ICIC Express Letters
Volume **10**, Number **6**, June **2016**

 ISSN 1881-803X
pp. **1399–1405**

A NEW RAPID CONJUGATE GRADIENT ALGORITHM FOR CONVOLUTIONAL NEURAL NETWORKS

Chao Zhang, Xi Peng, Yong Zhou and Zhenbao Liu

School of Aeronautics
Northwestern Polytechnical University
No. 127, West Youyi Road, Xi'an 710072, P. R. China
caec_zc@nwpu.edu.cn

Received November 2015; accepted February 2016

Abstract. *Convolutional neural network (CNN) is a typical architecture of deep neural networks. However, for most gradient descent algorithm of CNN, the convergence speed is very slow and is easy to fall into local minimum point. In this paper, a new efficient algorithm called the rapid conjugate gradient (RCG) algorithm was proposed to update the weights of CNN and to improve the performance of CNN. Two test experiments were used to validate the RCG algorithm, and the comparison results show that the RCG algorithm has a better convergence speed and performance than the PRP, FR, DY and GD algorithms.*
Keywords: Deep neural network, Convolutional neural networks, Rapid conjugate gradient algorithm, Weight update

1. **Introduction.** Convolutional neural network proposed by Hinton et al. [1] in 2006 is widely used in image processing, speech processing and pattern recognition, etc. Various methods have been proposed to improve the performance of CNN. Eigen et al. [2] discuss three variables' influences on the network: the numbers of layers, feature maps and parameters. The experiment result shows the number of layers and parameters affects more than the number of feature maps. Adding the layers and parameters makes the network perform better. Farabet et al. [3] use hardware architecture to implement large-scale convolutional neural networks and state-of-the-art multi-layered vision systems. It makes a comparison between software, FPGA and ASIC implementation. The result shows a speedup in hardware implementations. Mathieu et al. [4] compute convolutions as point-wise products in the Fourier domain while reusing the same transformed feature map many times. The algorithm accelerates training and inference by a significant factor and can yield improvements of over an order of magnitude compared to existing state-of-the-art implementations.

In most researches, the gradient descent algorithm is used to update the weights and bias. However, its convergence speed is very slow, and it is easy to fall into local minimum point. Many researches are done to solve the problem. Some of them advise to use a variable learning rate or add a momentum to the weight update formula. It improves the convergence speed largely, but it still performs worse than other algorithms, such as the conjugate gradient algorithm. The conjugate gradient algorithm can get twice speed than the classic gradient descent algorithm. The most important step of the conjugate gradient algorithm is the update of search direction. There are many algorithms to compute the search direction, such as the PRP, FR and DY. However, their computations are a little complicated.

This paper proposes a rapid conjugate gradient algorithm to simplify the calculation and speed up the convergence process. Convolutional neural network is chosen as the learning model for image classification. The study object is the MNIST database. Several kinds

of weight updating algorithm, including the BP algorithm and several conjugate gradient algorithms (PRP, FR and DY), are compared with our RCG algorithm. By comparing the results of experiments, it shows that our algorithm has a better convergence speed in training process.

The remainder of this paper is organized as follows: the architecture and training steps of the convolutional neural networks are presented in Section 2; the theory of the rapid conjugate gradient algorithm will be presented in Section 3; Section 4 shows the performance of the proposed techniques in classification and makes a comparison. Section 5 concludes this paper.

2. **Convolutional Neural Networks.** Convolutional neural networks combine three architectural ideas to ensure a good result: local receptive fields, shared weights and spatial sub-sampling. With local receptive fields, neurons can extract elementary visual features such as oriented edges, end-points, and corners. These features are then combined by the subsequent layers in order to detect high order features. The weight sharing technique has an obvious effect on reducing the number of parameters, thereby reducing the capacity of the machine and reducing the gap between testing error and training error. The sub-sampling technique reduces the resolution of the feature map and the sensitivity of the output to shifts and distortions.

Convolutional neural network is composed of three layers: a filter bank layer, a non-linearity layer, and a feature pooling layer [5]. A typical convolutional neural network's architecture is showed in Figure 1. The network has two convolution layers, two sub-sampling layers, two full connection layers and a Gaussian connection layer. The first convolution layer has 6 feature maps with 5×5 convolution filters. The C1 layer size is $(32 - 5 + 1) \times (32 - 5 + 1)$. Next layer S2 is the subsampling layer, and it has the same number feature maps as C1. We apply a 2×2 mean pooling or max pooling to reducing the parameters. C3 and S4 consist of 16 feature maps, and all the features of S4 are full connected to the C5. The C5 with 120 neurons is also full connected to the F6 with 84 neurons. The output layer consists of the same number of units as the number of classes, and each unit is connected to the F6.

