Neurocomputing 408 (2020) 42–50

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Brief papers

Convolutional neural networks with fractional order gradient method

Dian Sheng, Yiheng Wei, Yuquan Chen, Yong Wang*

Department of Automation, University of Science and Technology of China, Hefei 230026, China

ARTICLE INFO

Article history:
Received 17 April 2019
Revised 9 September 2019
Accepted 8 October 2019
Available online 17 October 2019

Communicated by Prof. Qiankun Song

Keywords:
Fractional order calculus
Gradient method
Neural networks
Backward propagation

ABSTRACT

This paper proposes a fractional order gradient method for the backward propagation of convolutional neural networks. To overcome the problem that fractional order gradient method cannot converge to real extreme point, a simplified fractional order gradient method is designed based on Caputo's definition. The parameters within layers are updated by the designed gradient method, but the propagations between layers still use integer order gradients, and thus the complicated derivatives of composite functions are avoided and the chain rule will be kept. By connecting every layers in series and adding loss functions, the proposed convolutional neural networks can be trained smoothly according to various tasks. Some practical experiments are carried out in order to demonstrate fast convergence, high accuracy and ability to escape local optimal point at last.

1. Introduction

Machine learning technology powers many aspects of modern society: from web searches to content filtering on social networks to recommendations on e-commerce websites, and it is increasingly presented in consumer products such as cameras and smart phones [1]. During the development of machine learning, a variety of artificial neural networks have been created and gradually playing a more and more important role. Although it is still far from perfection, artificial neural networks are proven to be excellent enough, especially for convolutional neural networks (CNN). Research on CNN can be traced back to 1979. Based on visual cortex, Fukushima designed the neural networks named 'neocognitron' which is regarded as the origin of CNN [2]. As a breakthrough has been made for the back propagation neural networks (BPNN) [3], the first CNN was invented by applying the backward propagation in training model [4]. After the emergence of two-dimensional CNN, scholars proposed a series of networks such as LeNet [5,6], AlexNet [7], VGG [8], GoogLeNet [9] and ResNet [10]. However, no matter how deep or large a neural network is, the key of algorithm is gradient method in backward propagation.

As fractional order calculus is successfully applied in LMS filtering [11–13], systems identification [14,15], control theories [16–19] and so on, there arises a new trend that introduces fractional order calculus into gradient method. Professor Pu is the first one who pays attention to fractional order gradient method. He adopts fractional order derivatives to replace the integer order derivatives in traditional gradient method directly [20]. According to the definitions of fractional order derivatives [21], not only current information but also historical one are taken into consideration, thus it is potential for fractional order gradient to escape local optimal point. Nevertheless, such gradient method cannot ensure the convergence to real extreme point, even if the objective function is a simple quadratic function. To remedy this congenital defect, Chen uses truncation and short memory principle to modify the fractional order gradient method [22,23] which turns out: it is convergent to real extreme point just as integer order method do, but with faster convergent speed than integer order method.

During the research of fractional order gradient method, some scholars have found its application to artificial neural networks at the same time. Considering that fractional order derivatives of composite functions are complicated, professor Wang only uses fractional order gradients for updating parameters so that the chain rule will be kept to calculate integer order gradients along backward propagation [24]. Similar method is followed but the different structure of networks is applied in [25]. Their experiments demonstrate that fractional order gradients improve the networks performance on accuracy, and the cost of time has changed little because of relatively simpler calculation by adopting Caputo's definition. However, their fractional order gradient method is based on the strict definition of fractional order derivatives, which leads to the same problem as [20].

Even if great efforts have been made to neural networks with fractional order gradient method, it is still a novel research and far away from perfection at present. There remain some aspects to be improved.

* Corresponding author.
E-mail address: yongwang@ustc.edu.cn (Y. Wang).

https://doi.org/10.1016/j.neucom.2019.10.017
0925-2312/

- The convergence to real extreme point is necessary for gradient method.
- The available range of fractional order can be extended to $0<\alpha<2$.
- Neural networks of more complicated structure are worth researching in depth.
- How to use the chain rule in fractional order neural networks is still a problem.
- Loss function may be chosen as not only quadratic function but cross-entropy function.

Therefore, this paper provides conventional CNN with a novel fractional order gradient method. To the best of our knowledge, no scholar has ever investigated the CNN by fractional order gradient method. The proposed method is creative for neural networks as well as gradient method. First, based on the Caputo's definition of fractional order derivatives, a fractional order gradient method is designed and proved to converge to real extreme point. Second, the gradients in backward propagation of neural networks are divided into two categories, namely the gradients transferred between layers and the gradients for updating parameters within layers. Third, the updating gradients are replaced by fractional order one, but transferring gradients are integer order so that the chain rule could be kept using. With connecting all layers end-to-end and adding loss functions, the CNN with fractional order gradient method is achieved. What's more, the proposed neural networks validate that fractional order gradients perform outstanding acts of fast convergence, good accuracy and ability to escape local optimal point.

The remainder of this article is organized as follows. Section 2 introduces a fractional order gradient method and provides some basic knowledge for subsequent use. Fractional order gradient method is recommended for the fully connected layers and convolution layers in Section 3, respectively. In Section 4, some experiments are provided to illustrate the validity of the proposed approach. Conclusions are given in Section 5.

2. Preliminaries

A general CNN is composed of convolution layers, pooling layers and fully connected layers. Fractional order gradient method is applicable for all layers except pooling layers, since there is no need of updating parameters in pooling layers. In the fully connected layers, each node is connected to all nodes of last layer, whereas the convolution layers use many convolution kernels to scan the input. In spite of different structures, their backward propagations are almost the same, which make the study on gradient method much easier. Before the introduction of fractional order gradients within backward propagations, some preliminary knowledge needs emphasizing here.

There are some widely accepted definitions of fractional order derivative $\mathscr{D}^{\alpha}$, such as Riemann–Liouville, Caputo and Grunwald–Letnikov, but the Caputo's one is chosen for subsequent use, since its derivative of constant equals zero. The Caputo's definition is

$$ {}_{t_0}\mathscr{D}_t^{\alpha} f(t)=\frac{1}{\Gamma(m-\alpha)}\int_{t_0}^{t}\frac{f^{(m)}(\tau)}{(t-\tau)^{\alpha-m+1}}\mathrm{d}\tau, \tag{1}$$

where $m-1<\alpha<m, m\in\mathbb{N}^{+}$, $\Gamma(\alpha)=\int_0^{\infty}x^{\alpha-1}\mathrm{e}^{-x}\mathrm{d}x$ is the Gamma function, t_0 is the initial value. Alternatively, (1) can be rewritten as the following form

$$ {}_{t_0}\mathscr{D}_t^{\alpha} f(t)=\sum_{i=m}^{\infty}\frac{f^{(i)}(t_0)}{\Gamma(i+1-\alpha)}(t-t_0)^{i-\alpha}. \tag{2}$$

Suppose $f(x)$ to be a smooth convex function with a unique extreme point x^*. It is well known that each iterative step of the conventional gradient method is formulated as

$$ x_{K+1}=x_K-\mu\ f^{(1)}(x_K), \tag{3}$$

where μ is the iterative step size or learning rate, K is iterative times. Similarly, the fractional order gradient method is written as

$$ x_{K+1}=x_K-\mu\ {}_{x_0}\mathscr{D}_{x_K}^{\alpha} f(x). \tag{4}$$

If fractional order derivatives are directly applied in (4), the above fractional order gradient method cannot converge to the real extreme point x^*, but to an extreme point under definition of fractional order derivatives, such extreme point is associated with initial value and order, generally not equal to x^* [20].

