Hindawi
Computational Intelligence and Neuroscience
Volume 2018, Article ID 7361628, 10 pages
https://doi.org/10.1155/2018/7361628

Research Article

Fractional-Order Deep Backpropagation Neural Network

Chunhui Bao, Yifei Pu, and Yi Zhang

College of Computer Science, Sichuan University, Chengdu 610065, China

Correspondence should be addressed to Yi Zhang; yizhang.scu@outlook.com

Received 13 March 2018; Accepted 6 June 2018; Published 3 July 2018

Academic Editor: Friedhelm Schwenker

In recent years, the research of artificial neural networks based on fractional calculus has attracted much attention. In this paper, we proposed a fractional-order deep backpropagation (BP) neural network model with L_2 regularization. The proposed network was optimized by the fractional gradient descent method with Caputo derivative. We also illustrated the necessary conditions for the convergence of the proposed network. The influence of L_2 regularization on the convergence was analyzed with the fractional-order variational method. The experiments have been performed on the MNIST dataset to demonstrate that the proposed network was deterministically convergent and can effectively avoid overfitting.

1. Introduction

It is well known that artificial neural networks (ANNs) are the abstraction, simplification, and simulation of the human brains and reflect the basic characteristics of the human brains [1]. In recent years, great progress has been made in the research of deep neural networks. Due to the powerful ability of complex nonlinear mapping, many practical problems have been successfully solved by ANNs in the fields of pattern recognition, intelligent robot, automatic control, prediction, biology, medicine, economics, and other fields [2, 3]. BP neural network is one of the most basic and typical multilayer forward neural networks, which are trained by backpropagation (BP) algorithm. BP, which is an efficient way for optimization of ANNs, was firstly introduced by Werbos in 1974. Then, Rumelhart and McCelland et al. implemented the BP algorithm in detail in 1987 and applied it to the multilayer network version of Minsky [4–6].

The fractional calculus has a history as long as the integral order calculus. In the past three hundred years, the theory of fractional calculus has made great progresses [7–11]. Its basics are differentiation and integration of arbitrary fractional order. Nowadays, fractional calculus is widely used in diffusion processes [12–14], viscoelasticity theory [15], automation control [16–18], signal processing [19–21], image processing [22–25], medical imaging [26–28], neural networks [29–37], and many other fields. Due to the long-term memory, nonlocality, and weak singularity characteristics [29–37], fractional calculus has been successfully applied to ANNs. For instance, Boroomand constructed the Hopfield neural networks based on fractional calculus [37]. Kaslik analyzed the stability of Hopfield neural networks [30]. Pu proposed a fractional steepest descent approach and offered a detailed analysis of its learning conditions, stability, and convergence [38]. Wang applied the fractional steepest descent algorithm to train BP neural networks and proved the monotonicity and convergence of a three-layer example [33]. However, there are three limitations in the proposed fractional-order BP neural network models in [33]. First, the neural network in [33] just had 3 layers, which was actually a shadow network and was not proper to demonstrate its potential for deep learning. Second, the fractional order v of this model was restricted to $(0, 1]$ without reasonable analysis. Third, the loss function did not contain the regularization term, which is an efficient way to avoid overfitting, especially when the training set has small scalar. Overfitting means that the model has high prediction accuracy on training set but has the low prediction accuracy on testing set. This makes the generalization ability of the model poor, and the application value is greatly reduced.

In this paper, we proposed a deep fractional-order BP neural network with L_2 regularization term, and the fractional-order v could be any positive real number. With the fractional-order variational method, the influence of L_2 regularization on the convergence of the proposed model was exploited. The performance of the proposed model was evaluated on the MINST dataset.

The structure of the paper is as follows: in Section 2, the definitions and simple properties of fractional calculus are introduced. In Section 3, the proposed fractional-order multilayer BP neural networks are given in detail. In Section 4, the necessary conditions and the influence of L_2 regularization for the convergence of the proposed BP algorithm are stated. In Section 5, experimental results are presented to illustrate the effectiveness of our model. Finally, the paper is concluded in Section 6.

2. Background Theory for Fractional Calculus

In this section, the basic knowledge of fractional calculus is introduced, including the definitions and several simple properties used in this paper.

Different from integer calculus, fractional derivative does not have a unified temporal definition expression up to now. The commonly used definitions of fractional derivative are Grünwald-Letnikov (G-L), Riemann-Liouville (R-L), and Caputo derivatives [7–11].

The following is the G-L definition of fractional derivative:

$$\begin{aligned} {}_{a}^{G\text{-}L}D_x^v f(x) &\triangleq \lim_{h\longrightarrow 0} h^{-v} \sum_{k=0}^{[(x-a)/h]} \binom{-v}{k} f(x-kh) \\ &\triangleq \lim_{N\longrightarrow\infty} \left\{ \frac{((x-a)/N)^{-v}}{\Gamma(-v)} \right. \\ &\left. \cdot \sum_{k=0}^{N-1} \frac{\Gamma(k-v)}{\Gamma(k+1)} f\left(x-k\left(\frac{x-a}{N}\right)\right)\right\} \end{aligned} \tag{1}$$

where

$$\binom{-v}{k} = \frac{(-v)(-v+1)\dots(-v+k-1)}{k!} \tag{2}$$

${}_{a}^{G\text{-}L}D_x^v$ denotes the fractional differential operator based on G-L definition, $f(x)$ denotes a differintegrable function, v is the fractional order, $[a, x]$ is the domain of $f(x)$, Γ is the Gamma function, and $[\cdot]$ is the rounding function.

The R-L definition of fractional derivative is as follows:

$${}_{a}^{R\text{-}L}D_x^v f(x) = \frac{1}{\Gamma(n-v)} \frac{d^n}{dx^n} \int_a^x \frac{f(y)}{(x-y)^{v-n+1}} dy \tag{3}$$

where ${}_{a}^{R\text{-}L}D_x^v$ denotes the fractional differential operator based on G-L definition; $n = [v + 1]$. Moreover, the G-L fractional derivative can be deduced from the definition of the R-L fractional derivative.