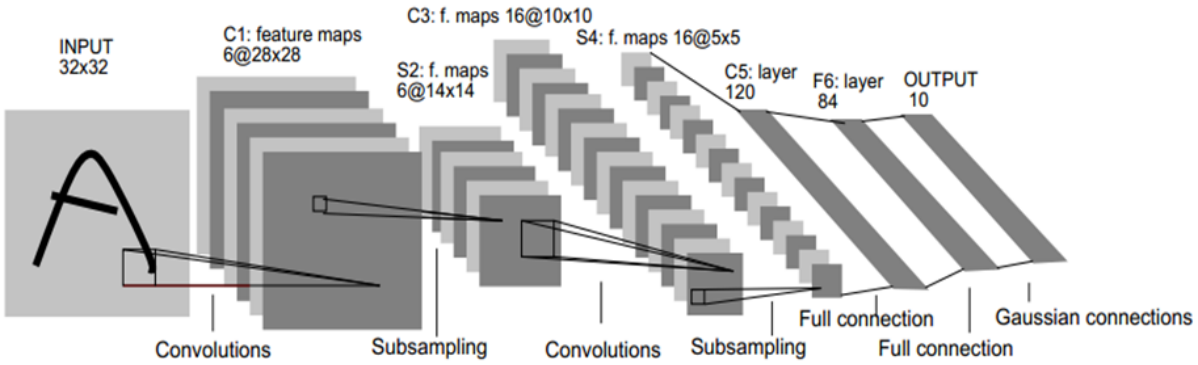

FIGURE 1. The architecture of CNN [5]

To train a CNN, we should follow these steps, as shown in Figure 2. At first, we should set the network's architecture, including the number of convolution layers, the size of filters and the number of feature maps. We should set parameters' starting value before computation. In forward feedback, we put a batch of data into the network and calculate the output of each layer one by one. The output of this layer is the input of next layer. In back propagation, we put the output of the network into the last layer and calculate each layer reversely. We compare each layer's output in forward feedback and back propagation, and calculate the deviation. Then we can update the weight according to these deviations.

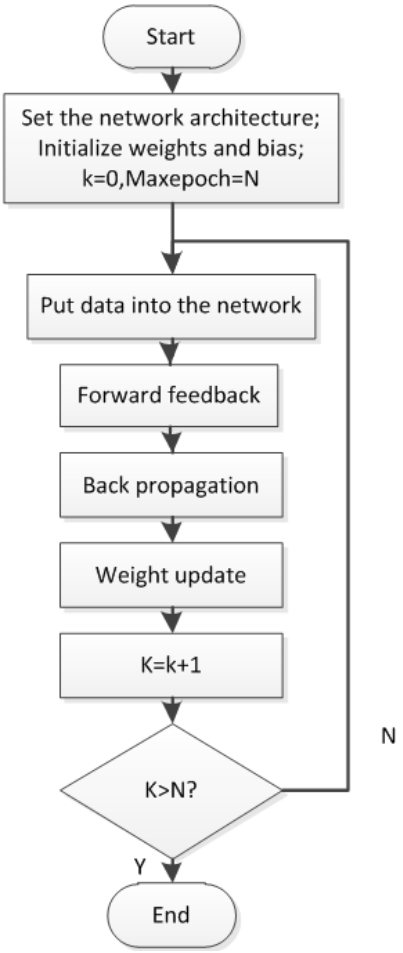

FIGURE 2. The training steps of CNN

3. **The Proposed Rapid Conjugate Gradient Algorithm.** The most common weight update algorithm is the gradient descent algorithm [7], but it is not a fast method in theory. Experts proposed many methods to improve it, such as the variable learning rate learning algorithm [8], the Newton method, the Hessian-free method, the LBFGS, the Levenberg-Marquardt method and the conjugate gradient method [9]. We choose the conjugate gradient algorithm as the research target and make an improvement.

