Hindawi
Scientific Programming
Volume 2023, Article ID 3873561, 12 pages
https://doi.org/10.1155/2023/3873561

Research Article

A Novel Activation Function of Deep Neural Network

Lin Xiangyang, Qinghua Xing, Zhang Han, and Chen Feng

Air Force Engineering University, Xi'an, Shaanxi 710051, China

Correspondence should be addressed to Lin Xiangyang; 95014052@qq.com

Received 29 August 2022; Revised 30 May 2023; Accepted 4 July 2023; Published 4 August 2023

Academic Editor: Tomàs Margalef

In deep neural networks, the activation function is an important component. The most popular activation functions at the moment are Sigmoid, Sin, rectified linear unit (ReLU), and some variants of ReLU. However, each of them has its own weakness. To improve the network fitting and generalization ability, a new activation function, TSin, is designed. The basic design idea for TSin function is to rotate the Sin function 45° counterclockwise and then finetune it to give it multiple better properties needed as an activation function, such as nonlinearity, global differentiability, unsaturated property, zero-centered property, monotonicity, quasi identity transformation property, and so on. The first is a theoretical derivation of TSin function by formulas. Then three experiments are designed for performance test. The results show that compared with some popular activation functions, TSin has advantages in terms of training stability, convergence speed, and convergence precision. The study of TSin not only provides a new choice of activation function in deep learning but also provides a new idea for activation function design in the future.

1. Introduction

In neural network training, backpropagation (BP) algorithm is a basic algorithm, which was first proposed by Werbos [1] in his doctoral thesis. Later, it was proposed again by Rumelhart et al. [2] and used to solve shallow neural networks. However, due to the problems of "gradient explosion" [3] and "gradient vanishing" [3], the deep neural network (DNN) had a slow development before the 1990s. To solve these problems, Hochreite et al. [4] proposed methods such as long- and short-term memory, residual network [5], data regularization [6], and so on [7–9]. Among them, the improvement of activation function is of milestone significance. Krizhevsky et al. [10] adopted the rectified linear unit (ReLU) as the activation function for the first time and proposed the famous AlexNet, which achieved excellent results in large-scale image evaluation and made a historic breakthrough in the development of deep learning (DL). Later, in order to further enhance the network performance, researchers tried a series of activation functions derived from ReLU, such as the leaky rectified linear unit (LReLU), parametric rectified linear unit (PReLU), self-normalizing linear unit, and so on, which have the property of correction linearity [11–16]. Sitzmann et al. [17] used the Sin function as the activation function and achieved excellent results, starting a new concept of periodic function as the activation function.

Analyzing the existing popular activation function, Sigmoid and Tanh functions have a smooth function curve and definite upper and lower limits, which are very effective at solving classification problems. However, the saturation of the function can cause the "gradient vanishing" problems. The ReLU function can solve the "gradient vanishing" problem by its linear advantages, but there is a "Dead ReLU Problem" [18]. The derivative functions of ReLU overcome the "Dead ReLU Problem," but there are problems such as asymmetry, difficulties on choosing parameters, and so on. The Sin function solves the above problems, but it is not monotonic, and the activation amplitude is limited.

Based on the above background, the characteristics of the better activation function are summarized in this paper after a detailed analysis. Based on the Sin function after counterclockwise rotation by 45°, through theoretical derivation, a new activation function, which integrates several advantages, is proposed step by step. That is the tilted sine function (TSin). As an activation function, the TSin has the following

advantages: (1) it has a linear change trend, (2) it has the periodic change characteristic; (3) it has a zero mean, (4) it has a symmetry about the origin, and (5) it has a derivative that has the characteristics of nonnegative and periodic change. Compared with several kinds of typical activation functions by theoretical analysis, TSin shows better properties. In order to verify the validity of the theory, two typical DL experiments are designed and carried out. The results show that the TSin has different degrees of advantages in the stability, convergence speed, and convergence precision in the DL training.

The remainder of this paper is organized as follows: Section 2 presents the characteristics of several typical activation functions analyzed. Section 3 presents the development principle of TSin activation function is elaborated. Section 4 is the experimental verification of theoretical reasoning. Section 5 is the manuscript conclusion.

2. Classification and Characteristics of Activation Functions

2.1. Requirement of Activation Function Characteristics. The significance of the activation function is to solve the linear indivisibility problem on the premise of no dimension explosion. In the neural networks, each neuron node accepts the output of the upper-layer neurons as its own input. If the input value is directly linear weighted and output to the next layer, it is easy to verify that no matter how many layers the network has, the final output is a linear combination of the original input. Then a large neural network is like a primitive perceptron, and its approximation capability is rather limited.

When a nonlinear activation function is added before output, the whole network will be nonlinear so as to obtain a strong fitting ability. Therefore, in essence, the basic function of the activation function is to introduce nonlinear factors into the neural network so that the neural network can fit various curves. And the most basic requirement of the activation function is nonlinearity.

In addition, the activation function should have the following characteristics as far as possible.

2.1.1. Global Differentiability. In BP, the calculation of the loss function requires the derivative of the activation function. The differentiability ensures the existence of the gradient and further ensures the feasibility of the gradient descent (GD) algorithm in the global region. Of course, for the stochastic gradient descent (SGD) algorithm, the finite nondifferentiable points will not have a great impact on the optimization results [19].

2.1.2. Unsaturation. Saturation refers to the fact that the gradient of the function tends to zero in some interval; as a result, the parameter cannot be updated.

2.1.3. Zero-Center. The zero-center means symmetry and zero mean, which makes the training more stable. The activation function of non-zero mean will lead to the the non-zero mean signals output to the rear layer, namely bias shift [10].

2.1.4. Monotonicity. The sign of derivative is constant. Monotony ensures that the gradient direction of the activation function will not change frequently. That speeds up the convergence rate during training.

2.1.5. Quasi-Identity Transformation. On the premise of nonlinearity activation, the output of the activation function is approximately equal to the input. In this way, the amplitude change between output and input will not be too much, so as to avoid the gradient explosion, enhance the stability of the network and make the gradient back easier.

2.1.6. Less Trainable Parameters. The existence of trainable parameters in the activation function will slow down the training efficiency greatly.

2.1.7. Simple Calculation. Simple calculation of the activation function can accelerate the network training speed and shorten the training time.

2.2. Saturation Activation Functions. Popular saturation activation functions mainly include the Sigmoid function and Tanh function. The shapes of them are shown in Figure 1.

(1) Sigmoid is one of the most classical activation functions [11], whose function and derivative are defined as Equation (1).

$$\begin{aligned} f(x) &= \frac{1}{1+e^{-x}} \\ f'(x) &= f(x)(1-f(x)). \end{aligned} \tag{1}$$

The output range of Sigmoid function is (0, 1), which has an excellent effect on solving logistic regression problems. The function shape is smooth. It is easy to differentiate, and the range of the derivative is (0, 0.5).

