Parallelizing Convolutional Neural Network for the Handwriting Recognition Problems with Different Architectures

Junhao Zhou
computer science
Macau university of science and technology
Macao, China
776253638@qq.com

Weibin Chen
computer science
Politecnico Di Milano
Milan, Italy
635772426@qq.com

Guishen Peng
computer science
Guangdong University of Technology
Guangzhou, China
2209899642@qq.com

Hong Xiao
computer science
Guangdong University of Technology
Guangzhou, China
wh_red@163.com

Hao Wang
computer science
Norwegian University of Science and Technology
Ålesund, Norway
iswanghao@acm.org

Zhigang Chen
computer science
Guangdong University of Technology
Guangzhou, China
569554269@qq.com

***Abstract*— As the convolutional neural network (CNN) algorithm is proved to be uncomplicated in the image preconditioning and relatively simple train the original image, it has become popular in image classification. Apart from the field of image classification, CNN has been widely used in many scientific area, especially in the field of pattern classification. In this paper, we use CNN for handwritten numeral recognition. The basic idea of our method is to use the multi-process to process the training samples in parallel, to exchange the training results and to get the final weight parameters. Compared with the conventional algorithm, the training time is greatly reduced, and the result can be obtained more quickly. Besides, the accuracy of the algorithm is proved to be almost the same as that of the conventional algorithm with sufficient training testing samples.. This significantly improves the efficiency of CNN in the hand written numeral recognition. Finally, we also implemented our proposed method with parallel acceleration optimization based on Many Integrated Core Architecture (MIC) architecture of Intel and GPU architecture of Nvidia.**

Keywords*— *Convolutional Neural Network, Handwritten Recognition, Parallel, Multi-Process, MIC, GPU

I. INTRODUCTION

Convolutional neural networks (CNNs) have made great improvements in object recognize for several years. In order to evaluate the performance and accuracy of the image recognition algorithms, the researchers establish an image database called ImageNet. CNN methods can be used many area, such as license plate recognition [8], face recognition, depth convolution network [7]. In particular, we apply CNN methods in handwritten font recognition.

In this paper, we improve the performance of CNN methods by using shared weights of convolutional neural networks. To ensure the accuracy, the conventional algorithm requires a larger number of training samples, which requires a longer training time. In this paper, we guarantee the accuracy by using a large number of training samples.

Besides, we optimize convolution neural network [9] [4] based on parallel acceleration optimization. Our method can greatly shorten the training time and improve the algorithm performance.

The remainder of this paper is organised as follows: Section 2 presents related work for Handwriting. Section 3 describes convolutional neural networks algorithm. Section 4 presents our improved convolutional neural networks algorithm in parallel. Experimental results with Handwriting are presented in Section 5. We conclude and point to our future work in Section 6.

II. RELATED WORK

In machine learning, a convolutional neural network (CNN, or ConvNet) is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex. In the

Guangdong application type science and technology and development special funds project. Base on terahertz imaging technology of urban rail transit security equipment delay. (NO. 2015B090923004)

978-1-5386-1978-0/17/$31.00 ©2017 IEEE

1960s, Hubel and Wiesel found that their unique network structure can effectively reduce the complexity of the feedback neural network in the study of neurons with local sensitivity and direction selection in the cats of cats, which makes Convolutional Neural Networks (CNN) invented. In 2012 , the convolutional neural network algorithm used to obtain excellent results. Since then, the convolutional neural network has become the mainstream of this competition. At present, more and more researchers study CNN. This paper is based on the research of CNN [10], which was written in C language by TOSTQ. The authors improved the accuracy of MNIST (THE MNIST DATABASE of hand written digits) by CNN. However, the program is not efficiently, because it is based on single process of CPU, and it costs substantial of running time. In this paper, we exploit parallel computing to reduce the time consumption. We next explain the preliminaries of parallel computing as follows.

A. Data Parallelism

In the MPI technology, the data set is divided into several subsets, which are then assigned to multiple processes for parallel processing.(see Figure 1)

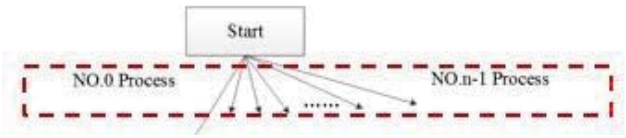

Fig. 1. Multiple Processes.

B. Thread Parallelism

OpenMP use parallel convolution kernel threads. Since the convolution process of each convolution kernels is not relevant, it can be paralleled in a thread.

C. Activation Function Optimization

Instead of sigma activation function in the original algorithm of CNN, we used ReLu (Rectified Linear Units) function. It can avoid disappearance of the weight of convolution kernel at last.