To guarantee the convergence to real extreme point, an alternative fractional order gradient method [22] is considered via following iterative step

$$ x_{K+1}=x_K-\mu\ {}_{x_{K-1}}\mathscr{D}_{x_K}^{\alpha} f(x), \tag{5}$$

with $0<\alpha<1$ and

$$ {}_{x_{K-1}}\mathscr{D}_{x_K}^{\alpha} f(x)=\sum_{i=0}^{\infty}\frac{f^{(i+1)}(x_{K-1})}{\Gamma(i+2-\alpha)}(x_K-x_{K-1})^{i+1-\alpha}. \tag{6}$$

When only the first item is reserved and its absolute value is introduced, the fractional order gradient method with $0<\alpha<2$ is simplified as

$$ x_{K+1}=x_K-\mu\ \frac{f^{(1)}(x_{K-1})}{\Gamma(2-\alpha)}|x_K-x_{K-1}|^{1-\alpha}. \tag{7}$$

Theorem 1. *If fractional order gradient method (7) is convergent, it will converge to the real extreme point x^*.*

Proof. It is a proof by contradiction. Assume that x_K converges to a different point $X\neq x^*$, namely $\lim_{K\to\infty}|x_K-X|=0$. Therefore, it can be concluded that for any sufficient small positive scalar ε, there exists a sufficient large number $N\in\mathbb{N}$ such that $|x_{K-1}-X|<\varepsilon<|x^*-X|$ for any $K-1>N$. Then $\delta=\inf_{K-1>N}|f^{(1)}(x_{K-1})|>0$ must hold.

According to (7), the following inequality is obtained

$$\begin{aligned}|x_{K+1}-x_K|&=\left|\mu\frac{f^{(1)}(x_{K-1})}{\Gamma(2-\alpha)}|x_K-x_{K-1}|^{1-\alpha}\right|\\&=\mu\frac{|f^{(1)}(x_{K-1})|}{\Gamma(2-\alpha)}|x_K-x_{K-1}|^{1-\alpha}\\&\geq d|x_K-x_{K-1}|^{1-\alpha},\end{aligned} \tag{8}$$

with $d=\mu\frac{\delta}{\Gamma(2-\alpha)}$.

Considering that one can always find a ε such that $2\varepsilon<d^{\frac{1}{\alpha}}$, then the following inequality will hold

$$ |x_K-x_{K-1}|\leq|x_K-X|+|x_{K-1}-X|<2\varepsilon<d^{\frac{1}{\alpha}}. \tag{9}$$

The above inequality could be rewritten as $d>|x_K-x_{K-1}|^{\alpha}$. When this inequality is introduced into (8), the result is

$$ |x_{K+1}-x_K|>|x_K-x_{K-1}|, \tag{10}$$

which implies that x_K is not convergent. It contradicts to the assumption that x_K is convergent to X, thus the proof is completed. □

Remark 1. When a small positive value $\delta>0$ is introduced, the following fractional order gradient method will avoid singularity caused by $x_K=x_{K-1}$.

$$ x_{K+1}=x_K-\mu\ \frac{f^{(1)}(x_{K-1})}{\Gamma(2-\alpha)}(|x_K-x_{K-1}|+\delta)^{1-\alpha}. \tag{11}$$

Compared with gradient method based on strict definition of fractional order derivatives [20], the modified fractional order gradient methods (7) and other similar methods [22] are proven to be

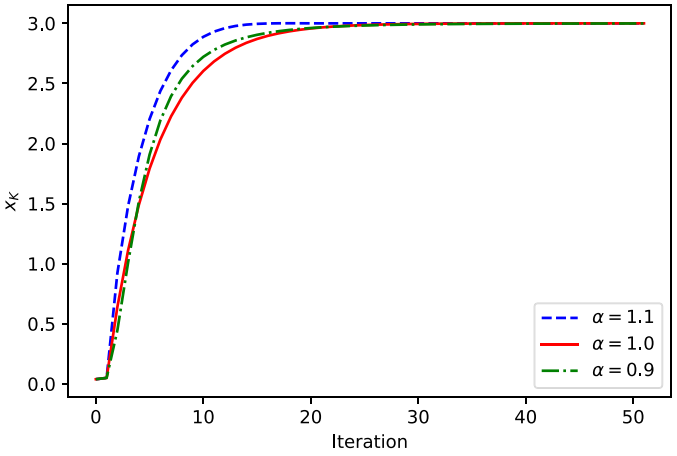


Fig. 1. The performance of different gradient methods.

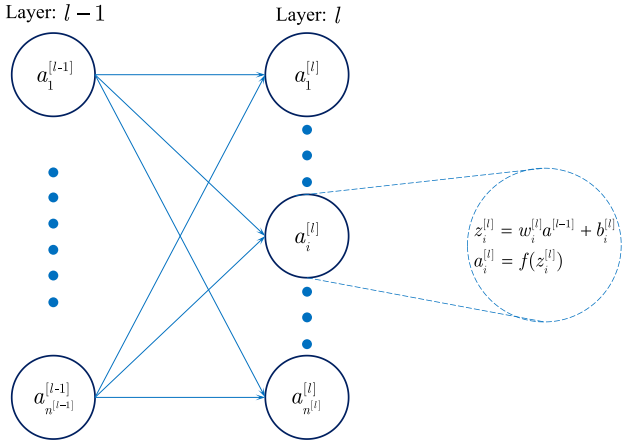


Fig. 2. Forward propagation of fully connected layers.

convergent to the real extreme point. To demonstrate faster convergence of proposed gradient method, a common type of objective function, namely quadratic function, is selected to show optimizing process.

The objective function is $f(x) = (x-3)^2$ where the step size is set to 0.1 and two initial points are randomly chosen in [0, 0.1]. Other initializations are completely available only if the distance between two points is restricted to a small range. As is shown by Fig. 1, the optimizing process is obviously promoted with the application of fractional order gradient method.

In view of theory and practice above, compared with existing integer order or fractional order gradient method, the proposed fractional order gradient method not only is convergent to real extreme point but also converges with faster speed.

3. CNN with fractional order gradients

Although the key procedure of mathematical calculation is quite similar in convolution layers and fully connected layers, the different structures lead to different ways to research. First of all, fully connected layers with fractional order gradient are introduced.

3.1. Fully connected layers

The training procedure of neural networks contains two steps, one of them is forward propagation. Such propagation between two layers is illustrated as Fig. 2, where superscript $[l]$ is the number of layer, subscript i is the number of node in certain layer, $a_i^{[l]} \in \mathbb{R}$ is the output of ith node in lth layer.

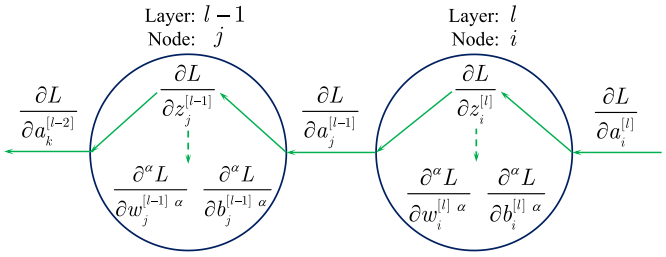


Fig. 3. Backward propagation of fully connected layers.

The output $a_i^{[l]}$ is from

$$\begin{cases} z_i^{[l]} = w_i^{[l]} a^{[l-1]} + b_i^{[l]}, \\ a_i^{[l]} = f(z_i^{[l]}), \end{cases} \tag{12}$$

where $w_i^{[l]} = [w_{i1}^{[l]}, w_{i2}^{[l]}, \ldots, w_{in^{[l-1]}}^{[l]}] \in \mathbb{R}^{n^{[l-1]}}$ is weight, $b_i^{[l]} \in \mathbb{R}$ is bias, $a^{[l-1]} = [a_1^{[l-1]}, a_2^{[l-1]}, \ldots, a_{n^{[l-1]}}^{[l-1]}]^{\mathrm{T}} \in \mathbb{R}^{n^{[l-1]}}$ is the output of last layer and function $f(\cdot)$ is activation function.

Another step of training procedure is backward propagation, in which fractional order gradient method takes the place of traditional method. Due to imperfect use of chain rule in fractional order derivatives, the gradients of backward propagation are a blend of fractional order and integer order. As is shown in Fig. 3, there are two types of gradients that pass through layers. One is the transferring gradient (solid line) which links nodes between two layers, the other is updating gradient (dotted line) which is used for parameters within layers. L is the loss function, α is the fractional order, $\frac{\partial^\alpha L}{\partial w_i^{[l]\alpha}}$ and $\frac{\partial^\alpha L}{\partial b_i^{[l]\alpha}}$ are defined as fractional order gradients of $w_i^{[l]}$ and $b_i^{[l]}$, respectively.