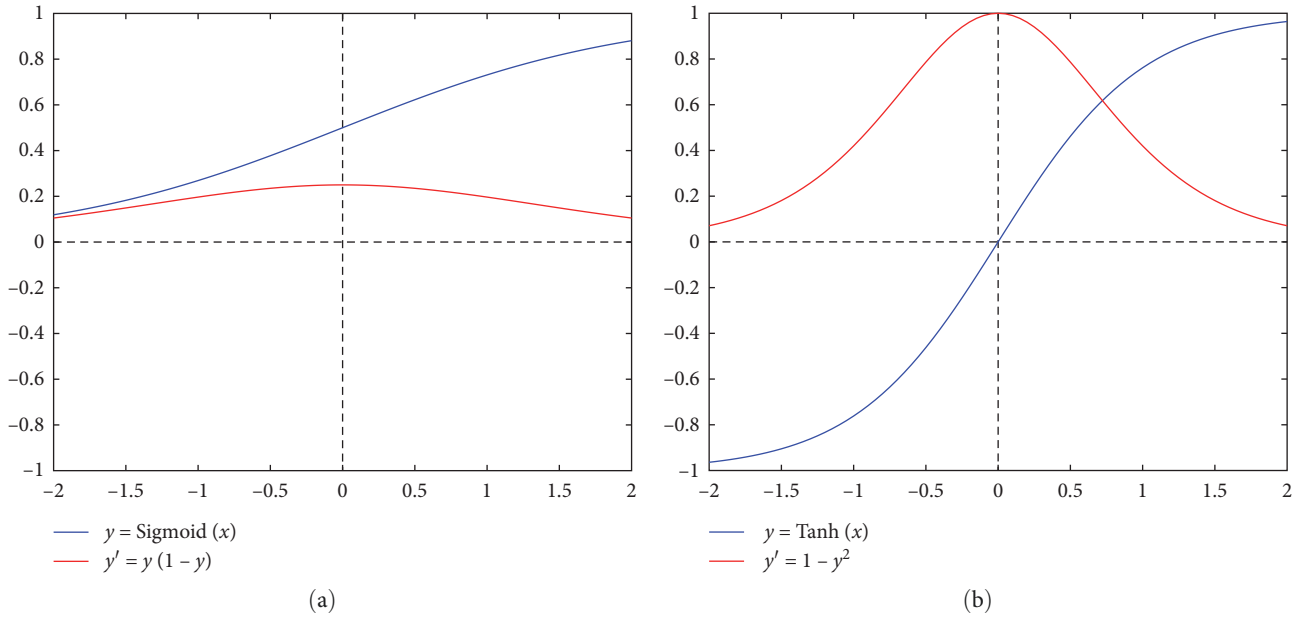
The disadvantage is that it has saturated regions, where the gradient is easy to vanish in BP, so the training of the DNN cannot be completed. Non-zero mean value results in uneven distribution of output signals. Besides, the existence of an exponential function makes the calculation complicated.

(2) Tanh function is improved by Sigmoid, and its function and derivative are defined as Equation (2).

$$\begin{aligned} f(x) &= \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \\ f'(x) &= 1-f^{2}(x). \end{aligned} \tag{2}$$

The Tanh inherits the smooth property of the Sigmoid and overcomes the defect of non-zero mean value by adjusting the output to (−1, 1). The Tanh image is symmetric about the origin, and the range of the derivative function is (0, 1).

2.3. Linear Activation Functions. Linear activation functions are a series of variant functions based on ReLU function. The shapes of them are shown in Figure 2.



(a) (b)

Figure 1: Shapes of typical saturation activation function and derivative. (a) Sigmoid and (b) Tanh.

As the most popular activation function at present, ReLU was first proposed by Nair and Hinton [20], which is defined as Equation (3).

$$f(x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases}. \tag{3}$$

In "Deep Sensity Rectifier Neural Networks," Glorot noted that neurons in the cerebral cortex hardly reach their maximum saturation zone [18].

Compared with the saturation activation function, the most outstanding improvement of ReLU is that there is no saturation area. There is no gradient vanishing phenomenon during network training, which is significant for training the deep network effectively. The output of ReLU is inhibited in the negative interval to achieve the sparsity of the network and reduce the overfitting problem to a certain extent. In addition, the ReLU function has a small amount of computation and easy derivation, so it can train the network quickly.

The most fatal defect of ReLU is that the undifferentiated signal discarding in the negative interval may lead to the permanent death of some neurons, which is the "Dead ReLU problem." Moreover, the non-negative nature of ReLU determines that it does not have zero mean value.

LReLU function is based on ReLU and partial leakage of negative interval signals, which is defined as Equation (4).

$$f(x) = \begin{cases} x & x \geq 0 \\ \alpha x & x < 0 \end{cases}. \tag{4}$$

The α is a fixed parameter in the interval $(0, 1)$. LReLU solved the "Dead ReLU problem," but the activation effect is sensitive to the α value. Besides, the mean of LReLU is also non-zero.

Based on LReLU, PReLU takes α as trainable and obtains the value from data training. PReLU is defined as same as LReLU, namely Equation (4).

Based on LReLU, ELU replaces the activation mode in a negative interval with an exponential function. ELU is defined as Equation (5).

$$f(x) = \begin{cases} x & x \geq 0 \\ \alpha e^x - \alpha & x < 0 \end{cases}. \tag{5}$$

SELU is a variant of ELU, which is defined as Equation (6).

$$f(x) = \lambda \begin{cases} x & x \geq 0 \\ \alpha e^x - \alpha & x < 0 \end{cases}. \tag{6}$$

The α and λ are determined values. The approximate values are $\alpha \approx 1.6733$ and $\lambda \approx 1.0507$, respectively. As a linear activation function, SELU not only solves the "Dead ReLU problem" but also realizes the normalized processing of sample signals.

2.4. Periodic Activation Function. The Sin function was first used as the activation function by Sitzmann et al. [17]. The shapes of function and derivative are shown in Figure 3, and the expressions are shown in Equation (7).

$$\begin{aligned} f(x) &= \sin(x) \\ f'(x) &= \cos(x) \end{aligned}. \tag{7}$$

As a kind of typical periodic function, Sin can be differentiated everywhere, and the derivative is cosine function, which can be regarded as the result of the phase shift of Sin by $\pi/2$. Moreover, Sin has neither the gradient vanishing