III. Algorithm Principle

A. Convolutional Neural Network

Convolutional neural network (CNN) is a kind of common deep learning architectures. It is inspired by biological natural visual cognition mechanism and now it is widely used in pattern recognition and image processing with less training parameters and strong adaptability. Its weight sharing network structure makes it more similar to biological neural network, which reduces the complexity of network model and reduces the weight quantity. The advantage of this method is that the input of the network is obvious when the input is multidimensional and the image can be directly inserted into the network. In this manner, it can avoid the complex feature extraction and data-reconstruction in the traditional recognition algorithms. A convolution network is a multilayer perceptron, which is specially designed to recognize a two-dimensional shape that is highly invariant to translational, scale, tilt, or other forms of deformation. Convolutional neural networks consist of one or more convolutions and topmost fully connected layers (corresponding to classical neural networks), and also include a pooling layer of associated weights. This structure allows the convolutional neural network to take advantage of the two-dimensional structure of the input data. As it is shown in Figure 1, compared with convolution neural network, it can provide better result in image and speech recognition than other depth learning structures.

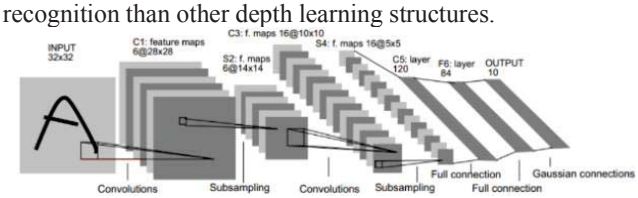

Fig. 2. LeNet-5 Network.

1) Convolution layer

The convolution of the gray-scale image with the template (convolution kernel) yields is a convolved layer image, which is obtained by an activation function as follows

$$I_j^{C_x}=\delta\left(\sum_{i=1}^{n} W_{ij} * I_i^{S_{x-1}}+b_j^{C_x}\right)$$
$$j=1\cdots m$$

where I is the image, W is the convolution kernel, b is the bias weight, δ is the activation function, i and j are the input and output image numbers respectively, x is the current layer number, and ∗ is the convolution operation. δ is the activation function,usually sigmoid function.

$$\delta=\frac{1}{1+e^{x}}$$

2) Pooling layer

The function of the pooling layer is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to control overfitting.

3) Output layer

It is usually a single-layer fully-connected neural network, and the number of output layer neurons is determined by actual situation.

4) Forward propagation

Actually, it refers to the input image data and the outputs of the process. The main operations involved are convolution and down-sampling operations. The convolution formula of the image is as follows,

$$g(x,y)=\sum_{c=0}^{c}\sum_{r=0}^{r} w(s,t)\cdot f\left(x-s-\frac{c-1}{2},y-t-\frac{r-1}{2}\right)$$
$$C,R=2K+1$$

5) Back propagation

(also named backward propagation of errors). It is a common method of training convolutional neural network used in conjunction with an optimization method such as gradient descent. Back propagation is a reflection of neural network training and learning process given as follows,

$$E_e^n=\frac{1}{2}\sum_{i}^{N}(d_i-y_i)^2$$

Error E is defined as the error between the actual output and the ideal output

$$\Delta W^n=\eta\frac{\partial E_e^n}{\partial W^n}=\eta\frac{\partial E_e^n}{\partial u^n}\frac{\partial u^n}{\partial W^n}=\eta\delta^n X^n$$
$$u^n=W^n X^n$$
$$\delta^n=\frac{\partial E_e^n}{\partial u^n}$$

6) Weight update

The weight adjustment gradient are obtained by back propagation as follows,

$$W^{n+1}=W^n-\Delta W^n$$

B. MNIST Handwritten Digital Database

This paper mainly investigates the optimization of the convolutional neural network (CNN) that is applied to the MNIST. In order to apply the neural network in recognition of handwritten image, lots of training samples are used to training the neural network. MNIST is a handwritten digital database consisting of 60,000 training sample sets and 10,000 test sample sets. It is a subset of the NIST database. The file in MNIST is not a standard image format. These image data are stored in a binary file with 28*28 of width and height of each sample image.

C. CNN Handwriting Recognition

1) Algorithm Description

Convolution neural network through local perception and weight sharing greatly reduces the parameters between the neurons so that the training speed has greatly accelerated. Neural network is trained to obtain the output value by calculating the data. The error gradient of each layer in the current network can be obtained after the back propagation through the network. Finally, the weight of the network is adjusted according to the error gradient. After many times of repetition, final training network model is available. The following is pseudo-code algorithm:

Algorithm 1: Convolution Neural Network Algorithm

```
Data: MNIST handwritten digit database
Result: Neural Network model
Initialize the structure of CNN and all related parameters;
Loading image data from MNIST database;
for  Image in Data do
        Forward propagation;
        Back propagation;
        Update weight;
end
Save the CNN model and test accuracy;
return accuracy;
```