The RCG algorithm is aimed to solve the nonlinear optimization. It can achieve a fast speed by adjusting the weights and bias along the conjugate gradient direction just as the scaled conjugate gradient does. And remember the first step iterative of the RCG algorithm must start from the steepest descent gradient direction. Then the update of weight or bias is based on the following rule:

$$d_0 = -g_0 \tag{1}$$

$$x_{k+1} = x_k + a_k d_k \tag{2}$$

$$d_k = -g_k + b_k d_{k-1} \tag{3}$$

g_0 is the first gradient direction; d_k is the search direction. x_k is the current weight or bias, and x_{k+1} is the updated weight or bias. a_k is the step size. g_k is the current gradient direction, and b_k is the coefficient of d_{k-1}. The weight update process is showed in Figure 3. After back propagation, we calculate the weight gradient g_k. The first search direction must start from the steepest descent gradient direction. Beyond that, the search direction is updated according to Formula (3). b_k is updated according to current gradient and last search direction. Then we can update the weight according to Formula (2).

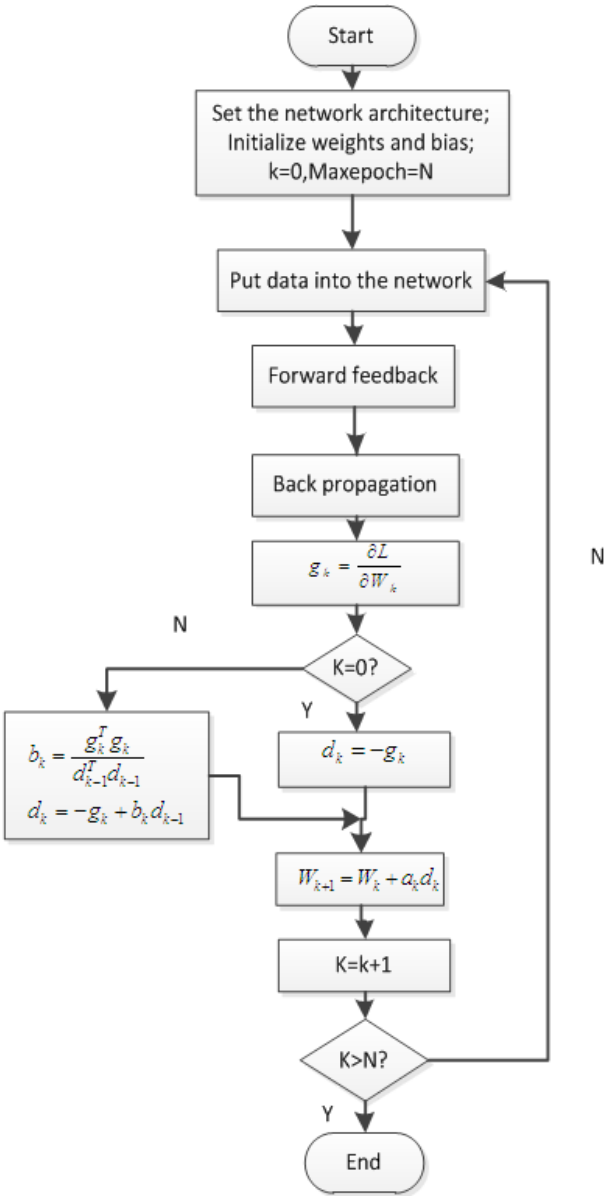

FIGURE 3. Weight update process

TABLE 1. Several kinds of b_k

Name	Structure
PRP	$b_k = \frac{g_k^T(g_k - g_{k-1})}{g_{k-1}^T g_{k-1}}$
FR	$b_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}}$
DY	$b_k = \frac{g_k^T g_k}{d_{k-1}^T(g_k - g_{k-1})}$
RCG	$b_k = \frac{g_k^T g_k}{d_{k-1}^T d_{k-1}}$

Different conjugate gradient algorithms correspond to different choices of the b_k. However, all are selected so that when applied to minimizing a strongly quadratic convex function, the two contiguous search directions are conjugate. Different structures of b_k have different effects on the final results. Several classic structures of b_k are as shown in Table 1.

4. **Experimental Results and Comparison.** We perform our experiments on a simple nonlinear function and the MNIST database to evaluate the developed rapid conjugate gradient algorithm. The MNIST dataset consists of 28 × 28 digit images: 60000 for training and 10000 for testing. The objective is to classify the digit images into their correct digit class.

4.1. **Case 1: nonlinear function.** We validate our algorithm in a nonlinear quadratic function. The target function is as follows

$$f = 0.8x_1^2 + 0.5x_2^2 \tag{4}$$

To make the function have the minimum value in restricted conditions, which we can guess is 0, the problem is equal to calculating following equation

$$\min f(x) = f(x^*) \tag{5}$$

where x is $[x_1, x_2]$, and x^* is the optimal solution under the expected accuracy.