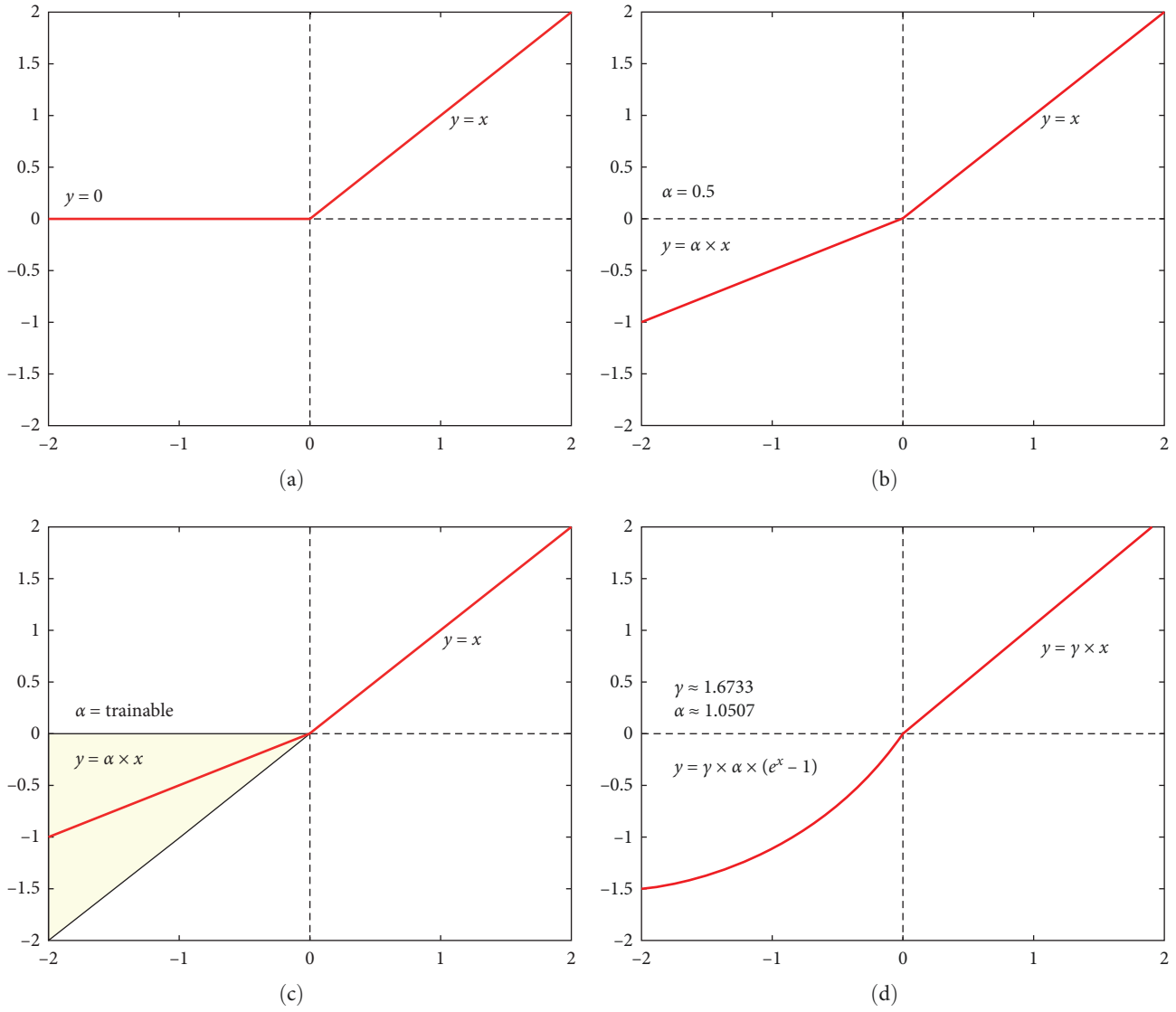


Figure 2: Shapes of typical linear activation function and derivative. (a) ReLU, (b) LreLU, (c) PReLU, and (d) SELU.

problem of the saturated function nor the "Dead ReLU problem" of ReLU. Sin has perfect symmetry about the origin. However, the monotony is destroyed due to the periodic cycle, and the limit of output amplitude makes it impossible to approach the identity transformation.

3. The Derivation and Characteristic Study of TSin Function

In order to make up for the defect of Sin, a new activation function is explored, which inherits the advantages of Sin and remains the monotonicity and quasi-identity transformation. Then it is considered that rotate the Sin function and use the transformed function as the activation function. This new function is named the tilted Sin function. The composition and properties of TSin are discussed in detail.

3.1. Definition of TSin. The basic design idea of TSin is to rotate the Sin function $\pi/4$ reverse firstly. The line which the function extends will change from the x-axis to $y = x$. Next, adjust the parameters of amplitude and phase one by one to achieve the optimal characteristics, and then the TSin is obtained.

Define the original function before rotation as follows:

$$y = A \sin(wx + \varphi)\ (A > 0). \tag{8}$$

According to the symmetry demand, there is $\varphi = 0$. Rotate the original coordinate system $x - y$ as well as the original function counterclockwise as the angle of $\theta = \pi/4$. Define the new rotated coordinate system as coordinate system $a - b$. Then the new rotated function is expressed in coordinate system $a - b$ as follows:

$$b = A \sin(wa + \varphi). \tag{9}$$

The images of the original function and the rotated one are shown in Figure 4. According to the "rotation theorem," there are the following:

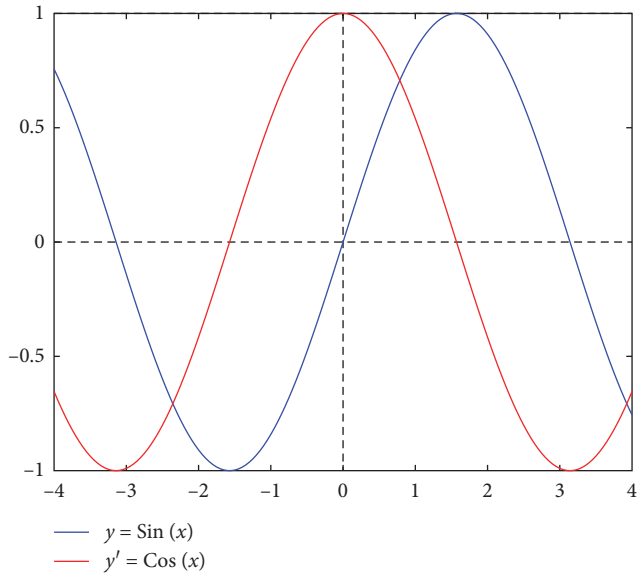


Figure 3: Shapes of the function and derivative of Sin.

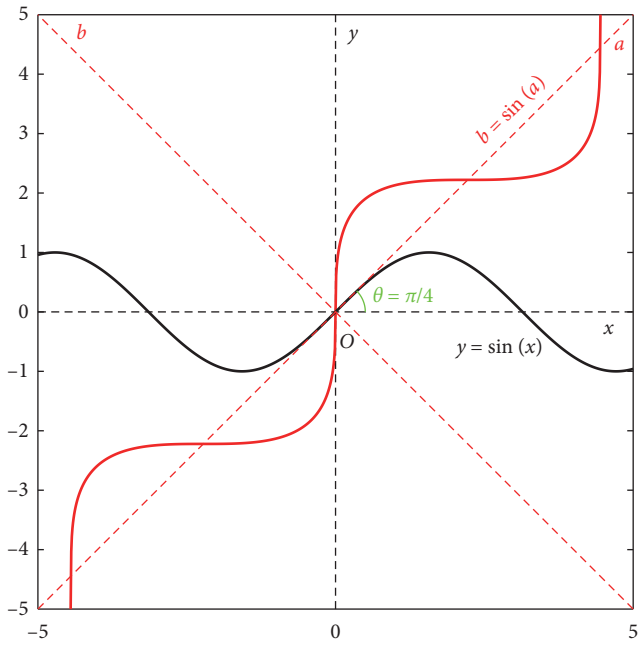


Figure 4: Standard Sin function and the rotated one.

$$\begin{pmatrix} b \\ a \end{pmatrix} = \begin{pmatrix} -\sin\theta & \cos\theta \\ \cos\theta & \sin\theta \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix}. \tag{10}$$

Then, in the original coordinate system $x - y$, the rotated function is as follows:

$$(y\cos\theta - x\sin\theta) = A\sin(\omega \cdot (x\cos\theta + y\sin\theta) + \varphi). \tag{11}$$

Define it as TSin function. This is an implicit function that can be expressed as follows:

$$\begin{aligned} F(x, y) &= (y\cos\theta - x\sin\theta) - A\sin(\omega \cdot (x\cos\theta + y\sin\theta) + \varphi) \\ F(x, y) &= 0. \end{aligned} \tag{12}$$

According to the rule of derivation, there is the following:

$$\begin{aligned} \frac{dy}{dx} &= -\frac{\partial F/\partial x}{\partial F/\partial y} \\ &= \frac{\sin\theta + A\omega\cos\theta \cdot \cos(\omega \cdot (x\cos\theta + y\sin\theta) + \varphi)}{\cos\theta - A\omega\sin\theta \cdot \cos(\omega \cdot (x\cos\theta + y\sin\theta) + \varphi)}. \end{aligned} \tag{13}$$

The derivative dy/dx is a periodic function, and its threshold value is jointly determined by A and ω. According to "Occam's razor," set $\omega = 1$. Substitute $\varphi = 0, \theta = \pi/4$ into Equation (11) and we get the derivative as follows:

$$\frac{dy}{dx} = \frac{1 + A\cos\left((x + y)/\sqrt{2}\right)}{1 - A\cos\left((x + y)/\sqrt{2}\right)}. \tag{14}$$

As shown in Figure 4, the TSin derivative dy/dx reaches a maximum at the origin $(0, 0)$. To satisfy the mapping relation as a function in the coordinate system $x - y$, the following relationship holds:

$$\begin{aligned} \frac{dy}{dx} &= \frac{1 + A}{1 - A} > 0 \\ &\Rightarrow (1 + A)(1 - A) > 0 \\ &\Rightarrow A < 1. \end{aligned} \tag{15}$$

Set A different values in the feasible region (0, 1). The shapes of TSin derivative in different A are as shown, respectively, in Figure 5.