In order to use the chain rule continuously, the transferring gradient is provided with integer order

$$\begin{cases} \dfrac{\partial L}{\partial z_i^{[l]}} = \dfrac{\partial L}{\partial a_i^{[l]}} \dfrac{\partial a_i^{[l]}}{\partial z_i^{[l]}} = \dfrac{\partial L}{\partial a_i^{[l]}} f^{(1)}(z_i^{[l]}), \\ \dfrac{\partial L}{\partial a_j^{[l-1]}} = \displaystyle\sum_{i=1}^{n^{[l]}} \dfrac{\partial L}{\partial a_i^{[l]}} \dfrac{\partial a_i^{[l]}}{\partial z_i^{[l]}} \dfrac{\partial z_i^{[l]}}{\partial a_j^{[l-1]}} = \sum_{i=1}^{n^{[l]}} \dfrac{\partial L}{\partial z_i^{[l]}} w_{ij}^{[l]}, \end{cases} \tag{13}$$

but the updating gradient is replaced by fractional order

$$\begin{cases} \dfrac{\partial^\alpha L}{\partial w_{ij}^{[l]\alpha}} = \dfrac{\partial L}{\partial z_i^{[l]}} \dfrac{\partial^\alpha z_i^{[l]}}{\partial w_{ij}^{[l]\alpha}} = \dfrac{\partial L}{\partial a_i^{[l]}} f^{(1)}(z_i^{[l]}) \dfrac{\partial^\alpha z_i^{[l]}}{\partial w_{ij}^{[l]\alpha}}, \\ \dfrac{\partial^\alpha L}{\partial b_i^{[l]\alpha}} = \dfrac{\partial L}{\partial z_i^{[l]}} \dfrac{\partial^\alpha z_i^{[l]}}{\partial b_i^{[l]\alpha}} = \dfrac{\partial L}{\partial a_i^{[l]}} f^{(1)}(z_i^{[l]}) \dfrac{\partial^\alpha z_i^{[l]}}{\partial b_i^{[l]\alpha}}, \end{cases} \tag{14}$$

with $i = 1, 2, \ldots, n^{[l]}$ and $j = 1, 2, \ldots, n^{[l-1]}$. When the fractional order gradient (7) is adopted, the gradient of the Kth iteration becomes

$$\begin{cases} \dfrac{\partial^\alpha z_i^{[l]}}{\partial w_{ij}^{[l]\alpha}} = \dfrac{a_{j\ (K-1)}^{[l-1]}}{\Gamma(2-\alpha)} |w_{ij\ (K)}^{[l]} - w_{ij\ (K-1)}^{[l]}|^{1-\alpha}, \\ \dfrac{\partial^\alpha z_i^{[l]}}{\partial b_i^{[l]\alpha}} = \dfrac{1}{\Gamma(2-\alpha)} |b_{i\ (K)}^{[l]} - b_{i\ (K-1)}^{[l]}|^{1-\alpha}, \end{cases} \tag{15}$$

where $w_{ij\ (K)}^{[l]}$ and $b_{i\ (K)}^{[l]}$ are parameters $w_{ij}^{[l]}$ and $b_i^{[l]}$ at the Kth iteration, $a_{j\ (K-1)}^{[l-1]}$ is output $a_j^{[l-1]}$ at the $(K-1)$th iteration. Consequently, the fractional order updating gradient is achieved by

introducing (15)–(14)

$$\begin{cases} \dfrac{\partial^\alpha L}{\partial w_{ij}^{[l]\alpha}} = \dfrac{\partial L}{\partial a_{i\ (K-1)}^{[l]}} f^{(1)}(z_{i\ (K-1)}^{[l]}) \dfrac{a_{j\ (K-1)}^{[l-1]}}{\Gamma(2-\alpha)} |w_{ij\ (K)}^{[l]} - w_{ij\ (K-1)}^{[l]}|^{1-\alpha}, \\ \dfrac{\partial^\alpha L}{\partial b_i^{[l]\alpha}} = \dfrac{\partial L}{\partial a_{i\ (K-1)}^{[l]}} f^{(1)}(z_{i\ (K-1)}^{[l]}) \dfrac{1}{\Gamma(2-\alpha)} |b_{i\ (K)}^{[l]} - b_{i\ (K-1)}^{[l]}|^{1-\alpha}, \end{cases} \tag{16}$$

where $z_{i\ (K-1)}^{[l]}$ and $\frac{\partial L}{\partial a_{i\ (K-1)}^{[l]}}$ are $z_i^{[l]}$ and $\frac{\partial L}{\partial a_i^{[l]}}$ at the $(K-1)$th iteration, respectively.

Actually, samples are not input one by one in most case. When a batch of samples are input each time, (16) turns into

$$\begin{cases} \dfrac{\partial^\alpha L}{\partial w_{ij}^{[l]\alpha}} = \displaystyle\sum_{s=1}^{m} \frac{\partial L}{\partial z_{is}^{[l]}} \frac{\partial^\alpha z_{is}^{[l]}}{\partial w_{ij}^{[l]\alpha}} \\ \qquad = \displaystyle\sum_{s=1}^{m} \frac{\partial L}{\partial a_{is\ (K-1)}^{[l]}} f^{(1)}(z_{is\ (K-1)}^{[l]}) \frac{a_{js\ (K-1)}^{[l-1]}}{\Gamma(2-\alpha)} |w_{ij\ (K)}^{[l]} \\ \qquad -w_{ij\ (K-1)}^{[l]}|^{1-\alpha}, \\ \dfrac{\partial^\alpha L}{\partial b_i^{[l]\alpha}} = \displaystyle\sum_{s=1}^{m} \frac{\partial L}{\partial z_{is}^{[l]}} \frac{\partial^\alpha z_{is}^{[l]}}{\partial b_i^{[l]\alpha}} \\ \qquad = \displaystyle\sum_{s=1}^{m} \frac{\partial L}{\partial a_{is\ (K-1)}^{[l]}} f^{(1)}(z_{is\ (K-1)}^{[l]}) \frac{1}{\Gamma(2-\alpha)} |b_{i\ (K)}^{[l]} \\ \qquad -b_{i\ (K-1)}^{[l]}|^{1-\alpha}, \end{cases} \tag{17}$$

where m is batch size, the subscript s means sth sample of a batch. After vectorization, above equations are simplified as

$$\begin{cases} \dfrac{\partial^\alpha L}{\partial W^{[l]\alpha}} = \dfrac{1}{\Gamma(2-\alpha)} \left[\dfrac{\partial L}{\partial A_{(K-1)}^{[l]}} \circ f^{(1)}(Z_{(K-1)}^{[l]}) \right] A_{(K-1)}^{[l-1]\ \mathrm{T}} \\ \qquad \circ |W_{(K)}^{[l]} - W_{(K-1)}^{[l]}|^{1-\alpha}, \\ \dfrac{\partial^\alpha L}{\partial b^{[l]\alpha}} = \dfrac{1}{\Gamma(2-\alpha)} \mathrm{sum}\left(\dfrac{\partial L}{\partial A_{(K-1)}^{[l]}} \circ f^{(1)}(Z_{(K-1)}^{[l]}) \right) \\ \qquad \circ |b_{(K)}^{[l]} - b_{(K-1)}^{[l]}|^{1-\alpha}, \\ \dfrac{\partial L}{\partial A_{(K-1)}^{[l-1]}} = W_{(K-1)}^{[l]\ \mathrm{T}} \left[\dfrac{\partial L}{\partial A_{(K-1)}^{[l]}} \circ f^{(1)}(Z_{(K-1)}^{[l]}) \right] \end{cases} \tag{18}$$

where

$$\frac{\partial^\alpha L}{\partial W^{[l]\alpha}} = \begin{bmatrix} \frac{\partial^\alpha L}{\partial w_{11}^{[l]\alpha}} & \cdots & \frac{\partial^\alpha L}{\partial w_{1n^{[l-1]}}^{[l]\alpha}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^\alpha L}{\partial w_{n^{[l]}1}^{[l]\alpha}} & \cdots & \frac{\partial^\alpha L}{\partial w_{n^{[l]}n^{[l-1]}}^{[l]\alpha}} \end{bmatrix}, \quad \frac{\partial^\alpha L}{\partial b^{[l]\alpha}} = \begin{bmatrix} \frac{\partial^\alpha L}{\partial b_1^{[l]\alpha}} \\ \vdots \\ \frac{\partial^\alpha L}{\partial b_{n^{[l]}}^{[l]\alpha}} \end{bmatrix},$$