Let us set the start value $x_0 = [-9.8, -4]$, calculate the function in four algorithms and repeat 10 times, respectively. The process and results are as Figure 4 and Table 2. From the figure and table, we can see that the RCG algorithm reaches the target value more rapidly than other three algorithms. The RCG algorithm performs well in solving simple nonlinear optimization, while how about solving complicated problems in practice? We will discuss next.

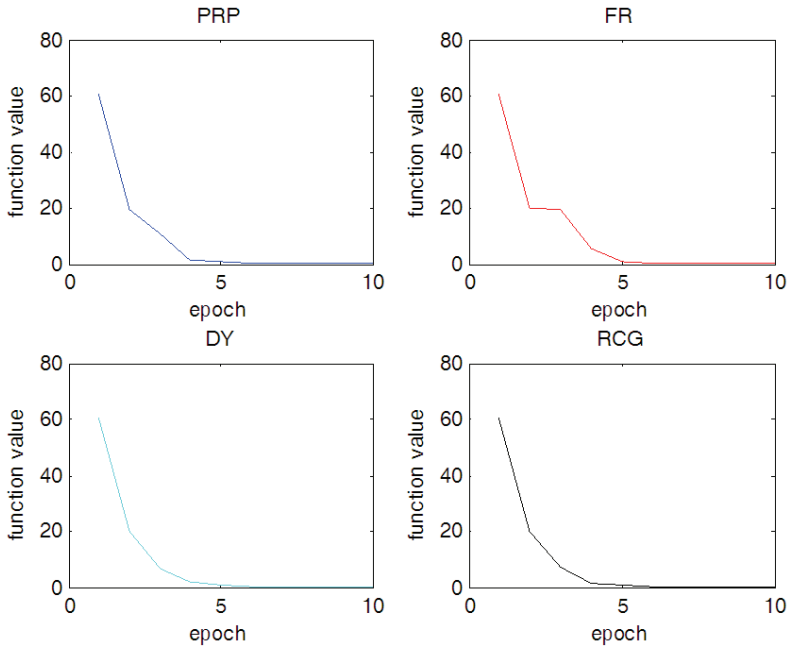

FIGURE 4. The progress of four algorithms in case 1

TABLE 2. The results of four algorithms

Name	Results
PRP	0.0018
FR	2.3597e-04
DY	9.0000e-04
RCG	2.8187e-05

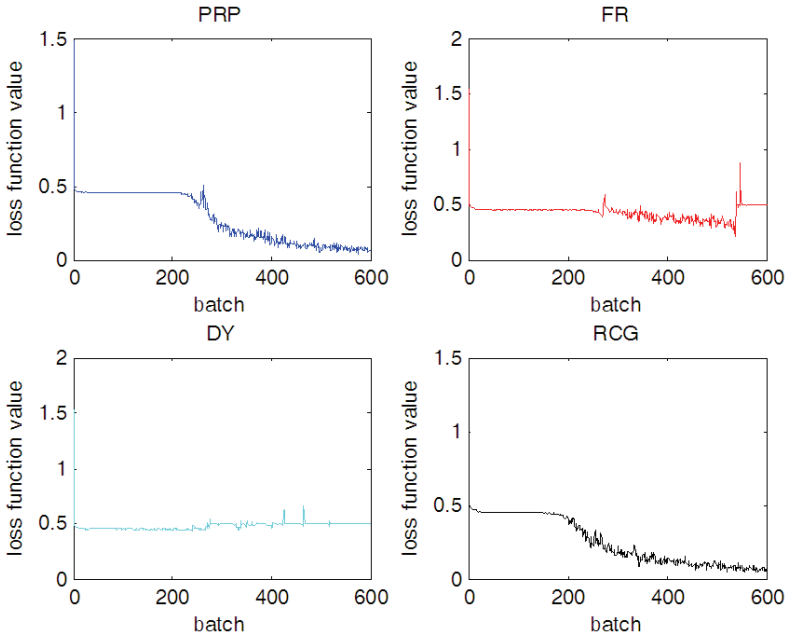

FIGURE 5. The progress of four algorithms in case 2

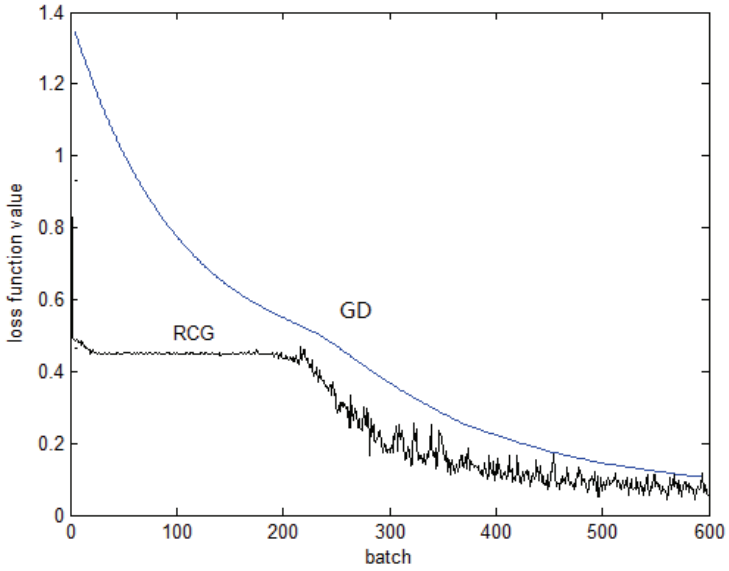

FIGURE 6. The progress of two algorithms

4.2. **Case 2: results on MNIST.** When applied in convolutional neural networks, the RCG algorithm also has an obvious advantage to other algorithms. We use the MNIST handwritten digit base as the test object to verify our theory. In this test, we choose a 6c-2s-12c-2s classic convolutional neural network structure. The step size a_k in $x_{k+1} = x_k + a_k d_k$ is the learning rate, which has an invariant value of 1.15. We devide the whole training data into batches, each batch has 100 data, and then we have 600 batches to train. The training process is as Figure 5, and the RCG algorithm shows a more smooth and rapid convergence than other three algorithms.

The conjugate gradient algorithm is a second-order method. We compare it with some linear algorithms such as the gradient descent (GD) algorithm, as Figure 6. In Figure 6, the rapid conjugate gradient algorithm has a more obvious descent than gradient algorithm at the beginning. The X axis is the training batch number, and the Y axis is the loss function value. The RCG reaches a relative low value after a few training batches, rapidly.

Then the convergence slows down for a while until it drops again with a rapid speed. However, the GD algorithm drops gradually with a middle speed.