As the figure shows, TSin derivatives in different A have the same periodicity, which is $T = \sqrt{2}\pi$. Just like the Sin, TSin gets the maximum in $kT = \sqrt{2}k\pi\ (k \in Z)$ and gets the minimum in $(k + 0.5)\,T = (2k + 1)\,\pi/\sqrt{2}(k \in Z)$. When $x = 0$ and $x = \pi/\sqrt{2}$, the specific effect of A value on the extreme point is studied. We get the following:

$$\frac{dy}{dx} = \begin{cases} \dfrac{1 + A}{1 - A} & (x = y = 0) \\ \dfrac{1 - A}{1 + A} & \left(x = y = \dfrac{\pi}{\sqrt{2}}\right) \end{cases}. \tag{16}$$

As A adds from 0 to 1, the maximum of derivative increases from 1 to $+\infty$, and the minimum decreases from 1 to 0. The change trends are shown in Figure 6. In fact, when

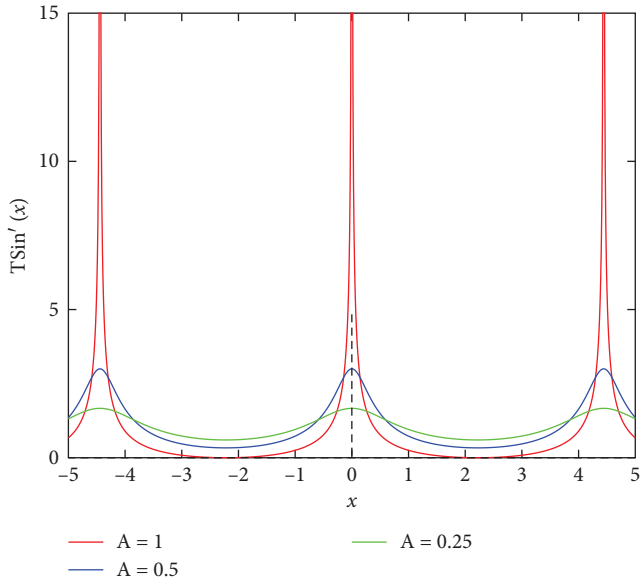


FIGURE 5: The shapes of TSin derivative in different A.

A is set the limiting value 1, TSin is the standard Sin in coordinate system $a - b$, and when A is set the limiting value 0, TSin is the identity function $y = x$. When A is too large, the function gradient changes too fast, and the training is easy to be unstable. On the other hand, when A is too small, the function is too close to linearity, and the convergence speed is too slow. According to the experimental experience, the effect is better when $A = 0.5$.

Summarize the above derivation process. The standard TSin function and derivative are defined as follows:

$$\begin{aligned} F(x, y) &= (y - x) - \sqrt{2}A \sin\left(\frac{\sqrt{2}}{2}(x + y)\right) = 0 \\ y' &= \frac{dy}{dx} = \frac{1 + A\cos\left((x + y)/\sqrt{2}\right)}{1 - A\cos\left((x + y)/\sqrt{2}\right)} \end{aligned} \quad (0 < A < 1). \tag{17}$$

In the actual experiments, it is found that when training the DNN by BP, if the derivative of the activation function is greater than 1 in some regions, then the loss function is easy to cause gradient explosion after multiple backward transmissions. Therefore, generally, the maximum absolute value of the derivative should be not more than 1 in the actual network training. According to $0 < A < 1$ in Equation (15), when $\cos\left((x + y)/\sqrt{2}\right) = 1$, y' gets max value $y'_{\max} = (1 + A)/(1 - A)$ for TSin, and $y'_{\max} > 1$ is constant. According to the generalized incremental rule [21], if the relative amplitude of the derivative is kept constant, the normal training of network parameters will not be affected.

$$\begin{aligned} \omega_{ij} &\leftarrow \omega_{ij} + \alpha\delta_i x_j \\ \delta_i &= \varphi'(v_i)e_i. \end{aligned} \tag{18}$$

So the derivative can be corrected by scaling down its amplitude. The normalized corrected derivative is as follows:

$$\begin{aligned} \hat{y}' &= \frac{dy}{dx} \Big/ \max\left(\frac{dy}{dx}\right) \\ &= \frac{1 - A}{1 + A} \cdot \frac{1 + A\cos\left((x + y)/\sqrt{2}\right)}{1 - A\cos\left((x + y)/\sqrt{2}\right)}. \end{aligned} \tag{19}$$

3.2. Property of TSin. Set $A = 0.5$, and the graph of TSin and related functions are shown in Figure 7. Look at the TSin function with the graph.

First, look at the TSin represented by the blue curve, which has the necessary nonlinearity as an activation function. The curve is singularly symmetric about the origin and has zero mean property. It oscillates periodically along the identity function $y = x$ and intersects the identity function at the fixed periodic points. Observing the derivative curve in cyan, it can be found that not only is TSin differentiable in the whole region, but the derivative oscillates periodically between 1/3 and 3. The constant positive derivative indicates that the TSin has a strict monotony, and there is no saturation region, which completely avoids the possible gradient vanishing during network training.

Finally, observe the normalized modified derivative curve in red. After the maximum value is normalized, the possible gradient explosion is limited. For the TSin, the only parameter is A, which does not require separate training and is generally chosen $A = 0.5$ according to experience.

4. Experiment and Analysis

Above, from the theoretical level, the excellent properties of TSin as activation function are introduced. The following specific verification is carried out through the design experiments. There are two experiments in which a general fully connected DNN and a deep convolutional neural network (CNN) are designed.

4.1. Fully Connected DNN Experiments. A simple digital image recognition experiment is designed as an example of fully connected DNN. The learning aim is to realize the recognitions of 5×5 pixel blocks representing 1–10 digits, as shown in Figure 8.

Each image is transformed into a 5×5 logical matrix and sent to the input node as input. The outputs and labels are 10×1 classification vectors. The structure of the DNN designed is shown in Figure 9. There are five layers of fully connected DNN, including three hidden layers, and each hidden layer contains 20 nodes.