$$W_{(K)}^{[l]} = \begin{bmatrix} w_{11\ (K)}^{[l]} & \cdots & w_{1n^{[l-1]}\ (K)}^{[l]} \\ \vdots & \ddots & \vdots \\ w_{n^{[l]}1\ (K)}^{[l]} & \cdots & w_{n^{[l]}n^{[l-1]}\ (K)}^{[l]} \end{bmatrix},$$

$$b_{(K)}^{[l]} = \begin{bmatrix} b_{1\ (K)}^{[l]} \\ \vdots \\ b_{n^{[l]}\ (K)}^{[l]} \end{bmatrix},$$

$$A_{(K-1)}^{[l-1]\mathrm{T}} = \begin{bmatrix} a_{11\ (K-1)}^{[l-1]} & \cdots & a_{1m\ (K-1)}^{[l-1]} \\ \vdots & \ddots & \vdots \\ a_{n^{[l-1]}1\ (K-1)}^{[l-1]} & \cdots & a_{n^{[l-1]}m\ (K-1)}^{[l-1]} \end{bmatrix}^{\mathrm{T}},$$

$$\frac{\partial L}{\partial A_{(K-1)}^{[l]}} = \begin{bmatrix} \frac{\partial L}{\partial a_{11\ (K-1)}^{[l]}} & \cdots & \frac{\partial L}{\partial a_{1m\ (K-1)}^{[l]}} \\ \vdots & \ddots & \vdots \\ \frac{\partial L}{\partial a_{n^{[l]}1\ (K-1)}^{[l]}} & \cdots & \frac{\partial L}{\partial a_{n^{[l]}m\ (K-1)}^{[l]}} \end{bmatrix},$$

$$f^{(1)}(Z_{(K-1)}^{[l]}) = \begin{bmatrix} f^{(1)}(z_{11\ (K-1)}^{[l]}) & \cdots & f^{(1)}(z_{1m\ (K-1)}^{[l]}) \\ \vdots & \ddots & \vdots \\ f^{(1)}(z_{n^{[l]}1\ (K-1)}^{[l]}) & \cdots & f^{(1)}(z_{n^{[l]}m\ (K-1)}^{[l]}) \end{bmatrix},$$

signs like $-, |\cdot|$ and $(\cdot)^{1-\alpha}$ are the element-wise calculation, $\circ$ is the Hadamard product, $\mathrm{sum}(\cdot)$ is the sum of a matrix along horizontal axis. Then the updating parameters of fully connected layers can be summarized as

$$\begin{cases} W_{(K+1)}^{[l]} = W_{(K)}^{[l]} - \mu \ \frac{\partial^\alpha L}{\partial W^{[l]\alpha}}, \\ b_{(K+1)}^{[l]} = b_{(K)}^{[l]} - \mu \ \dfrac{\partial^\alpha L}{\partial b^{[l]\alpha}}. \end{cases} \tag{19}$$

Theorem 2. *The fully connected layers updated by fractional order gradient method (18, 19) are convergent to real extreme point.*

Proof. When integer order gradients are adopted in backward propagation, the gradients of $W_{(K-1)}^{[l]}$ and $b_{(K-1)}^{[l]}$ can be written as

$$\begin{cases} \dfrac{\partial L}{\partial W_{(K-1)}^{[l]}} = \left[\dfrac{\partial L}{\partial A_{(K-1)}^{[l]}} \circ f^{(1)}(Z_{(K-1)}^{[l]}) \right] A_{(K-1)}^{[l-1]\ \mathrm{T}}, \\ \dfrac{\partial L}{\partial b_{(K-1)}^{[l]}} = \mathrm{sum}\left(\dfrac{\partial L}{\partial A_{(K-1)}^{[l]}} \circ f^{(1)}(Z_{(K-1)}^{[l]}) \right). \end{cases} \tag{20}$$

Thus fractional order gradients (18) turn into

$$\begin{cases} \dfrac{\partial^\alpha L}{\partial W^{[l]\alpha}} = \dfrac{1}{\Gamma(2-\alpha)} \dfrac{\partial L}{\partial W_{(K-1)}^{[l]}} \circ |W_{(K)}^{[l]} - W_{(K-1)}^{[l]}|^{1-\alpha}, \\ \dfrac{\partial^\alpha L}{\partial b^{[l]\alpha}} = \dfrac{1}{\Gamma(2-\alpha)} \dfrac{\partial L}{\partial b_{(K-1)}^{[l]}} \circ |b_{(K)}^{[l]} - b_{(K-1)}^{[l]}|^{1-\alpha}. \end{cases} \tag{21}$$

Then the updating parameters (19) becomes

$$\begin{cases} W_{(K+1)}^{[l]} = W_{(K)}^{[l]} - \mu \ \dfrac{1}{\Gamma(2-\alpha)} \dfrac{\partial L}{\partial W_{(K-1)}^{[l]}} \\ \qquad \circ |W_{(K)}^{[l]} - W_{(K-1)}^{[l]}|^{1-\alpha}, \\ b_{(K+1)}^{[l]} = b_{(K)}^{[l]} - \mu \ \frac{1}{\Gamma(2-\alpha)} \frac{\partial L}{\partial b_{(K-1)}^{[l]}} \\ \qquad \circ |b_{(K)}^{[l]} - b_{(K-1)}^{[l]}|^{1-\alpha}. \end{cases} \tag{22}$$

For a certain element in $W^{[l]}$ or $b^{[l]}$, the $w_{ij}^{[l]}$ or $b_i^{[l]}$ can be updated by

$$\begin{cases} w_{ij\ (K+1)}^{[l]} = w_{ij\ (K)}^{[l]} - \mu \ \dfrac{1}{\Gamma(2-\alpha)} \dfrac{\partial L}{\partial w_{ij\ (K-1)}^{[l]}} \\ \qquad |w_{ij\ (K)}^{[l]} - w_{ij\ (K-1)}^{[l]}|^{1-\alpha}, \\ b_{i\ (K+1)}^{[l]} = b_{i\ (K)}^{[l]} - \mu \ \dfrac{1}{\Gamma(2-\alpha)} \dfrac{\partial L}{\partial b_{i\ (K-1)}^{[l]}} \\ \qquad |b_{i\ (K)}^{[l]} - b_{i\ (K-1)}^{[l]}|^{1-\alpha}. \end{cases} \tag{23}$$

It is similar to the proof of Theorem 1. Assume that $w_{ij\ (K)}^{[l]}$ converges to a point $w_{ij}^{[l]\prime}$ that is different from real extreme point $w_{ij}^{[l]*}$, namely $\lim_{K\to\infty}|w_{ij\ (K)}^{[l]}-w_{ij}^{[l]\prime}|=0$. Therefore, it can be concluded that for any sufficient small positive scalar ε, there exists a sufficient large number $N\in\mathbb{N}$ such that $|w_{ij\ (K)}^{[l]}-w_{ij}^{[l]\prime}|<\varepsilon<|w_{ij}^{[l]*}-w_{ij}^{[l]\prime}|$ for any $K-1>N$. Then $\delta=\inf_{K-1>N}|\frac{\partial L}{\partial w_{ij\ (K-1)}^{[l]}}|>0$ must hold.

According to (23), the following inequality is obtained

$$\begin{aligned}
&|w_{ij\ (K+1)}^{[l]}-w_{ij\ (K)}^{[l]}|\\
&=\left|\mu\ \frac{1}{\Gamma(2-\alpha)}\frac{\partial L}{\partial w_{ij\ (K-1)}^{[l]}}|w_{ij\ (K)}^{[l]}-w_{ij\ (K-1)}^{[l]}|^{1-\alpha}\right|\\
&=\frac{\mu}{\Gamma(2-\alpha)}\left|\frac{\partial L}{\partial w_{ij\ (K-1)}^{[l]}}\right||w_{ij\ (K)}^{[l]}-w_{ij\ (K-1)}^{[l]}|^{1-\alpha}\\
&\geq d|w_{ij\ (K)}^{[l]}-w_{ij\ (K-1)}^{[l]}|^{1-\alpha},
\end{aligned}\tag{24}$$

with $d=\frac{\mu\delta}{\Gamma(2-\alpha)}$.