The Caputo definition of fractional derivative is as follows:

$${}_{a}^{C}D_x^v f(x) = \frac{1}{\Gamma(n-v)} \int_a^x (x-y)^{n-v-1} f^{(n)}(y)\, dy \tag{4}$$

where ${}_{a}^{C}D_x^v$ is the fractional differential operator based on Caputo definition, $n = [v + 1]$.

Fractional calculus is more difficult to compute than integer calculus. Several mathematical properties used in this paper are given here. The fractional differential of a linear combination of differintegral functions is as follows:

$$D_x^v(\lambda f(x) + \beta g(x)) = \lambda D_x^v f(x) + \beta D_x^v g(x) \tag{5}$$

where $f(x)$ and $g(x)$ are differintegral functions and λ and β are constants.

The fractional differential of constant function $f(x) = C$, (C is a constant) is different under different definitions:

For the G-L definition,

$$\begin{aligned} {}_{a}^{G\text{-}L}D_x^v f(x) &= \lim_{N\longrightarrow\infty} \left\{ \frac{((x-a)/N)^{-v}}{\Gamma(-v)} \sum_{k=0}^{N-1} \frac{\Gamma(k-v)}{\Gamma(k+1)} C \right\} \\ &= C\frac{(x-a)^{-v}}{\Gamma(1-v)} \end{aligned} \tag{6}$$

For the R-L definition,

$${}_{a}^{R\text{-}L}D_x^v f(x) = C\frac{(x-a)^{-v}}{\Gamma(1-v)}, \quad v > 0 \tag{7}$$

And for the Caputo definition

$${}_{a}^{C}D_x^v f(x) = 0, \quad v > 0 \tag{8}$$

According to (6), (7) and (8), we can know that for the G-L and R-L definition, the fractional differential of constant function is not equal to 0. Only with the Caputo definition, the fractional differential of constant function equals to 0, which is consistent to the integer-order calculus. Therefore, the Caputo definition is widely used in solving engineering problems and it was employed to calculate the fractional-order derivative in this paper. The fractional differential of function $f(x) = (x-a)^p$, $(p > -1)$ is as follows:

$$\frac{d^v (x-a)^p}{dx^v} = \frac{\Gamma(p+1)(x-a)^{p-v}}{\Gamma(p-v+1)} \tag{9}$$

3. Algorithm Description

3.1. Fractional-Order Deep BP Neural Networks. In this section, we introduce the fractional-order deep BP neural network with L layers. n^l, $l = 1, 2, \dots, L$, is the number of neurons for the l-th layer. $W^l = (w_{ji}^l)_{n^{l+1}\times n^l}$ denotes the weight matrix connecting the l-th layer and the $(l+1)$-th layer. f_l denotes the corresponding activation function for the l-th layer. X^j and O^j are the input and the corresponding ideal output of the j-th sample and the training sample set is $\{X^j, O^j\}_{j=1}^J$. $Z^l = (z_1^l, z_2^l \dots, z_{n^{l+1}}^l)$ denotes the total inputs of l-th layer. If neurons in the l-th layer are not connected to any neurons in previous layer, these neurons are called external outputs of the l-th layer, denoted as A_1^l. On the contrary, if neurons in the l-th layer are connected to every neuron in previous layer, these neurons are called internal outputs of l-th layer, denoted as A_2^l. $A^l = (a_1^l, a_2^l \dots, a_{n^l}^l)$ denotes the total

outputs of l-th layer. The forward computing of the fractional-order deep BP neural networks is as follows:

$$A_2^l = f_l\left(Z^l\right) \tag{10}$$

$$A^l = \begin{bmatrix} A_1^l \\ A_2^l \end{bmatrix} \tag{11}$$

$$Z^{l+1} = \mathrm{W}^l \cdot A^l \tag{12}$$

Particularly, external outputs can exist in any layer except the last one. With the square error function, the error corresponding to j-th sample can be denoted as:

$$E_j = \frac{1}{2}\left\|A_j^L - O_j\right\|^2 = \frac{1}{2}\sum_{i=1}^{n^L}\left(a_{ji}^L - o_{ji}\right)^2 \tag{13}$$

where a_{ji}^L denotes the i-th element of A_j^L, o_{ji} denotes the i-th element of O_j.

The total error of the neural networks is defined as

$$\mathrm{E} = \sum_{j=1}^{J} E_j = \frac{1}{2}\sum_{j=1}^{J}\left\|A_j^L - O_j\right\|^2 = \frac{1}{2}\sum_{j=1}^{J}\sum_{i=1}^{n^L}\left(a_{ji}^L - o_{ji}\right)^2. \tag{14}$$

In order to minimize the total error of the fractional-order deep BP neural network, the weights are updated by the fractional gradient descent method with Caputo derivative. Let $i = 1, 2, ..., n^l$. The backpropagation of fractional-order deep BP neural networks can be derived with the following steps.

Firstly, we define that

$$\delta_i^l = \frac{\partial E}{\partial z_i^l}. \tag{15}$$

According to (13), we can know that

$$\delta_i^L = \frac{\partial E}{\partial z_i^L} = \sum_{j=1}^{J}\left(a_{ji}^L - o_{ji}\right) f_L'\left(z_i^L\right). \tag{16}$$

Then the relationship between δ_i^l and δ_i^{l+1} can be given by

$$\begin{aligned}\delta_i^l &= \frac{\partial E}{\partial z_i^l} = \sum_{j=1}^{n^{l+1}}\frac{\partial E}{\partial z_j^{l+1}}\frac{\partial z_j^{l+1}}{\partial z_i^l} = \sum_{j=1}^{n^{l+1}}\delta_j^{l+1}\cdot w_{ji}^l f_l'\left(z_i^l\right) \\ &= f_l'\left(z_i^l\right)\left(\sum_{j=1}^{n^{l+1}}\delta_j^{l+1}\cdot w_{ji}^l\right).\end{aligned} \tag{17}$$

Then, according to the chain rule and (17), we have

$$D_{w_{ji}^l}^{v}\mathrm{E} = \frac{\partial E}{\partial z_j^{l+1}}\cdot D_{w_{ji}^l}^{v} z_j^{l+1} = \delta_j^{l+1}\cdot a_i^l\cdot\frac{\left(w_{ji}^l\right)^{1-v}}{\Gamma(2-v)}. \tag{18}$$

The updating formula is

$$\left(w_{ji}^l\right)^{t+1} = \left(w_{ji}^l\right)^t - \eta D_{w_{ji}^{l\,t}}^{v}\mathrm{E} \tag{19}$$

where $t \in \mathrm{N}$ denotes the t-th iteration and $\eta > 0$ is the learning rate.