The activation function of the output layer is set as the classification function Softmax. The activation function of the hidden layer is set as TSin, ReLU, LReLU, and Sin, respectively, where the leak parameter of LReLU is $\alpha = 0.5$. In each epoch, 10 simple numbers are sent into the network in turn for training, and 100 epochs of training are completed. The learning algorithm is SGD, the learning rate is 0.01, and no node discarding mechanism is set. Experiments are

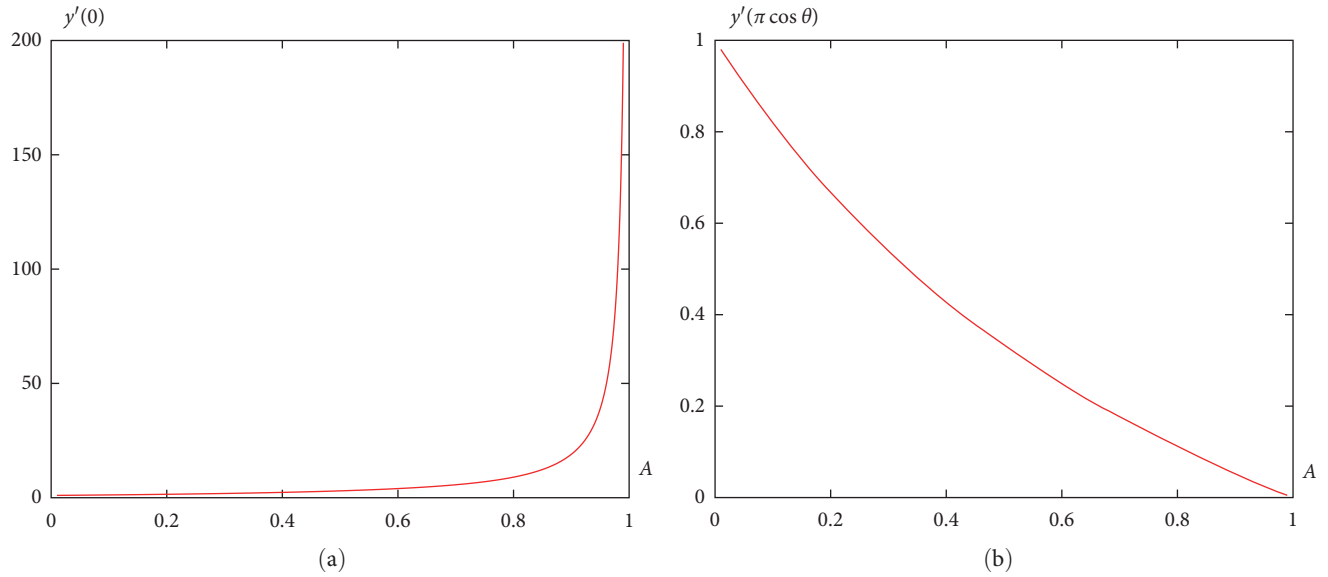


FIGURE 6: The trend of TSin derivative with A *when* $x = 0$ and $x = T/2$ ($T = \sqrt{2}\pi$). (a) The trend of y' (0) with respect to A and (b) The trend of y' (π cos θ) with respect to A.

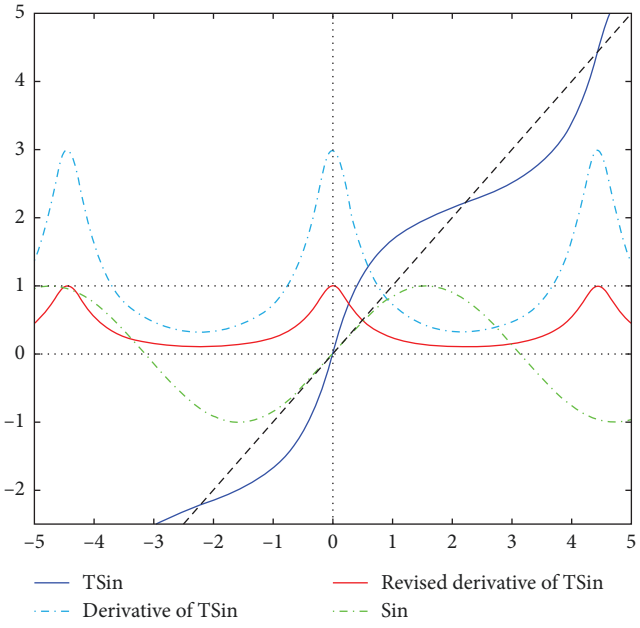


FIGURE 7: TSin and some related functions.

conducted using MATLAB. The hardware used includes Intel Xeon Silver 4210 CPU and NVIDIA TITAN RTX GPU. The variation trend of the mean absolute errors using the different activation functions between 100 epochs is recorded and shown in Figure 10.

As shown in Figure 10, with the increase of training, all curves have a process of decreasing first and then tending to be stable. The changes of the black curve, green curve, and blue curve are similar. They all converge to about −20 dB finally. By comparison, the red curve drops faster and reaches a minimum of −100 dB. It shows that TSin, as an activation function, has far more advantages than other classical activation functions in fully connected DNN. After 85 epochs, it is found that the error comes into an upward trend. According to the law, during the process of DNN training, it is very likely that the training has entered the over-fitting stage since then, which is a normal situation in the training. That also verifies the validity of the experiment in a certain sense.

4.2. CNN Experiments Based on MNIST. The CNN experiment uses the classical MNIST datasets. MNIST contain 70,000 hand-written digital images, each of which is a 28 × 28 pixel black and white image. In order to save training time, the first 5,000 images in MNIST are selected for the experiment. The data used for training and testing are 4,000 and 1,000, respectively, a ratio of 8 : 2.

The each 28 × 28 2D matrix converted from the 28 × 28 pixel image is fed into a simple CNN as input. This network consists of a convolutional layer, a pooling layer, a hidden layer, and an output layer. A schematic diagram of the network structure is made, as shown in Figure 11, for a more intuitive representation of the network.

In detail, the convolutional layer consists of 20 sets of 9 × 9 filters. The convolutional layer output first passes the experimental activation function. Then it is sent to the 2 × 2 average pooling layer. The pooling layer output is sent to a fully connected layer of 100 nodes. After activated by the experimental activation function again, the fully connected layer output is sent to the next fully connected layer of 10 nodes. Finally, the output is classified by Softmax. Table 1 summarizes the detailed information of the network as follows:

> The experimental activation functions are set as TSin, ReLU, LReLU, and Sin. And the CNN is trained 10 epochs with different experimental activation functions respectively by the same training data. In order to improve the

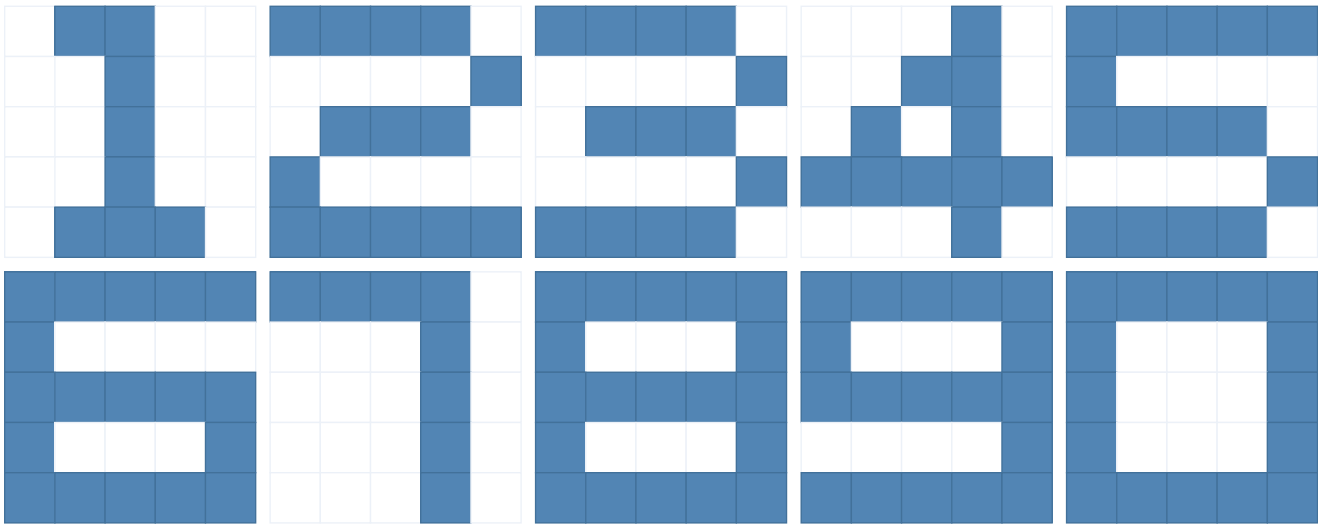


Figure 8: Simple digital images of 1–10.