Considering that one can always find a ε such that $2\varepsilon<d^{\frac{1}{\alpha}}$, then the following inequality will hold

$$\begin{aligned}
|w_{ij\ (K)}^{[l]}-w_{ij\ (K-1)}^{[l]}|&\leq|w_{ij\ (K)}^{[l]}-w_{ij}^{[l]\prime}|+|w_{ij\ (K-1)}^{[l]}-w_{ij}^{[l]\prime}|\\
&<2\varepsilon<d^{\frac{1}{\alpha}}.
\end{aligned}\tag{25}$$

The above inequality could be rewritten as $d>|w_{ij\ (K)}^{[l]}-w_{ij\ (K-1)}^{[l]}|^{\alpha}$. When this inequality is introduced into (24), the result is

$$|w_{ij\ (K+1)}^{[l]}-w_{ij\ (K)}^{[l]}|>|w_{ij\ (K)}^{[l]}-w_{ij\ (K-1)}^{[l]}|,\tag{26}$$

which implies that $w_{ij\ (K)}^{[l]}$ is not convergent. Similarly, the same result will be easily obtained for $b_{i\ (K)}^{[l]}$. It contradicts to the assumption mentioned before, thus the proof is completed □

Compared with integer order backward propagation [26], the same transferring gradient $\frac{\partial L}{\partial A^{[l-1]}_{(K-1)}}$ is kept, but the difference exists in updating gradient where the order is changed as $\frac{\partial^{\alpha}L}{\partial W^{[l]\alpha}}$ and $\frac{\partial^{\alpha}L}{\partial b^{[l]\alpha}}$. Even so, based on the Theorem 2, $W^{[l]}$ and $b^{[l]}$ updated by fractional order gradients will converge to the same real extreme points as integer order gradients do.

Remark 2. Because of integer order transferring gradient, the chain rule is still available for the proposed gradient method (18,19), which avoids complicated calculation caused by fractional order derivatives, especially derivatives of activation function. As modified fractional order gradient (7) is applied smoothly, the speed of convergence is improved and real extreme point can be reached now.

3.2. Convolution layers

Although the key calculation of convolution layers is similar to fully connected layers, the complicated structure makes its iterative algorithm different. It is hard to understand the algorithm without help of figures or auxiliary descriptions.

For subsequent research, the forward propagation of convolution layers is drawn briefly in Fig. 4 where $a^{[l]}\in\mathbb{R}^{n_H^{[l]}\times n_W^{[l]}\times n_C^{[l]}}$ is the

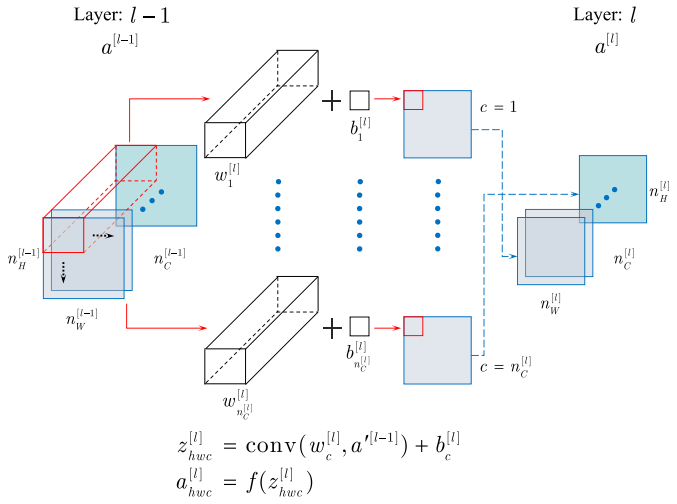


Fig. 4. Forward propagation of convolution layers.

output of the lth layer, $w_c^{[l]}\in\mathbb{R}^{F^{[l]}\times F^{[l]}\times n_C^{[l-1]}}$ and $b_c^{[l]}\in\mathbb{R}$ are weight and bias for channel c, $F^{[l]}$ is the size of convolution kernel, $a'^{[l-1]}$ is a slice of $a^{[l-1]}$ by selecting $F^{[l]}$ rows and $F^{[l]}$ columns over all channels (red cube), $n_H^{[l]}$, $n_W^{[l]}$ and $n_C^{[l]}$ are height, width and channels of output $a^{[l]}$, respectively.

Similarly, the gradients of backward propagation are divided into two types. The transferring gradient of convolution layers is also kept the same as integer order gradient. Considering that the input is a batch of samples, the updating gradient is

$$\begin{cases}
\dfrac{\partial^{\alpha}L}{\partial w_{ijkc}^{[l]\alpha}}=\displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\frac{\partial L}{\partial A_{shwc}^{[l]}}\frac{\partial A_{shwc}^{[l]}}{\partial Z_{shwc}^{[l]}}\frac{\partial^{\alpha}Z_{shwc}^{[l]}}{\partial w_{ijkc}^{[l]\alpha}}\\
\qquad\qquad=\displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\frac{\partial L}{\partial A_{shwc}^{[l]}}f^{(1)}(Z_{shwc}^{[l]})\frac{\partial^{\alpha}Z_{shwc}^{[l]}}{\partial w_{ijkc}^{[l]\alpha}},\\
\dfrac{\partial^{\alpha}L}{\partial b_{c}^{[l]\alpha}}=\displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\frac{\partial L}{\partial A_{shwc}^{[l]}}\frac{\partial A_{shwc}^{[l]}}{\partial Z_{shwc}^{[l]}}\frac{\partial^{\alpha}Z_{shwc}^{[l]}}{\partial b_{c}^{[l]\alpha}}\\
\qquad\qquad=\displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\frac{\partial L}{\partial A_{shwc}^{[l]}}f^{(1)}(Z_{shwc}^{[l]})\frac{\partial^{\alpha}Z_{shwc}^{[l]}}{\partial b_{c}^{[l]\alpha}},
\end{cases}\tag{27}$$

where $W^{[l]}=[w_{ijkc}^{[l]}]\in\mathbb{R}^{F^{[l]}\times F^{[l]}\times n_C^{[l-1]}\times n_C^{[l]}}$ is the weight that contains all $w_c^{[l]}$, $A^{[l]}\in\mathbb{R}^{m\times n_H^{[l]}\times n_W^{[l]}\times n_C^{[l]}}$ is the $a^{[l]}$ over all m samples.

When the fractional order gradient method (7) is introduced, the updating gradient at the Kth iteration is changed to

$$\begin{cases}
\dfrac{\partial^{\alpha}L}{\partial w_{ijkc}^{[l]\alpha}}=\displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\left[\frac{\partial L}{\partial A_{shwc\ (K-1)}^{[l]}}f^{(1)}(Z_{shwc\ (K-1)}^{[l]})\right.\\
\qquad\left.\dfrac{A'^{[l-1]}_{ijk\ (K-1)}}{\Gamma(2-\alpha)}|w_{ijkc\ (K)}^{[l]}-w_{ijkc\ (K-1)}^{[l]}|^{1-\alpha}\right],\\
\dfrac{\partial^{\alpha}L}{\partial b_{c}^{[l]\alpha}}=\displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\left[\frac{\partial L}{\partial A_{shwc\ (K-1)}^{[l]}}f^{(1)}(Z_{shwc\ (K-1)}^{[l]})\right.\\
\qquad\left.\dfrac{1}{\Gamma(2-\alpha)}|b_{c\ (K)}^{[l]}-b_{c\ (K-1)}^{[l]}|^{1-\alpha}\right],
\end{cases}\tag{28}$$

where $A'^{[l-1]}$ is $a'^{[l-1]}$ of the sth sample. It could be simply regarded as $A'^{[l-1]}=A^{[l-1]}[s,V_{start}:V_{end},H_{start}:H_{end}]\in\mathbb{R}^{F^{[l]}\times F^{[l]}\times n_C^{[l-1]}}$

with

$$V_{start} = (h-1) \times stride + 1,\ V_{end} = V_{start} + F^{[l]},$$

$$H_{start} = (w-1) \times stride + 1,\ H_{end} = H_{start} + F^{[l]},$$

and *stride* is moving length of convolution kernel each time. After vectorization, (28) is further simplified as

$$\begin{cases} \dfrac{\partial^\alpha L}{\partial w_c^{[l]\alpha}} = \displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\left[\frac{1}{\Gamma(2-\alpha)}\frac{\partial L}{\partial A^{[l]}_{shwc\ (K-1)}} f^{(1)}\left(Z^{[l]}_{shwc\ (K-1)}\right)\right. \\ \qquad \left. A'^{\,[l-1]}_{(K-1)} \circ |w^{[l]}_{c\ (K)} - w^{[l]}_{c\ (K-1)}|^{1-\alpha}\right], \\ \dfrac{\partial^\alpha L}{\partial b_c^{[l]\alpha}} = \displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\left[\frac{1}{\Gamma(2-\alpha)}\frac{\partial L}{\partial A^{[l]}_{shwc\ (K-1)}} f^{(1)}\left(Z^{[l]}_{shwc\ (K-1)}\right)\right. \\ \qquad \left. |b^{[l]}_{c\ (K)} - b^{[l]}_{c\ (K-1)}|^{1-\alpha}\right], \end{cases} \tag{29}$$

In order to show the algorithm clearly, the calculation of (29) is transformed to following process.