3.2. Fractional Deep BP Neural Networks with L_2 Regularization. Fractional-order BP neural network can be overfitted easily when the training set has small scalar. L_2 regularization is a useful way to avoid models to be overfitted without modifying the architecture of network. Therefore, by introducing the L_2 regularization term into the total error, the modified error function can be presented as

$$E_{L2} = E + \frac{\lambda}{2}\|W\|^2 \tag{20}$$

where $\|W\|^2$ denotes the sum of squares of all weights and $\lambda \geq 0$ denotes the regularization parameter.

By introducing (18), we have

$$D_{w_{ji}^l}^{v} E_{L2} = D_{w_{ji}^l}^{v}\mathrm{E} + \lambda\frac{\left(w_{ji}^l\right)^{2-v}}{\Gamma(3-v)}. \tag{21}$$

The updating formula is

$$\left(w_{ji}^l\right)^{t+1} = \left(w_{ji}^l\right)^t - \eta D_{(w_{ji}^l)^t}^{v} E_{L2} \tag{22}$$

where $t \in \mathrm{N}$ denotes the t-th iteration and $\eta > 0$ is the learning rate.

4. Convergence Analysis

In this section, the convergence of the proposed fractional-order BP neural network is analyzed. According to previous studies [39–42], there are four necessary conditions for the convergence of BP neural networks:

(1) The activation functions f_l, $(l = 1, 2, \ldots, L)$ are bounded and infinitely differentiable on R and all of their corresponding derivatives are also continuous and bounded on R. This condition can be easily satisfied because the most common sigmoid activation functions are uniformly bounded on R and infinitely differentiable.

(2) The boundedness of the weight sequence $\{(w_{ji}^l)^t\}$ is valid during training procedure and $\Omega \in R^{\sum_1^{L-1} n^l\cdot n^{l+1}}$ is the domain of all weights with certain boundary.

(3) The learning rate $\eta > 0$ has an upper bound.

(4) Let W denote the weights matrix that consists of all weights and $\phi = \{W \mid D_W^v E_{L2} = 0\}$ be the v-order stationary point set of the error function. One necessary condition is that ϕ is a finite set.

Then, the influence of L_2 regularization on the convergence is derived by using the fractional-order variational method.

According to (20), E_{L2} is defined as a fractional-order multivariable function. The proposed fractional-order BP algorithm is to minimize E_{L2}. Let U denote the fractional-order extreme point of E_{L2} and ξ denotes an admissible point. In addition, U is composed of $U^1, U^2, \ldots, U^{L-1}$ where U^l $(l = 1, 2, \ldots, L-1)$ denotes the weights matrix between the l-th and $(l+1)$-th layer when E_{L2} reaches the extreme value. ξ is composed of $\xi^1, \xi^2, \ldots, \xi^{L-1}$ where ξ^l corresponds to U^l. The initial weights are random values, so the initial points of weights can be represented as $U + (\alpha - 1)\xi$, where α is a

vector that consists of small parameters $\alpha_1, \alpha_2, \ldots, \alpha_{L-1}$, and α_l corresponds to U^l and ξ^l. If $\alpha = 1$, it means $\alpha_l = 1 (l = 1, 2, \ldots, L-1)$, then $U + (\alpha - 1)\xi = U$, and E_{L2} reaches the extreme value. Thus, the process of training the BP neural networks from a random initial weight W to U can be treated as the process of training α with a random initial value to $\alpha = 1$.

The fractional-order derivative of E_{L2} on $U + (\alpha - 1)\xi$ is given as

$$\begin{aligned} D_\alpha^\nu \delta E_{L2}\big|_{\alpha=1} &= D_\alpha^\nu \left[\delta_1(\alpha) + \delta_2(\alpha)\right]\big|_{\alpha=1} \\ &= D_\alpha^\nu E\left(U + (\alpha-1)\xi\right) + D_\alpha^\nu \frac{\lambda}{2} \left\| U + (\alpha-1)\xi \right\|^2 \Big|_{\alpha=1} \\ &= 0 \end{aligned} \tag{23}$$

where ν is the fractional order, which is a positive real number.

From (23), we can see that when $\alpha = 1$, if the ν-order differential of $E(U + (\alpha - 1)\xi)$ with respect to α is existent, $\delta_1(\alpha)$ has a ν-order extreme point and we have

$$D_\alpha^\nu \delta_1(\alpha)\big|_{\alpha=1} = D_\alpha^\nu E\left(U + (\alpha-1)\xi\right)\big|_{\alpha=1} = 0. \tag{24}$$

In this case, the output of each layer in the neural networks is still given by (10) and (11) and the input of each layer is turned into the following:

$$Z^{l+1} = \left(U^l + (\alpha_l - 1)\xi^l\right) \cdot A^l. \tag{25}$$

When $\alpha_l = 1$, we have

$$Z^{l+1} = U^l \cdot A^l. \tag{26}$$

Without loss of generality, according to (18), for the l-th layer of the networks, the ν-order differential of E with respect to α_l can be calculated as

$$\begin{aligned} D_{\alpha_l}^\nu E\left(U^l + (\alpha_l - 1)\xi^l\right)\big|_{\alpha_l=1} &= \frac{\partial E}{\partial Z^{l+1}} \cdot D_{\alpha_l}^\nu Z^{l+1}\Big|_{\alpha_l=1} \\ &= \left(\delta^{l+1} \cdot \left(A^l\right)^T\right) \xi^l \frac{(\alpha_l)^{1-\nu}}{\Gamma(2-\nu)}\Big|_{\alpha_l=1} \\ &= \xi^l \frac{\left(\delta^{l+1} \cdot \left(A^l\right)^T\right)}{\Gamma(2-\nu)} = 0. \end{aligned} \tag{27}$$

where δ^l denotes the column vector $D_{Z^l}^1 E$.