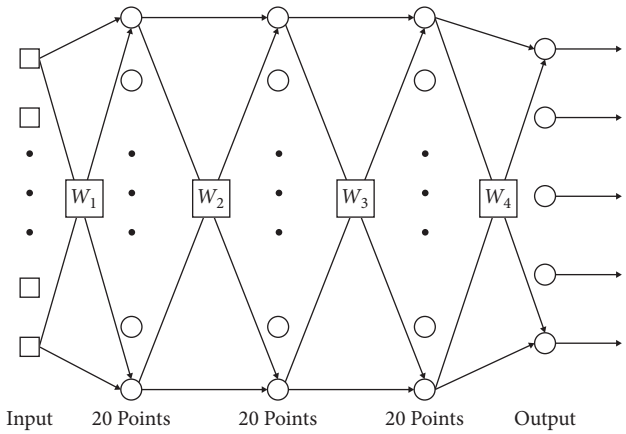


Figure 9: The structure of five layers fully connected DNN.

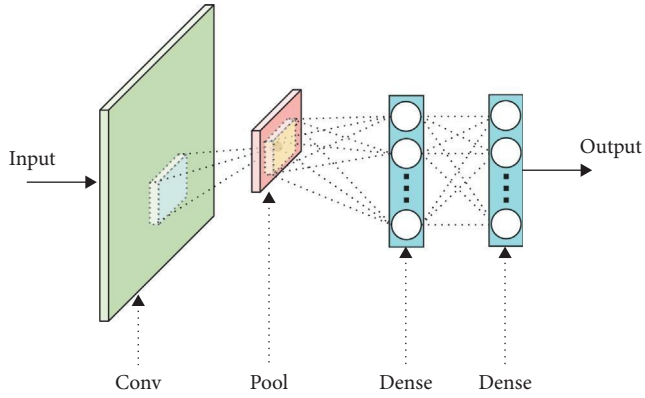


Figure 11: The schematic diagram of CNN for training MNIST.

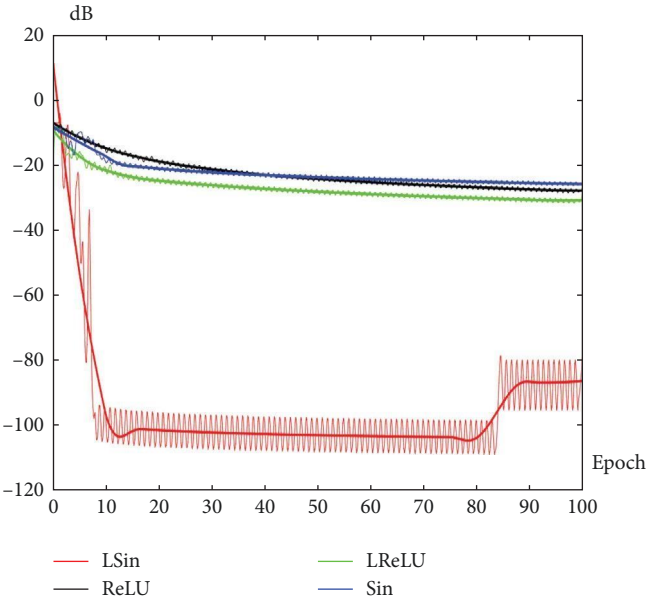


Figure 10: The variation trend of the mean absolute errors using the different activation functions.

training speed as much as possible under the premise of training stability, the learning algorithm is chosen as minibatch learning method, the batch size is 100, and the learning rate is 0.01. At the same time, the momentum method [21] is adopted to adjust the weight to make the training more stable. The momentum hyperparameter is set at 0.95. Experiments are conducted using MATLAB. The hardware used includes Intel Xeon Silver 4210 CPU and NVIDIA TITAN RTX GPU. With the increase of training times, the change of mean absolute errors and accuracy using different activation functions are recorded, as shown in Figure 12.

Figure 12(a) shows the change of the mean absolute of loss value with training using different experimental activation functions. The whole process can be roughly divided into two stages. In the first stage, loss decreases rapidly with training. Then, due to the limitation of network structure, loss gradually tends to converge slowly. By comparing the smoothed curves in the figure, it can be seen that, in the decline stage, the red curve representing the TSin has the fastest decline speed, and the following are the blue curve representing the Sin, the green curve

Table 1: The detailed structure of the simple CNN for training MNIST.

Layer name	Input size	Process	Output size	Parameter size	Activation function
Input layer	None	None	28×28	None	None
Convolutional layer	28×28	9×9 Convolution 20 channels	$20 \times 20 \times 20$	$9 \times 9 \times 20$	Experimental function
Pooling layer	$20 \times 20 \times 20$	2×2 Average pooling	$10 \times 10 \times 20$	None	None
Hidden layer	$10 \times 10 \times 20$	Fully connection	100	$100 \times 2{,}000$	Experimental function
Output layer	100	Fully connection	10	10×100	Softmax

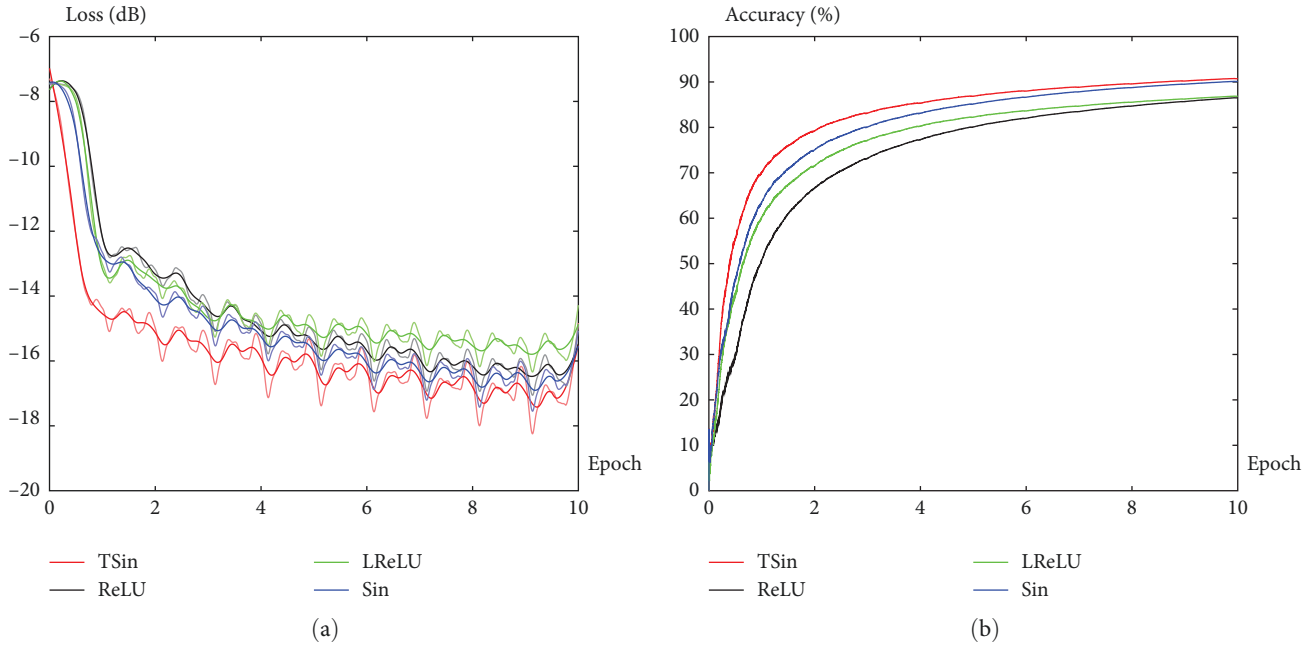


Figure 12: The change of mean absolute errors and accuracy using different activation functions. (a) Loss tendency and (b) accuracy tendency.