Then the updating parameters of convolution layers is as follows

$$\begin{cases} W^{[l]}_{(K+1)} = W^{[l]}_{(K)} - \mu\ \dfrac{\partial^\alpha L}{\partial W^{[l]\alpha}}, \\ b^{[l]}_{(K+1)} = b^{[l]}_{(K)} - \mu\ \dfrac{\partial^\alpha L}{\partial b^{[l]\alpha}}. \end{cases} \tag{30}$$

Theorem 3. *The convolution layers updated by fractional order gradient method (29, 30) are convergent to real extreme point.*

The proof resembles Theorem 2. By introducing integer order gradients, the gradients in (29) are changed into the form like (21). Then it is a proof by contradiction that could be done for each element of $W^{[l]}$ and $b^{[l]}$ in (30).

Remark 3. Based on backward propagation of convolution layers, when padding is introduced into the lth layers, the transferring gradient will be influenced. The gradient $\frac{\partial L}{\partial A^{[l-1]}}$ calculated by Algorithm 1 is the gradient of padded output. The padded part of $\frac{\partial L}{\partial A^{[l-1]}}$ needs deleting. However, there is no change happened for the updating gradient of fractional order.

Remark 4. During the training procedure, a tiny value could be added to (18,29) so that the singularity caused by $W^{[l]}_{(K)} = W^{[l]}_{(K-1)}$ or $b^{[l]}_{(K)} = b^{[l]}_{(K-1)}$ is avoided easily. Hence the gradients modified by (11) are listed below

$$\begin{cases} \dfrac{\partial^\alpha L}{\partial W^{[l]\alpha}} = \dfrac{1}{\Gamma(2-\alpha)}\left[\dfrac{\partial L}{\partial A^{[l]}_{(K-1)}} \circ f^{(1)}(Z^{[l]}_{(K-1)})\right] A^{[l-1]\ \mathrm{T}}_{(K-1)} \\ \qquad \circ |W^{[l]}_{(K)} - W^{[l]}_{(K-1)} + \delta|^{1-\alpha}, \\ \dfrac{\partial^\alpha L}{\partial b^{[l]\alpha}} = \dfrac{1}{\Gamma(2-\alpha)}\mathrm{sum}\left(\dfrac{\partial L}{\partial A^{[l]}_{(K-1)}} \circ f^{(1)}(Z^{[l]}_{(K-1)})\right) \\ \qquad \circ |b^{[l]}_{(K)} - b^{[l]}_{(K-1)} + \delta|^{1-\alpha}, \end{cases} \tag{31}$$

Algorithm 1 Backward propagation of convolution layers by fractional order gradient method.

$\frac{\partial L}{\partial Z^{[l]}_{(K-1)}} = \frac{\partial L}{\partial A^{[l]}_{(K-1)}} \circ f^{(1)}(Z^{[l]}_{(K-1)})$
for $s = 1, 2, \ldots, m$ **do**
  for $h = 1, 2, \ldots, n_H^{[l]}$ **do**
    for $w = 1, 2, \ldots, n_W^{[l]}$ **do**
      $V_{start} = (h-1) \times stride + 1,\ V_{end} = V_{start} + F^{[l]}$
      $H_{start} = (w-1) \times stride + 1,\ H_{end} = H_{start} + F^{[l]}$
      for $c = 1, 2, \ldots, n_C^{[l]}$ **do**
        $A'^{\,[l-1]}_{(K-1)} = A^{[l-1]}_{(K-1)}[s, V_{start} : V_{end}, H_{start} : H_{end}]$
        $\frac{\partial^\alpha L}{\partial w_c^{[l]\alpha}} += \frac{\partial L}{\partial Z^{[l]}_{shwc\,(K-1)}} \frac{A'^{\,[l-1]}_{(K-1)}}{\Gamma(2-\alpha)} \circ |w^{[l]}_{c\,(K)} - w^{[l]}_{c\,(K-1)}|^{1-\alpha}$
        $\frac{\partial^\alpha L}{\partial b_c^{[l]\alpha}} += \frac{1}{\Gamma(2-\alpha)} \frac{\partial L}{\partial Z^{[l]}_{shwc\,(K-1)}} |b^{[l]}_{c\,(K)} - b^{[l]}_{c\,(K-1)}|^{1-\alpha}$
        $\frac{\partial L}{\partial A'^{[l-1]}_{(K)}} = \frac{\partial L}{\partial Z^{[l]}_{shwc\,(K)}} w^{[l]}_{c\,(K)}$
        $\frac{\partial L}{\partial A^{[l-1]}_{(K)}[s, V_{start}:V_{end}, H_{start}:H_{end}]} += \frac{\partial L}{\partial A'^{[l-1]}_{(K)}}$
      end for
    end for
  end for
end for
return $\frac{\partial^\alpha L}{\partial W^{[l]\alpha}}, \frac{\partial^\alpha L}{\partial b^{[l]\alpha}}, \frac{\partial L}{\partial A^{[l-1]}_{(K)}}$

$$\begin{cases} \dfrac{\partial^\alpha L}{\partial w_c^{[l]\alpha}} = \displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\left[\frac{1}{\Gamma(2-\alpha)}\frac{\partial L}{\partial A^{[l]}_{shwc\ (K-1)}} f^{(1)}(Z^{[l]}_{shwc\ (K-1)})\right. \\ \qquad \left. A'^{\,[l-1]}_{(K-1)} \circ |w^{[l]}_{c\ (K)} - w^{[l]}_{c\ (K-1)} + \delta|^{1-\alpha}\right], \\ \dfrac{\partial^\alpha L}{\partial b_c^{[l]\alpha}} = \displaystyle\sum_{s=1}^{m}\sum_{h=1}^{n_H^{[l]}}\sum_{w=1}^{n_W^{[l]}}\left[\frac{1}{\Gamma(2-\alpha)}\frac{\partial L}{\partial A^{[l]}_{shwc\ (K-1)}} f^{(1)}(Z^{[l]}_{shwc\ (K-1)})\right. \\ \qquad \left. |b^{[l]}_{c\ (K)} - b^{[l]}_{c\ (K-1)} + \delta|^{1-\alpha}\right]. \end{cases} \tag{32}$$

When convolution layers, pooling layers and fully connected layers are connected end-to-end, it will fulfill some tasks like classification. Many types of functions, such as quadratic function $L = (y-\hat{y})^2$ and cross-entropy function $L = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$, are available to loss function of proposed method. And the gradient on output of the last layer $\frac{\partial L}{\partial \hat{y}}$ is still integer order. The backward propagation between layers of different types is simple. Reshaping is needed only when the gradients correspond to different shapes. Finally, a type of complete convolutional neural networks will combine with fractional order gradient to fulfill a task and demonstrate the effectiveness of proposed method. For example, the LeNet [6] is a simple CNN which will be adopted for subsequent experiment.

4. Experiments

The task of experiments is to identify handwriting number by fractional order convolutional neural networks. The experiments are carried out by the MNIST dataset which consists of 60,000 handwritten digit images for the training and another 10,000 samples for testing. Consequently, the whole structure of LeNet is presented in Fig. 5, and some samples can be found in Fig. 6.