Since the value of ξ is stochastic, according to variation principle [43], to allow (24) to be set up, a necessary condition is that for every layer of the networks

$$\frac{\left(\delta^{l+1} \cdot \left(A^l\right)^T\right)}{\Gamma(2-\nu)} = 0. \tag{28}$$

Secondly, without loss of generality, for $\delta_2(\alpha)$ we have

$$\begin{aligned} D_{\alpha_l}^\nu \delta_2(\alpha_l)\big|_{\alpha_l=1} &= D_{\alpha_l}^\nu \frac{\lambda}{2} \left\| U^l + (\alpha_l - 1)\xi^l \right\|^2 \Big|_{\alpha_l=1} \\ &= \sum \left(\frac{\lambda U_{ji}^l \xi_{ji}^l}{\Gamma(2-\nu)} + \frac{\lambda {\xi_{ji}^l}^2}{\Gamma(3-\nu)} - \frac{\lambda {\xi_{ji}^l}^2}{\Gamma(2-\nu)} \right) \\ &= \frac{\lambda}{\Gamma(2-\nu)\Gamma(3-\nu)} \sum \Big(U_{ji}^l \xi_{ji}^l \Gamma(3-\nu) \\ &\quad + {\xi_{ji}^l}^2 \Gamma(2-\nu) - {\xi_{ji}^l}^2 \Gamma(3-\nu) \Big) = 0 \end{aligned} \tag{29}$$

To allow (29) to be set up, a necessary condition is

$$\frac{\lambda}{\Gamma(2-\nu)\Gamma(3-\nu)} = 0. \tag{30}$$

With (28) and (30), the Euler-Lagrange equation of $D_\alpha^\nu \delta E_{L2}|_{\alpha=1}$ can be written as

$$\frac{\left(\delta^{l+1} \cdot \left(A^l\right)^T\right)}{\Gamma(2-\nu)} + \frac{\lambda}{\Gamma(2-\nu)\Gamma(3-\nu)} = 0. \tag{31}$$

Equation (31) is the necessary condition for the convergence of the proposed fractional-order BP neural networks with L_2 regularization. From (31), we can see that if $\lambda > 0$, then $(\delta^{l+1} \cdot (A^l)^T) \neq 0$. $(\delta^{l+1} \cdot (A^l)^T)$ is the first-order derivative of E in terms of U and can be calculated by U and input sample X. It means that the extreme point U of the proposed algorithm is not equal to the extreme point of integer-order BP algorithm or fractional-order BP algorithm. U changes with the different value of λ and ν. In addition, it is also clear that the regularization parameter λ is bounded since the values of input samples X and weights W are bounded and ν is a constant during the training process.

5. Experiments

In this section, the following simulations were carried out to evaluate the performance of the presented algorithm. The simulations have been performed on the MNIST handwritten digital dataset. Each digit in the dataset is a 28×28 image. Each image is associated with a label from 0 to 9. We divided each image into four parts, which were top-left, bottom-left, bottom-right, and top-right, and each part was a 14×14 matrix. We vectorized each part of the image as a 196×1 vector and each label as a 10×1 vector.

In order to identify the handwritten digits in MNIST dataset, a neural network with 8 layers was proposed. Figure 1 shows the topological structure of the neural networks. For the first four layers of the network, each layer has 196 external neurons and 32 internal neurons. The outputs of the external neurons are in turn four parts of an image and the outputs of the internal neurons of the first layer are 1. The last four layers have no external neurons. The fifth layer, sixth layer, and seventh layer have 64 internal nodes and the output layer has ten nodes. The activation functions of all neurons except

Table 1: Performances of the algorithms when v>2.

Size of training set	$v = 19/9$		$v = 20/9$	
	Train Accuracy	Test Accuracy	Train Accuracy	Test Accuracy
10000	88.65%	83.52%	76.31%	72.66%
20000	91.04%	89.52%	78.93%	75.97%
30000	93.03%	90.65%	82.51%	80.79%
40000	93.20%	90.53%	82.47%	80.61%
50000	93.02%	91.23%	82.53%	81.60%
60000	93.85%	91.71%	87.32%	86.05%

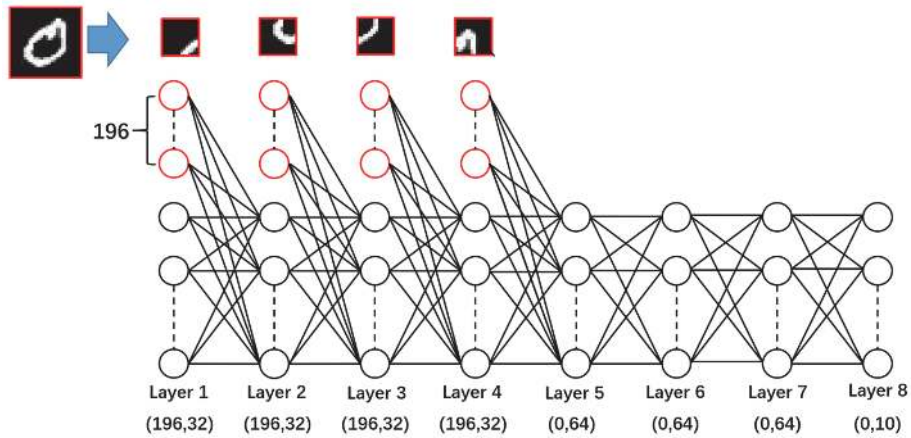

Figure 1: The topological structure of the neural networks.