5. **Conclusions.** In this paper, we propose a new rapid conjugate gradient method to update the weights of CNN for image classification. Compared with the gradient descent algorithm and other conjugate gradient methods, our algorithm is easier to achieve the twice convergence speed. It shows great potential in solving more complicated problems. However, it cannot deal with the unlabeled data because the convolutional neural network is a supervised learning algorithm. It is essential to find an unsupervised algorithm to solve some actual problems. Deep belief network (DBN) is a well-known neural network of unsupervised learning, which is presented by Hinton. Our future work is to apply the RCG algorithm to the DBN and solve some more complicated problems.

Acknowledgement. This work was supported by the National Natural Science Foundation of China (Nos. 61104030 and 51207129), the Natural Science Basic Research Plan in Shaanxi Province of China (No. 2015JM6302) and the Fundamental Research Funds for the Central Universities (No. 310201401JCQ01018). The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

REFERENCES

[1] G. E. Hinton, S. Osindero and Y. Teh, A fast learning algorithm for deep belief nets, *Neural Computation*, vol.18, pp.1527-1554, 2006.

[2] D. Eigen, J. Rolfe, R. Fergus and Y. LeCun, Understanding deep architectures using a recursive convolutional network, *International Conference on Learning Representations*, arXiv: 1312.1847, 2014.

[3] C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun and E. Culurciello, Hardware accelerated convolutional neural networks for synthetic vision systems, *IEEE International Symposium on Circuits & Systems*, vol.54, no.3, pp.257-260, 2010.

[4] M. Mathieu, M. Henaff and Y. LeCun, Fast training of convolutional networks through FFTs, *International Conference on Learning Representations*, arXiv: 1312.5851, 2014.

[5] Y. LeCun, K. Kavukcuoglu and C. Farabet, Convolutional networks and application in vision, *Proc. of International Symposium on Circuits and Systems*, pp.253-256, 2010.

[6] J. Bouvrie, *Notes on Convolutional Neural Networks*, 2006.

[7] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, Gradient-based learning applied to document, *Proc. of the IEEE*, vol.86, no.11, pp.2278-2324, 1998.

[8] C. Si, Learning rate parameter improve methods for BP neutral network, *Journal of Changchun Normal University (Natural Science)*, vol.29, no.1, pp.26-28, 2010.

[9] I. E. Livieris and P. Pintelas, An improved spectral conjugate gradient neural network training algorithm, *International Journal on Artificial Intelligence Tools*, vol.21, no.1, 2012.