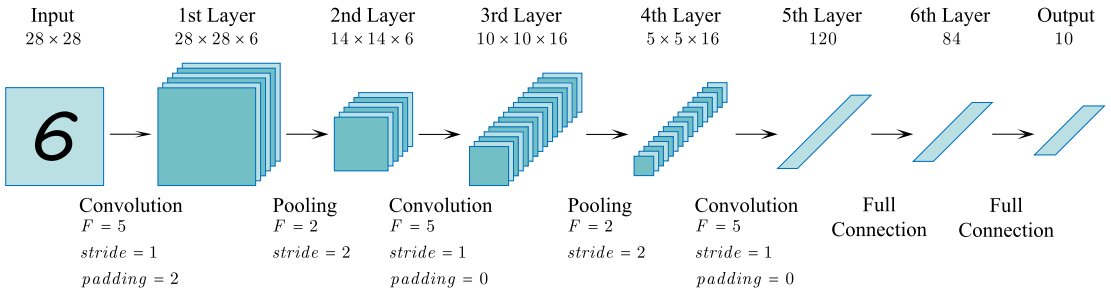


Fig. 5. Architecture of LeNet-5 with input of MNIST dataset.

Fig. 6. Some samples of MNIST dataset.

Table 1
The average training and testing accuracy.

Order(α)	Training accuracy	Testing accuracy
1.9	0.0980	0.1006
1.8	0.0978	0.1009
1.7	0.0978	0.1009
1.6	0.0978	0.1009
1.5	0.2556	0.2581
1.4	0.5924	0.5938
1.3	0.8739	0.8741
1.2	0.9734	0.9728
1.1	0.9813	0.9803
1.0	0.9783	0.9781
0.9	0.9805	0.9799
0.8	0.9780	0.9767
0.7	0.9724	0.9711
0.6	0.9646	0.9637
0.5	0.9498	0.9516
0.4	0.9267	0.9322
0.3	0.8913	0.9004
0.2	0.6671	0.6759
0.1	0.3052	0.3050

Table 2
The variance of loss function.

Order(α)	0.9	1.0	1.1
Variance	0.65370	0.56961	0.57697

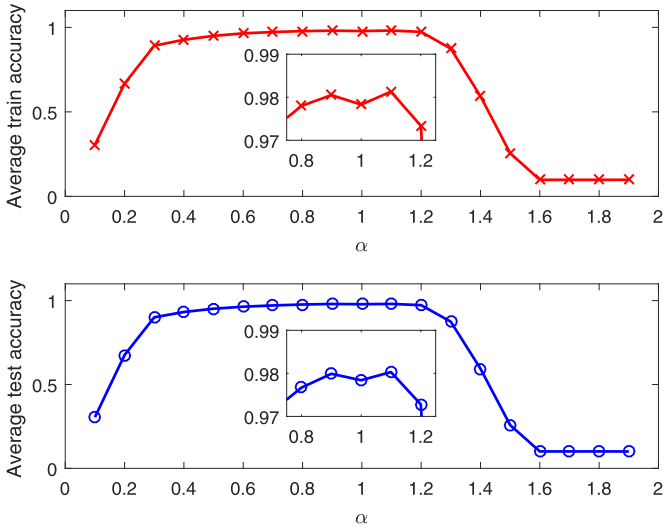


Fig. 7. Average accuracy of training and testing results.

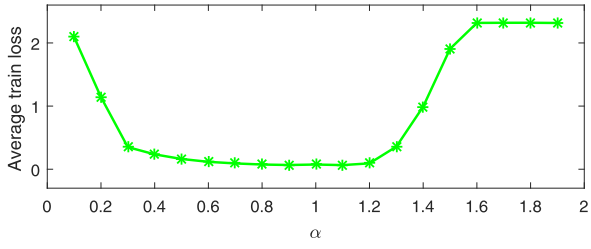


Fig. 8. Average loss of training results.

The corresponding parameters are listed below. $\hat{y}_s \in \mathbb{R}^{10}$ is the output of networks for the sth sample and $y_s \in \mathbb{R}^{10}$ is label with one-hot form.

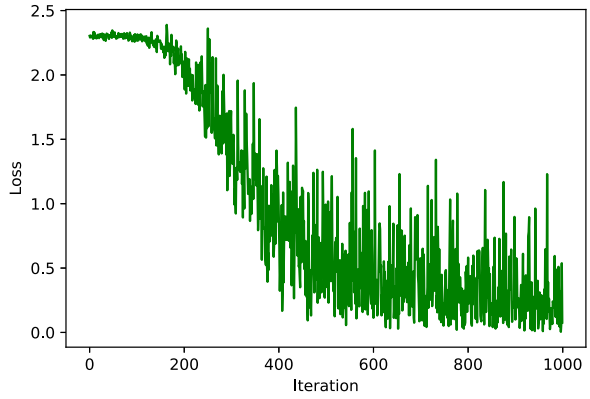
Loss function:	$L = -\frac{1}{m}\sum_{s=1}^{m} y_s^{\mathrm{T}} \log(\hat{y}_s)$
Learning rate:	$\mu = 0.1$
Batch size:	$m = 10$
Initial weight:	$w \in [-0.1, 0.1]$
Initial bias:	$b \in [-0.1, 0.1]$
Number of iteration:	$Iteration = 6000$
Number of epoch:	$Epoch = 1$

The experiments are carried out by 10 times. All parameters, such as weights, bias and the inputting order of samples, are randomly initialized each time. Consequently, the training accuracy and testing accuracy with different fractional order are shown in Table 1.

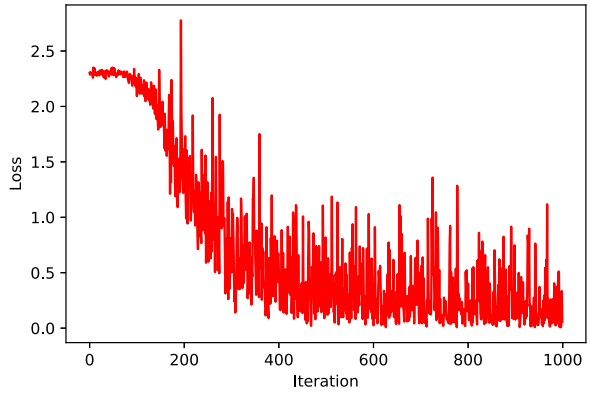
It could be observed that the accuracy of fractional order gradient methods with $\alpha = 0.9$ and $\alpha = 1.1$ is higher than the integer order gradient method in most cases. What's more, when average accuracy of 10 experiments is taken into consideration, the integer order one shows a little less accuracy. In Fig. 7, the average accuracy of training and testing results is drawn over $\alpha = 0.1, \ldots, 1.9$.

Although fractional order gradient method works well in CNN, it is not effective enough all the time when $\alpha < 0.3$ or $\alpha > 1.2$. The reason of such low accuracy is caused by the Gamma function in fractional order calculus (1). The Gamma function $\Gamma(2-\alpha)$ in fractional order gradient method (18,29) is a very large number for $\alpha < 0.3$ or $\alpha > 1.2$. As a result, the gradients are too small to reduce the loss function and sink into a local extreme point quickly. Since it is often a point close to the initial point, the loss does not decrease or only decreases a little bit. This phenomenon is also demonstrated by Fig. 8.

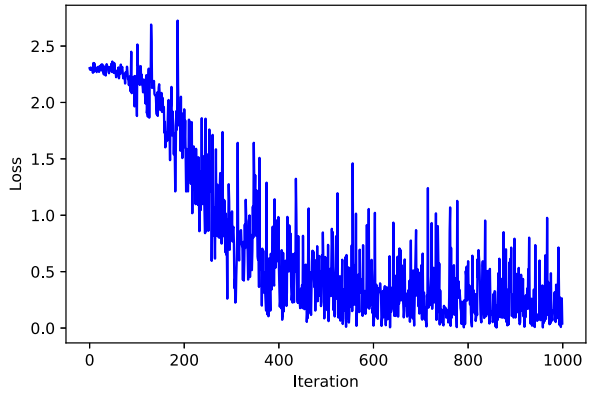
To analyze fractional order gradients further, Fig. 9 is drawn here to show the average loss function of first 1000 iterations. Even if all loss functions are decreasing during training, the loss iterated by fractional order gradients seems to prefer jumping farther. Com-



(a) $\alpha = 0.9$



(b) $\alpha = 1.0$



(c) $\alpha = 1.1$

Fig. 9. Loss functions for different order.

pared with integer order gradient method in Fig. 9(b), the points are more dispersed for fractional order cases, especially for $\alpha = 0.9$ in Fig. 9(a). In addition, each variance of different loss function is listed in Table 2.