the first layer are sigmoid functions, which can be given as follows:

$$f(x) = \frac{1}{1 + e^{-x}}. \tag{32}$$

The MNIST dataset has a total number of 60000 training samples and 10000 testing samples. The simulations demonstrate the performance of the proposed fractional-order BP neural network with L_2 regularization, fractional-order BP neural network, traditional BP neural network, and traditional BP neural network with L_2 regularization. To evaluate the robustness of our proposed network for a small set of training samples, we set the number of training samples to be (10000, 20000, 30000, 40000, 50000, and 60000). Different fractional v-order derivatives were employed to compute the gradient of error function, where $v = 1/9, 2/9, 3/9, 4/9, 5/9, 6/9, 7/9, 8/9, 9/9, 10/9, 11/9, 12/9, 13/9, 14/9, 15/9, 16/9, 17/9, 19/9$, and $20/9$ separately ($v = 9/9 = 1$ corresponds to standard integer-order derivative for the common BP; $v \neq 2$ because if $v = 2$ the change of weights after each iteration is 0, and the weights of the neural networks cannot be updated). The learning rate was set to be 3 and the batch size was set to be 100. The number of epochs n was 300. Two main metrics—training accuracy and testing accuracy—were used to measure the performance of the results from different networks. Each network was trained 5 times and the average values were calculated.

In order to explore the relationship between the fractional orders and the neural network performance, the fractional-order neural networks with different orders were trained. Figure 2 shows the results of different networks with different sizes of training set. We can find that when the fractional order v exceeds 1.6, both the training and testing accuracies declined rapidly, and when the fractional order $v > 2$, the performances of the fractional BP neural networks were much poorer than that with $0 < v < 2$. The results of $v = 19/9$ and $20/9$ were shown in Table 1 as examples. This result is consistent with that for describing physical problems, and usually the limitation $0 < v < 2$ is adopted in the fractional-order models.

From Figure 2, it can be observed that, with the increase of the size of training set, the performances of the networks were improved visibly. Furthermore, it is also obvious that the training and testing accuracies raised gradually with increasing fractional orders and then reached the peak while v equaled 10/9 or 11/9 order. After that, the training and testing accuracies began to decline rapidly.

Table 2 shows the optimal orders under training set and testing set separately with different size of training set and it can be noticed that the optimal orders almost concentrated in 10/9 and 11/9. The only exception is that when the number of training samples was 50000, the training accuracy of order 1 was slightly higher than that in 10/9 or 11/9 order case. Generally, for the MNIST dataset the performances of fractional-order BP neural networks are better than integer order.

Table 2: Optimal Orders and Highest Accuracies.

Size of training set	Optimal order of training set	Optimal order of testing set	Highest training accuracy	Highest testing accuracy
10000	10/9	11/9	98.53%	90.31%
20000	10/9	10/9	98.84%	92.34%
30000	11/9	11/9	99.05%	93.50%
40000	10/9	11/9	99.18%	93.92%
50000	1	10/9	99.20%	94.56%
60000	11/9	11/9	99.20%	95.00%

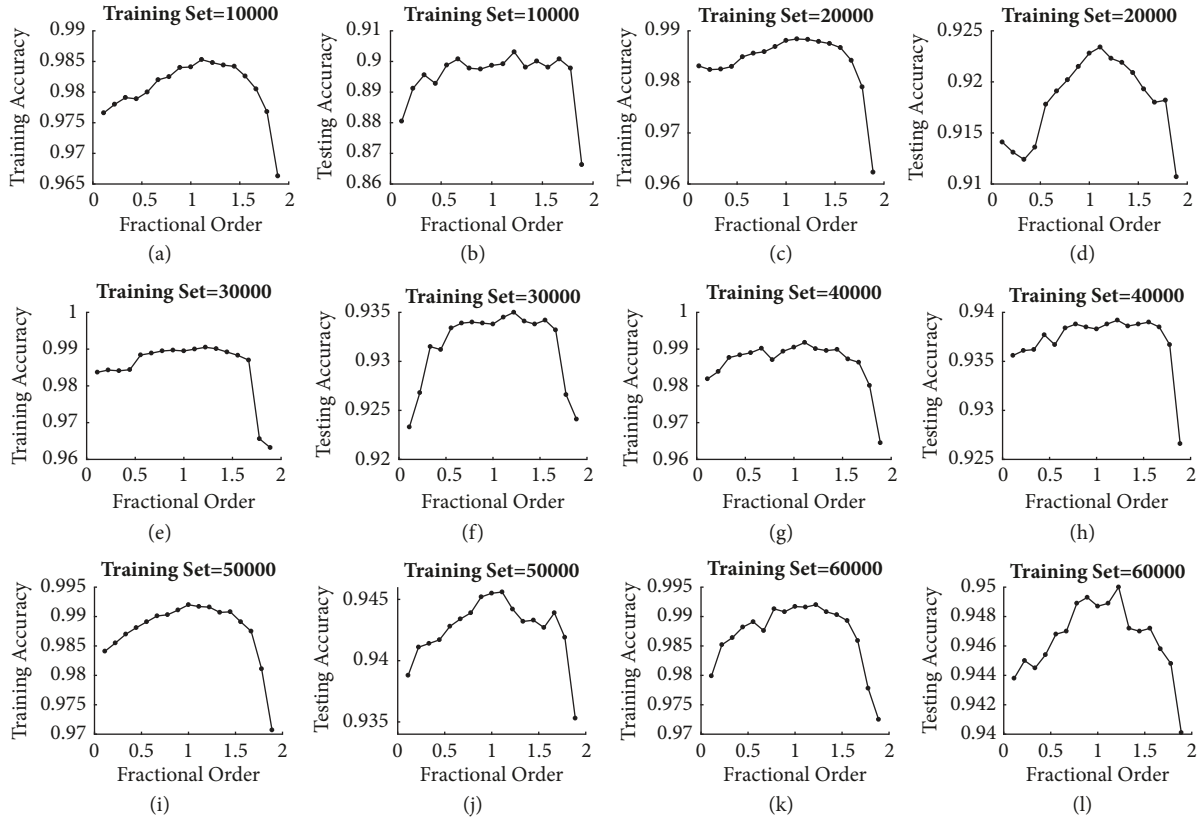

Figure 2: The relationship between the fractional order of gradient descent method and the neural network performance.