representing the LReLU and the black curve representing ReLU, respectively. In the convergence stage, TSin has the smallest convergence value, followed by Sin and ReLU, and L ReLU has the largest convergence value.

Figure 12(b) shows the change of the prediction accuracy in training with the training progress, which is also roughly divided into two stages. In the first stage, the accuracy rate increases rapidly. As the training becomes saturated, the accuracy rate tends to be stable slowly. By comparing the four curves, it can be found that the red curve representing TSin has the fastest rising speed in the rising stage, and the following are the blue curve representing Sin, the green curve representing LReLU, and the black curve representing ReLU. In the stationary stage, the accuracy rate of TSin is in the highest position, which is slightly higher than that of Sin, followed by LReLU and ReLU. Besides, in the data test, the test accuracy rate of TSin is 91.0%, that of SIN is 90.7%, that of ReLU is 90.3%, and that of LReLU is 89.3%. Based on the above, it can be proved that the TSin has certain advantages compared with other typical activation functions in the CNN training.

4.3. CNN Experiments Based on CIFAR-10. The CNN experiment uses the CIFAR-10 datasets. The CIFAR-10 datasets consist of 60,000 images, with a total of 10 categories. Each is a 32×32 pixel RGB color image. In order to save training time, the first 5,000 images in CIFAR-10 are selected for the experiment. The data used for training and testing are 4,000 and 1,000, respectively, a ratio of 8 : 2.

Each image is divided into three channels; each channel is a 32×32 matrix, and each image is converted into a $3 \times 32 \times 32$ 3D matrix as fed into a simple CNN. This network consists of two convolutional layers, two pooling layers, a hidden layer, and an output layer. A schematic diagram of the network structure is made, as shown in Figure 13, for a more intuitive representation of the network.

In detail, the convolutional layers consist of 30 sets of 5×5 filters and 50 sets of 3×3 filters, respectively. After activated by the experimental functions, the data are sent to the 2×2 average pooling layers. The last pooling layer output is sent to a fully connected layer of 100 nodes. After activated by the experimental activation function again, the fully connected layer output is sent to the next fully connected layer of 10 nodes. Finally, the output is classified by Softmax. Table 2

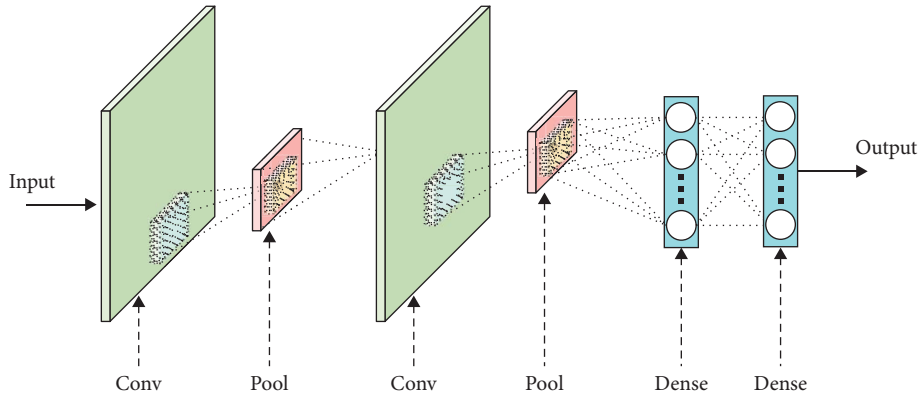


Figure 13: The schematic diagram of a simple CNN for training CIFAR-10.

Table 2: The detailed structure of CNN for training CIFAR-10.

Layer name	Input size	Process	Output size	Parameter size	Activation function
Input layer	None	None	$32 \times 32 \times 3$	None	None
Convolutional layer	$32 \times 32 \times 3$	5×5 Convolution 30 channels	$28 \times 28 \times 30$	$5 \times 5 \times 30$	Experimental function
Pooling layer	$28 \times 28 \times 30$	2×2 Average pooling	$14 \times 14 \times 20$	None	None
Convolutional layer	$14 \times 14 \times 20$	3×3 Convolution 50 channels	$12 \times 12 \times 50$	$3 \times 3 \times 50$	Experimental function
Pooling layer	$12 \times 12 \times 50$	2×2 Average pooling	$6 \times 6 \times 50$	None	None
Hidden layer	$6 \times 6 \times 50$	Fully connection	100	$100 \times 1{,}800$	Experimental function
Output layer	100	Fully connection	10	10×100	Softmax

summarizes the detailed information of the network as follows:

The experimental activation functions are set as TSin, ReLU, LReLU, and Sin. And the CNN is trained 10 epochs with different experimental activation functions respectively by the same training data. The learning algorithm is the same as in Section 4.2, namely, the minibatch learning method. The batch size is 100. The learning rate is 0.01. At the same time, the momentum method [21] is adopted to adjust the weight to make the training more stable. The momentum hyperparameter is set at 0.95. Experiments are conducted using MATLAB, and the hardware used is Intel Core i9 CPU. With the increase of training times, the change of mean absolute errors and accuracy using different activation functions are recorded, as shown in Figure 14.

Figure 14(a) shows the change of the mean absolute of loss value with training using different experimental activation functions. Similar to the experimental results in Section 4.2, the curves also decline rapidly at first and then gradually and slowly tend to converge. Comparing the smoothed curves, it is shown that the red curve representing the TSin has the fastest decline speed in the decline stage. The following is the black curve representing ReLU. In the convergence stage, TSin has the smallest convergence value, followed by ReLU and LReLU, and Sin has the largest convergence value.

Figure 14(b) shows the change of the prediction accuracy in training with the training progress. The early curves experienced a short period of instability. As the training becomes saturated, the accuracy rate tends to be stable slowly. By comparing the four curves, it can be found that the red curve representing TSin has the fastest rising speed, and the following is the blue curve representing Sin, the black curve representing ReLU, and the green curve representing LReLU.

In the data test, the test accuracy rate of TSin is 27.0%, that of Sin is 20.5%, that of ReLU is 22.9%, and that of LReLU is 15.2%. That highlights the advantages of TSin functions.

Based on the above experiments, it can be proved that the TSin has a better activation ability in both the fully connected DNN and the CNN. Compared with the experiments in Section 4, the more activation times in a network, the more obvious the effect will be, which is also in line with the theory in Section 3.

