e-prints*, pp. 1-20, 2018.

[8] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251-257, 1991.

[9] B.L. Kalman and S.C. Kwasny, "Why tanh: choosing a sigmoidal function," *IJCNN International Joint Conference on Neural Networks*. doi:10.1109/ijcnn.1992.227257

[10] X. Wang, Y. Qin, Y. Wang, S. Xiang, and H. Chen, "ReLTanh: An activation function with vanishing gradient resistance for SAE-based DNNs and its application to rotating machinery fault diagnosis," *Neurocomputing, 2019.*

[11] D. Elliot, "A better activation function for artificial neural networks, the National Science Foundation," *Institute for Systems Research, Washington, DC, ISR Technical Rep.* TR-8, 1993.

[12] W. Duch and N. Jankowski, "Survey of neural transfer functions," *Neural Computing Surveys*, vol. 2, pp. 163-212, 1999.

[13] K.V. Naresh Babu, and D.R. Edla, " New Algebraic Activation Function for Multi-Layered Feed Forward Neural Networks," *IETE Journal of Research*, vol. 63(1), pp. 71-79, 2016.

[14] V. Nair and G.E. Hinton, "Rectified linear units improve restricted boltzmann machines," *In Proc.* 27th International Conference on International Conference on Machine Learning, 2010, pp. 807-814.

[15] D.A. Clevert, T. Unterthiner and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *International Conference on Learning Representations*, 2016.

[16] Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. 30 (1), 3.

[17] Qian, S., Liu, H., Liu, C., Wu, S., & Wong, H. S. (2018). Adaptive activation functions in convolutional neural networks. Neurocomputing, 272, 204-212. doi:10.1016/j.neucom.2017.06.070

[18] S.S. Liew, M. Khalil-Hani and R. Bakhteri, "Bounded activation functions for enhanced training stability of deep neural networks on visual pattern recognition problems," *Neurocomputing*, vol. 216, pp. 718-734, 2016.

[19] M. Tanaka, "Weighted Sigmoid Gate Unit for an Activation Function of Deep Neural Network," *Pattern Recognition Letters*, vol. 135, pp. 354-359, 2020.

[20] V. S. Bawa, and V. Kumar, "Linearized si moidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability," *Expert Systems with Applications*, vol. 120, pp. 346-356, 2019.

[21] P. Ramachandran, B. Zoph and V.Q. Le, "Swish: a Self-Gated activation function," *arXiv: Neural and Evolutionary Computing*, 2017.

[22] Hock Hung Chieng, Noorhaniza Wahid, Pauline Ong, Sai Raj Kishore Perla, "Flatten-T Swish: a thresholded ReLU-Swish-like activation function for deep learning," *International Journal of Advances in Intelligent Informatics*, vol 7, pp.76, 2018.

[23] S. Elfwing, E. Uchibe and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks,* 2018.

[24] A.L. Maas and A.Y. Hannun, "Rectifier nonlinearities improve neural network acoustic models," *In Proc. International Conference on Machine Learning*, vol 30(1), pp.3, 2013.

[25] K. He, X. Zhang, S. Ren and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1026-1034.

A-IPLA: Hàm số kích hoạt đối xứng, không đơn điệu cho mạng nơ-ron tích chập

Nguyễn Văn Trường

TÓM TẮT

Trong bài viết này, một hàm kích hoạt đối xứng tâm, không bão hòa và không đơn điệu mới gọi là hàm kích hoạt tuyến tính nghịch đảo thích ứng (A-IPLA) được đề xuất để tránh vấn đề sai lệch ở đầu ra của các nơ-ron, cải thiện được hiệu quả nhận dạng và tốc độ hội tụ của mạng nơ-ron. Hàm này sử dụng phương pháp kích hoạt phân đoạn, đồng thời tùy theo sự thay đổi của điểm phân đoạn và hệ số góc, từng phân đoạn sử dụng các hành vi kích hoạt khác nhau bao gồm kích hoạt tuyến tính và kích hoạt nghịch đảo. Đầu tiên, hàm A-IPLA có ưu điểm về đối xứng tâm giúp tránh được giá trị trung bình đầu ra khác 0, do đó giải quyết được vấn đề dịch chuyển đầu ra. Thứ hai, hệ số góc của A-IPLA có thể tự động cập nhật thông qua quá trình huấn luyện giúp cải thiện tính linh hoạt của hàm kích hoạt. Các thử nghiệm sử dụng mô hình Lenet-5 được thực hiện trên các bộ dữ liệu chuẩn khác nhau (MNIST, CIFAR-10) cho thấy rằng hàm kích hoạt được đề xuất hoạt động tốt hơn các hàm kích hoạt hiện hành trong tất cả các thử nghiệm với mức cải thiện độ nhận dạng chính xác là 0.20% trên MNIST, 5.34% trên CIFAR-10. Đồng thời, trong điều kiện độ nhận dạng chính xác như nhau, tốc độ hội tụ của hàm được đề xuất nhanh hơn 3.55 lần trên MNIST và nhanh hơn 2.69 lần trên CIFAR-10.

Từ khóa: *Hàm kích hoạt, hàm tuyến tính nghịch đảo, kích hoạt không đơn điệu, mạng nơ-ron tích chập*

Received: 14/03/2022
Revised: 05/04/2022
Accepted for publication: 09/05 /2022