It also can be seen that, in each case, the training accuracy is much bigger than testing accuracy, which means that the BP neural networks have obvious overfitting phenomenon. To avoid overfitting, the integer-order and fractional-order BP neural networks with L_2 regularization were trained. With different sizes of training set we chose the regularization parameter λ to be (2×10^{-5}, 1×10^{-5}, 5×10^{-6}, 5×10^{-6}, 5×10^{-6}, and 3×10^{-6}). For the fractional-order neural networks, we chose the fractional order v that had highest testing accuracy in previous simulations. When the numbers of training samples were (10000, 20000, 30000, 40000, 50000, and 60000), we separately set the fractional order v to be (11/9, 10/9, 11/9, 11/9, 10/9, 11/9).

The performance of the proposed fractional-order BP neural networks with L_2 regularization and the performance comparison with integer-order BP neural networks (IOBP), integer-order BP neural networks with L_2 regularization, and fractional-order BP neural networks (FOBP) in terms of training and testing accuracy are shown in Table 3 and the change of the testing accuracy with the iterations was given in Figure 3

In Table 3 and Figure 3, it can be seen that, after the addition of L_2 regularization to BP neural networks, the training accuracy is slightly decreased but the testing accuracy significantly increased, which indicated that adding L_2 regularization can effectively suppress overfitting and improve the generalization of BP neural networks. Furthermore, it can be noticed that after adding L_2 regularization the performance of fractional-order BP neural network is better than integer order. One important merit of the L_2 regularization is that it gained more benefit while the training set is small. The most possible reason is that the network trained with the smallest number of training samples was affected most by the overfitting. With the increase of the training samples, the model gradually changed from overfitting to underfitting, so the improvement of the regularization method became faint.

Then, the stability and convergence of the proposed fractional-order BP neural networks with L_2 regularization

Table 3: Performance comparison of different type BP neural networks.

Size of training set	Integer-order BP neural networks		Fractional-order BP neural networks		Integer-order BP neural networks with L_2 regularization		Fractional-order BP neural networks with L_2 regularization			
	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy	Improvement relative to IOBP	Improvement relative to FOBP
10000	98.41%	89.87%	98.48%	90.31%	98.45%	93.35%	98.43%	93.95%	4.54%	4.03%
20000	98.81%	92.28%	98.84%	92.34%	98.75%	95.09%	98.79%	95.13%	3.09%	3.02%
30000	98.95%	93.38%	99.05%	93.50%	98.92%	95.15%	98.88%	95.62%	2.40%	2.27%
40000	99.05%	93.83%	99.01%	93.92%	98.96%	95.63%	98.95%	95.83%	2.13%	2.03%
50000	99.20%	94.55%	99.17%	94.56%	99.11%	96.08%	99.15%	96.45%	2.01%	2.00%
60000	99.17%	94.87%	99.20%	95.00%	99.13%	96.51%	99.17%	96.70%	1.93%	1.79%

We use the following formula to calculate improvement: improvement of A compared with B = (A-B)÷B.

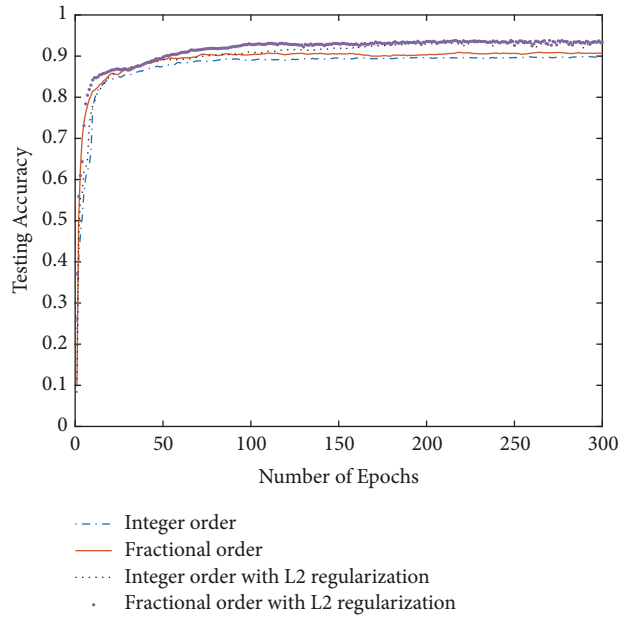

Figure 3: Performance comparison in terms of testing accuracy.

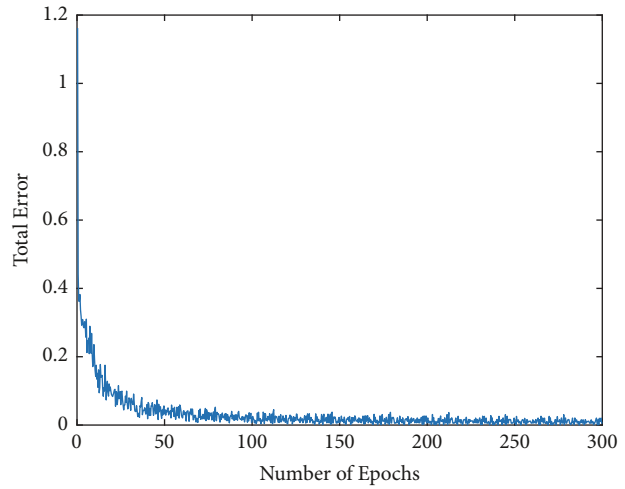

Figure 4: Changes of total error E_{L2} during the training process.