The only note is that the above experiments are proofs based on general experimental datasets in order to highlight the advantages of TSin in general datasets. As for the structure of the network, there may be better specific structures for specific datasets. For example, compared with MNIST datasets, the correct rate of CIFAR-10 datasets is significantly reduced overall. The reason may be that CIFAR-10 are multichannel RGB images, which are more complex than MNIST. Correspondingly, the network structure may need

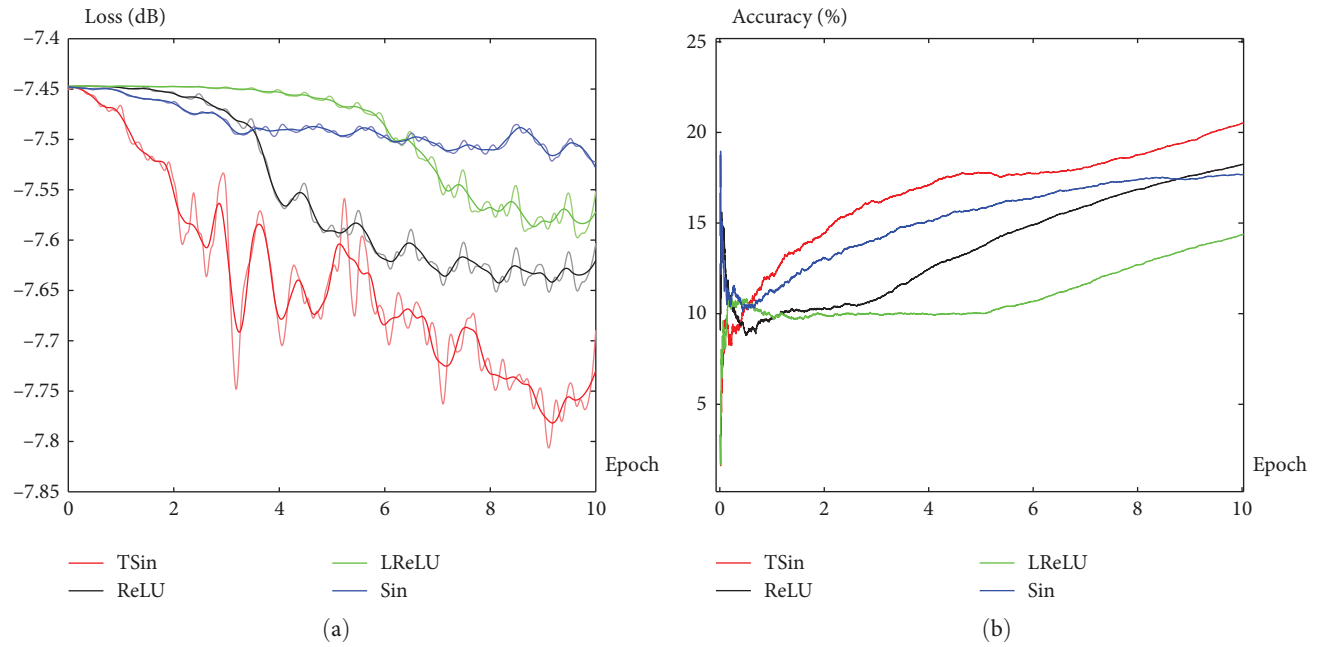


Figure 14: The change of mean absolute errors and accuracy using different activation functions. (a) Loss tendency and (b) accuracy tendency.

to be more optimized, and the training data need to be more. But none of this makes difference to the main proof of the experiment. So it will be gradually explored in later according to the needs and has not been studied much as a focus in this paper.

5. Conclusion

Based on the deficiency of the existing activation functions, a new activation function is proposed. First, through the analysis of classical activation functions, the characteristics of excellent activation functions are summarized. After that, with a design goal of integrating the advantages of the classical activation functions, a new activation function, TSin is obtained through rotating the Sin function and relevant theoretical derivation step by step. The essence of TSin is to rotate anticlockwise the Sin function with certain amplitude and phase by 45°. The TSin has many excellent properties, such as global differentiability, unsaturation, zero-center, monotonic, and quasi-identity transformation. Finally, according to the experience obtained in the actual experiments, the TSin derivative used for training is modified in amplitude to ensure the stability of the training.

After theoretical derivation, the performance tests are carried out through two typical experiments. In the experiment of fully connected DNN, each function quickly reaches the training saturation state. The TSin shows an extremely fast convergence rate and excellent convergence precision, which is far better than that of ReLU, LReLU, and Sin.

In the CNN experiments, because the dataset are more complex, the performance of TSin is not extremely different from the others, but it still maintains the advantage. The three experiment results fully verify the validity of the theory and effectively show that the TSin has better performance than other activation functions in some aspects. Some details in the experiments, such as the appearance of supersaturation, all accord with the normal theory of neural network training, which also reflects the authenticity of the experiments to a certain extent

Combined with the theoretical derivation and experimental verification in this paper, the good performance of TSin as the activation function in ANN has been verified, which not only provides a new choice for the use of the activation function in ANN but also provides a new idea for the innovation of activation function in the future.

Data Availability

In the experiment of Section 4.2, the dataset is MNIST. Readers can access this at http://yann.lecun.com/exdb/mnist/. In the experiment Section 4.3, the dataset is CIFAR-10. Readers can access this at http://www.cs.toronto.edu/~kriz/cifar-10-matlab.tar.gz.

Ethical Approval

This study does not contain any studies with human or animal subjects performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant no. 71771216, grant no. 71701209, and grant no. 72001214.

References

[1] P. Werbos, *Beyond regression: new tools for prediction and analysis in the behavioral sciences*, Ph.D. Thesis, Harvard University, Cambridge, 1974.

[2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[3] S. Hochreiter, *Untersuchungen Zu Dynamischen Neuronalen Netzen*, Technische Universität München, 1991.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," arXiv: 1512.03385 [cs], 2015.

[6] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus, "Regularization of neural network using dropconnect," in *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, pp. 1058–1066, PMLR, 2013.

[7] J. Schmidhuber, "Curious model-building control systems," in *[Proceedings] 1991 IEEE International Joint Conference on Neural Networks*, vol. 2, pp. 1458–1463, IEEE, Singapore, November 1991.

[8] T. Bachlechner, B. P. Majumder, H. H. Mao, G. W. Cottrell, and J. McAuley, "ReZero is all you need: fast convergence at large depth," arXiv: 2003.04887 [cs, stat], 2020.

[9] J. Martens, "Deep learning via Hessian-free optimization," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pp. 735–742, ACM Digital Library, June 2010.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25, pp. 1–9, Curran Associates, Inc., 2012.

[11] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, Stanford University, 2013, https://ai.stanford.edu/~amaas/papers/relu_hybrid_icml2013_final.pdf.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imageNet classification," in *IEEE. 2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, IEEE, Santiago, Chile, December 2015.

[13] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," arXiv: 1505.00853 [cs, stat], 2015.

[14] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponentiallinear units (ELUs)," arXiv: 1511.07289 [cs], 2016.

[15] H. Zhao, F. Liu, L. Li, and C. Luo, "A novel softplus linear unit for deep convolutional neural networks," *Applied Intelligence*, vol. 48, pp. 1707–1720, 2018.

[16] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," arXiv: 1706.02515 [cs, stat], 2017.

[17] V. Sitzmann, J. N. P. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," arXiv: 2006.09661 [cs, eess], 2020.

[18] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323, PMLR, 2011.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, The MIT Press, 2016.

[20] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pp. 807–814, ACM Digital Library, June 2010.

[21] S. Roy and J. J. Shynk, "Analysis of the momentum LMS algorithm," *IEEE Transactions on Acoustics Speech and Signal Processing*, vol. 38, no. 12, pp. 2088–2098, 1990.