The larger variance often indicates looser distribution, which implies that fractional order gradient method helps optimizing process jump frequently and far. Therefore, it is provided with more possibility to escape the local optimal point.

In view of seemingly complicated calculation, it seems that fractional order gradient method needs more time than integer order one. However, the training speed of fractional order CNN is almost as fast as integer order CNN. Taking all experiments into consideration, the average training time spent by integer order CNN is only 0.53% less than fractional order CNN with $\alpha = 1.1$. Similar speed also exists in other cases for $\alpha = 0.1, \ldots, 1.9$. There are two reasons that result in such fast speed of fractional order gradient method for CNN. One reason is that only updating gradients are replaced by fractional order. The other reason is that the fractional order updating gradients are obtained according to integer order gradients and the additional calculation in fractional order updating gradients are quite simple.

5. Conclusions

The backward propagation of neural networks is investigated by fractional order gradient method in this paper. After modification of fractional order gradient, the proposed gradient method can ensure the convergence to real extreme point, and has been successfully applied in CNN for updating parameters. It is the first time for CNN to cooperate with fractional order calculus. The chain rule in backward propagation is completely preserved since integer order gradients are still used for transferring between layers. Both the range of fractional order and the type of loss function are enlarged. Moreover, the proposed fractional order gradient method verifies its fast convergence, high accuracy and ability to escape local optimal point in neural networks when compared with integer order case. It is believed that this paper provides a new way to study gradient method and its application.

Declaration of Competing Interest

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled, ‘Convolutional neural networks with fractional order gradient method’.

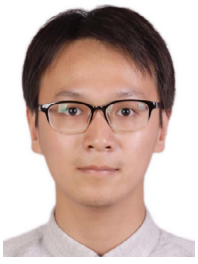
Acknowledgments

The work described in this paper was fully supported by the National Natural Science Foundation of China (No. 61573332, No. 61601431), the Fundamental Research Funds for the Central Universities (No. WK2100100028), the Anhui Provincial Natural Science Foundation (No. 1708085QF141) and the General Financial Grant from the China Postdoctoral Science Foundation (No. 2016M602032).

References

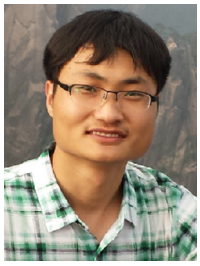
[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.
[2] K. Fukushima, Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, Biol. Cybern. 36 (4) (1980) 193–202.
[3] D.E. Rumelhart, J.L. McClelland, P.R. Group, et al., Parallel Distributed Processing, MIT press Cambridge, Boston, 1988.
[4] W. Alexander, T. Hanazawa, G. Hinton, S. Kiyohiro, K. Lang, Phoneme recognition using time-delay neural networks, Read. Speech Recognit. 1 (2) (1990) 393–404.
[5] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551.
[6] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[7] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: 2012 Neural Information Processing Systems (NIPS), Lake Tahoe, USA, 2012, pp. 1097–1105.
[8] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: 2015 International Conference on Learning Representations (ICLR), San Diego, USA, 2015, pp. 1–14.
[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, USA, 2015, pp. 1–9.

[10] K.M. He, X.Y. Zhang, S.Q. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, USA, 2016, pp. 770–778.
[11] M.A.Z. Raja, N.I. Chaudhary, Two-stage fractional least mean square identification algorithm for parameter estimation of carma systems, Signal Process. 107 (2015) 327–339.
[12] S.S. Cheng, Y.H. Wei, Y.Q. Chen, Y. Li, Y. Wang, An innovative fractional order lms based on variable initial value and gradient order, Signal Process. 133 (2017) 260–269.
[13] W.D. Yin, Y.H. Wei, T.Y. Liu, Y. Wang, A novel orthogonalized fractional order filtered-x normalized least mean squares algorithm for feedforward vibration rejection, Mech. Syst. Signal Process. 119 (2019) 138–154.
[14] S.S. Cheng, Y.H. Wei, D. Sheng, Y. Chen, Y. Wang, Identification for hammerstein nonlinear armax systems based on multi-innovation fractional order stochastic gradient, Signal Process. 142 (2018) 1–10.
[15] R.Z. Cui, Y.H. Wei, S.S. Cheng, Y. Wang, An innovative parameter estimation for fractional order systems with impulse noise, ISA Trans. 82 (2018) 120–129.
[16] Y. Li, Y.Q. Chen, I. Podlubny, Mittag–Leffler stability of fractional order nonlinear dynamic systems, Automatica 45 (8) (2009) 1965–1969.
[17] J.G. Lu, Y.Q. Chen, Robust stability and stabilization of fractional-order interval systems with the fractional order α: the $0 < \alpha < 1$ case, IEEE Trans. Autom. Control 55 (1) (2010) 152–158.
[18] C. Yin, Y.Q. Chen, S.M. Zhong, Fractional-order sliding mode based extremum seeking control of a class of nonlinear systems, Automatica 50 (12) (2014) 3173–3181.
[19] Y.H. Wei, B. Du, S.S. Cheng, Y. Wang, Fractional order systems time-optimal control and its application, J. Opt. Theory Appl. 174 (1) (2017) 122–138.
[20] Y.F. Pu, J.L. Zhou, Y. Zhang, N. Zhang, G. Huang, P. Siarry, Fractional extreme value adaptive training method: fractional steepest descent approach, IEEE Trans. Neural Netw. Learn. Syst. 26 (4) (2015) 653–662.
[21] C.A. Monje, Y.Q. Chen, B.M. Vinagre, D.Y. Xue, V. Feliu, Fractional-Order Systems and Controls: Fundamentals and Applications, Springer, London, 2010.
[22] Y.Q. Chen, Q. Gao, Y.H. Wei, Y. Wang, Study on fractional order gradient methods, Appl. Math. Comput. 314 (2017) 310–321.
[23] Y.Q. Chen, Y.H. Wei, Y. Wang, Y.Q. Chen, Fractional order gradient methods for a general class of convex functions, in: 2018 Annual American Control Conference (ACC), Milwaukee, USA, 2018, pp. 3763–3767.
[24] J. Wang, Y.Q. Wen, Y.D. Gou, Z.Y. Ye, H. Chen, Fractional-order gradient descent learning of bp neural networks with caputo derivative, Neural Netw. 89 (2017) 19–30.
[25] C.H. Bao, Y.F. Pu, Z. Yi, Fractional-order deep backpropagation neural network, Comput. Intell. Neurosci. 2018 (2018) 1–10.
[26] D.E. Rumelhart, G.E. Hinton, R.J. Williams, et al., Learning representations by back-propagating errors, Nature 323 (1986) 533–536.



Dian Sheng received his B.E. degree in Detection Guidance and Control from Nanjing University of Aeronautics and Astronautics, in 2015. He is currently pursing the Ph.D. degree in control science and engineering with University of Science and Technology of China, Hefei, China. His current research interests include the fractional order systems, backstepping control and gradient method.

Yiheng Wei received his B.E. and Ph.D. degrees from the Northeast University and the University of Science and Technology of China in 2010 and 2015, respectively. He was a postdoctoral fellow of the City University of Hong Kong and the University of Science and Technology of China from 2015 to 2017. Dr. Wei is currently a research associate professor in Department of Automation of the University of Science and Technology of China. His research interests include fractional order system theory, signal processing and deep learning.



Yuquan Cheng received his B.E. degree from the Department of Automation, University of Science and Technology of China, in 2014. He is currently pursuing the Ph.D. degree in control science and engineering with University of Science and Technology of China, Hefei, China. His current research interests include the fractional order adaptive control and gradient method.

Yong Wang received his B.E. degree in Automatic Control from the University of Science and Technology of China, Hefei, China, in 1982, and the M.E. and Ph.D. degrees in Navigation, Guidance and Control from Nanjing Aeronautical Institute, Nanjing, China.

Since 2001, he has been with the Department of Automation, University of Science and Technology of China, where he is currently a professor and Deputy Dean of the School of Information Science and Technology. His research interests include fractional order systems, new aircrafts, active vibration control and robot control.