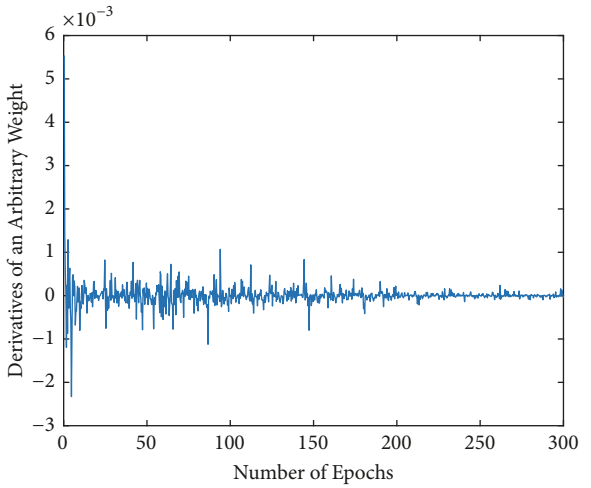

Figure 5: Changes of $D^{v}_{w^{5}_{20,20}} E_{L2}$ during the training process.

are demonstrated in Figures 4 and 5. We used the network with optimal order, which means that the size of training set was 60000, fractional-order v was 11/9, and the regularization parameter λ was 3×10^{-6}. Figure 4 shows the change of the total error E_{L2} during the training process. Without loss of generality, the change of $D^{v}_{w^{5}_{20,20}} E_{L2}$ was randomly selected and Figure 5 shows the change of it during the training process. It is clear to see that E_{L2} and $D^{v}_{w} E_{L2}$ converged fast and stably and were finally close to zero. These observations effectively verify the proposed algorithm is deterministically convergent.

6. Conclusion

In this paper, we applied fractional calculus and regularization method to deep BP neural networks. Different from previous studies, the proposed model had no limitations on the number of layers and the fractional-order was extended to arbitrary real number bigger than 0. L_2 regularization was also imposed into the errorless function. Meanwhile, we analyzed the profits introduced by the L_2 regularization on the convergence of this proposed fractional-order BP network. The numerical results support that the fractional-order BP neural networks with L_2 regularization are deterministically convergent and can effectively avoid the overfitting phenomenon. Then, how to apply fractional calculus to other more complex artificial neural networks is an attracted topic in our future work.

Data Availability

The code of this work can be downloaded at https://github.com/BaoChunhui/Deep-fractional-BP-neural-networks.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key R&D Program of China under Grant 2017YFB0802300, the National Natural Science Foundation of China under Grant 61671312, the Science and Technology Project of Sichuan Province of China under Grant 2018HH0070, and the Strategic Cooperation Project of Sichuan University and Luzhou City under Grant 2015CDLZ-G22.

References

[1] M. Kubat, "Neural networks: a comprehensive foundation by Simon Haykin, Macmillan, 1994, ISBN 0-02-352781-7," *The Knowledge Engineering Review*, vol. 13, no. 4, pp. 409–412.

[2] S. A. Kalogirou, "Applications of artificial neural networks in energy systems: a review," *Energy Conversion and Management*, vol. 40, no. 10, pp. 1073–1087, 1999.

[3] H. B. Demuth, M. H. Beale, O. De Jess, and M. T. Hagan, *Neural network design: Martin Hagan*, 2014.

[4] D. E. Rumelhart, J. L. McClelland, and P. R. Group, *Parallel Distributed Processing*, vol. 1, MIT press, Cambridge, MA, USA, 1987.

[5] J. Jia and H. Duan, "Automatic target recognition system for unmanned aerial vehicle via backpropagation artificial neural network," *Aircraft Engineering and Aerospace Technology*, vol. 89, no. 1, pp. 145–154, 2017.

[6] Z. Wu and H. Wang, *Super-resolution Reconstruction of SAR Image based on Non-Local Means Denoising Combined with BP Neural Network*, 2016, arXiv preprint arXiv:1612.04755.

[7] E. R. Love, "Fractional derivatives of imaginary order," *Journal Of The London Mathematical Society-Second Series*, vol. 3, pp. 241–259, 1971.

[8] Y. Povstenko, *Linear Fractional Diffusion-Wave Equation for Scientists and Engineers*, Birkhäuser, New York, 2015.

[9] A. McBride and G. Roach, *Fractional Calculus (Pitman Research Notes in Mathematics, No 138)*, Longman Science & Technology, 1986.

[10] K. Nishimoto, *Fractional Calculus: Integrations and Differentiations of Arbitrary Order*, University of New Haven Press, New Haven, Conn, USA, 1989.

[11] I. Podlubny, *Fractional differential equations: an introduction to fractional derivatives, fractional differential equations, to methods of their solution and some of their applications*, vol. 198, Academic press, 1998.

[12] N. Özdemir and D. Karadeniz, "Fractional diffusion-wave problem in cylindrical coordinates," *Physics Letters A*, vol. 372, no. 38, pp. 5968–5972, 2008.

[13] N. Özdemir, O. P. Agrawal, D. Karadeniz, and B. B. İskender, "Analysis of an axis-symmetric fractional diffusion-wave problem," *Journal of Physics A: Mathematical and Theoretical*, vol. 42, no. 35, p. 355208, 2009.

[14] Y. Povstenko, "Solutions to the fractional diffusion-wave equation in a wedge," *Fractional Calculus and Applied Analysis An International Journal for Theory and Applications*, vol. 17, no. 1, pp. 122–135, 2014.

[15] R. L. Bagley and P. J. Torvik, "A theoretical basis for the application of fractional calilus to visoelasticity," *Journal of Rheology*, vol. 27, no. 3, pp. 201–210, 1983.

[16] D. Baleanu, J. A. T. Machado, and A. C. J. Luo, *Fractional Dynamics and Control*, Springer, New York, NY, USA, 2012.

[17] C. Li and G. Chen, "Chaos in the fractional order Chen system and its control," *Chaos, Solitons $ Fractals*, vol. 22, pp. 549–554, 2004.

[18] C. A. Monje, B. M. Vinagre, V. Feliu, and Y. Chen, "Tuning and auto-tuning of fractional order controllers for industry applications," *Control Engineering Practice*, vol. 16, no. 7, pp. 798–812, 2008.

[19] L. B. Almeida, "Fractional fourier transform and time-frequency representations," *IEEE Transactions on Signal Processing*, vol. 42, no. 11, pp. 3084–3091, 1994.

[20] M. S. Aslam and M. A. Z. Raja, "A new adaptive strategy to improve online secondary path modeling in active noise control systems using fractional signal processing approach," *Signal Processing*, vol. 107, pp. 433–443, 2015.

[21] R. Panda and M. Dash, "Fractional generalized splines and signal processing," *Signal Processing*, vol. 86, no. 9, pp. 2340–2350, 2006.

[22] M. Xu, J. Yang, D. Zhao, and H. Zhao, "An image-enhancement method based on variable-order fractional differential operators," *Bio-Medical Materials and Engineering*, vol. 26, pp. S1325–S1333, 2015.

[23] Y.-F. Pu, N. Zhang, Y. Zhang, and J.-L. Zhou, "A texture image denoising approach based on fractional developmental mathematics," *PAA. Pattern Analysis and Applications*, vol. 19, no. 2, pp. 427–445, 2016.

[24] Y.-F. Pu, J.-L. Zhou, and X. Yuan, "Fractional differential mask: a fractional differential-based approach for multiscale texture enhancement," *IEEE Transactions on Image Processing*, vol. 19, no. 2, pp. 491–511, 2010.

[25] J. Bai and X.-C. Feng, "Fractional-order anisotropic diffusion for image denoising," *IEEE Transactions on Image Processing*, vol. 16, no. 10, pp. 2492–2502, 2007.

[26] Y. Zhang, W. Zhang, Y. Lei, and J. Zhou, "Few-view image reconstruction with fractional-order total variation," *Journal of the Optical Society of America A: Optics, Image Science & Vision*, vol. 31, no. 5, pp. 981–995, 2014.

[27] Y. Zhang, Y. Wang, W. Zhang, F. Lin, Y. Pu, and J. Zhou, "Statistical iterative reconstruction using adaptive fractional order regularization," *Biomedical Optics Express*, vol. 7, no. 3, pp. 1015–1029, 2016.

[28] Y. Zhang, Y.-F. Pu, J.-R. Hu, Y. Liu, Q.-L. Chen, and J.-L. Zhou, "Efficient CT metal artifact reduction based on fractional-order curvature diffusion," *Computational and Mathematical Methods in Medicine*, Art. ID 173748, 9 pages, 2011.

[29] Y.-F. Pu, Z. Yi, and J.-L. Zhou, "Defense Against Chip Cloning Attacks Based on Fractional Hopfield Neural Networks," *International Journal of Neural Systems*, vol. 27, no. 4, Article ID 1750003, 2017.

[30] E. Kaslik and S. Sivasundaram, "Dynamics of fractional-order neural networks," in *Proceedings of the 2011 International Joint Conference on Neural Network, IJCNN 2011*, pp. 611–618, usa, August 2011.

[31] Y.-F. Pu, Z. Yi, and J.-L. Zhou, "Fractional Hopfield neural networks: fractional dynamic associative recurrent neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2319–2333, 2017.

[32] C. Song and J. Cao, "Dynamics in fractional-order neural networks," *Neurocomputing*, vol. 142, pp. 494–498, 2014.

[33] J. Wang, Y. Wen, Y. Gou, Z. Ye, and H. Chen, "Fractional-order gradient descent learning of BP neural networks with Caputo derivative," *Neural Networks*, vol. 89, pp. 19–30, 2017.

[34] R. Rakkiyappan, R. Sivaranjani, G. Velmurugan, and J. Cao, "Analysis of global O (t- a) stability and global asymptotical periodicity for a class of fractional-order complex-valued neural networks with time varying delays," *Neural Networks*, vol. 77, pp. 51–69, 2016.

[35] H. Wang, Y. Yu, and G. Wen, "Stability analysis of fractional-order Hopfield neural networks with time delays," *Neural Networks*, vol. 55, pp. 98–109, 2014.

[36] H. Wang, Y. Yu, G. Wen, S. Zhang, and J. Yu, "Global stability analysis of fractional-order Hopfield neural networks with time delay," *Neurocomputing*, vol. 154, pp. 15–23, 2015.

[37] A. Boroomand and M. B. Menhaj, "Fractional-Order Hopfield Neural Networks," in *Advances in Neuro-Information Processing*, vol. 5506 of *Lecture Notes in Computer Science*, pp. 883–890, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[38] Y.-F. Pu, J.-L. Zhou, Y. Zhang, N. Zhang, G. Huang, and P. Siarry, "Fractional extreme value adaptive training method: fractional steepest descent approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 4, pp. 653–662, 2015.

[39] H. Shao and G. Zheng, "Boundedness and convergence of online gradient method with penalty and momentum," *Neurocomputing*, vol. 74, no. 5, pp. 765–770, 2011.

[40] W. Wu, G. Feng, Z. Li, and Y. Xu, "Deterministic convergence of an online gradient method for BP neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 16, no. 3, pp. 533–540, 2005.

[41] W. Wu, J. Wang, M. Cheng, and Z. Li, "Convergence analysis of online gradient method for BP neural networks," *Neural Networks*, vol. 24, no. 1, pp. 91–98, 2011.

[42] H. Zhang, W. Wu, F. Liu, and M. Yao, "Boundedness and convergence of online gadient method with penalty for feedforward neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 20, no. 6, pp. 1050–1054, 2009.

[43] G. Leitmann, *The calculus of variations and optimal control: an introduction*, vol. 24, Springer Science & Business Media, 2013.