2) Algorithm Flow Chart as shown in Figure 3

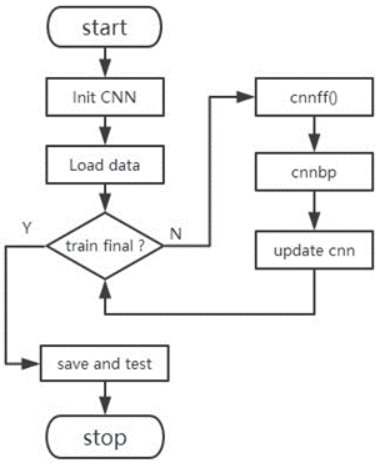

Fig. 3. Algorithm flow chart

3) Main Function Description

We next describe main function as follows.

Cnnsetup(CNN* cnn, inputSize, outputSize): Initialize the CNN
- Parameter: CNN*. Custom CNN structure
- Parameter: nSize. Custom image specification structure
- Parameter: outputSize. Image output specification

Read_img(filename, rank, process_num):Load image data
- Parameter: rank. Sequence number of process
- Parameter: process_num. Number of processes

Cnnff(CNN*cnn, inputData): Forward propagation

Cnnbp(CNN*cnn, outputData): Back propagation

Cnnupdate(CNN* cnn, opts, inputData): Weight update
- Parameter: CNNOpts. Custom training parameters

Cnntest(CNN* cnn, inputData, outputData, testNum): Test training results

IV. Parallel Computing Optimization

A. Parallel Technology Introduction

MPI. Message Passing Interface (MPI) is a cross-language communication protocol for writing parallel programs MPI supports for point-to-point and broadcast and is an information delivery application program interface [6], including protocol and semantic descriptions. The strengths of MPI are highperformance, large-scale, and portability. Today it is still the main model of high-performance computing. Although MPI belongs to the OSI reference model of the fifth layer or above, its implementation may cover most of the layers through the transport layer's socket and Transmission Control Protocol (TCP). Furthermore, most MPI implementations consist of some specified set of API, which can be called directly from C, C++, Fortran, or languages with such libraries, such as C#, Java, or Python.

MPI is better than the old-style information transfer library regarding its portability and speed. It is able to support dynamic processes, parallel I/O, and remote storage access from version 1.2. In the application platform, it has been implemented on IBM PCs, MS Windows, on all major Unix workstations, and on all major parallel machines. In addition, C

or Fortran parallel programs using MPI for message passing can run unchanged on IBM PC, MS Windows, Unix workstations, and a variety of parallel machines.

OpenMP. OpenMP [1], invented by the OpenMP Architecture Review Board, has been widely accepted, which is used as a set of instructional compiling solutions for multiprocessor programming of shared memory parallel system. OpenMP [2]supports programming languages including C, C ++, and Fortran. And also it supports programming languages including C, C++, and Fortran [2]. It provides a high-level, abstract description of a parallel algorithm, where programmers specify their intent by adding a dedicated pragma to the source code, so that the compiler can automatically parallelize the program and add synchronization mutexes and communication where necessary. When these pragma as are chose to ignore, or the compiler does not support OpenMP, the program can degenerate into the usual program (usually serial). The code can still work, but cannot use multi-thread to speed up program execution

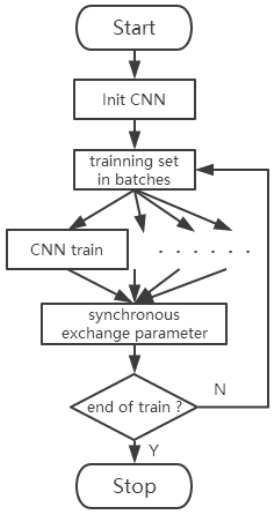

Fig. 4. Parallel calculation flow chart

B. Parallel Optimization

We next show the parallel optimization as the following procedure.

1) Data calculation Parallel by MPI [3] [5]

(1) The training sample set is roughly divided into k sub-training sets with N scale, and these sets were transmitted in different processes by MPI. After completing the experiment 20 times, we find when k=75 and N= 800, the result can reach the peak of acceleration ratio.

(2) Then, each process will train its own sub training set, after the end of training by the MPI reduction function, it reduces the results of each process and exchange the weight parameter.

(3) After the parameter exchanging, all training processes enter into the next round of training of the new weight model until the sample training is completed.

(4) Finally, we test the accuracy of the network model as shown in Figure 4.

2) The Same Level of Feature Map in Parallel by OpenMP

Convolution neural network is generally composed of multiple layers. Each layer contains multiple feature maps. Each feature map is generated from the previous layer by convolution or down-sampling. Owing to each feature map is not related to each other. Thus generating of the same Layer feature map can be used in parallel manner by OpenMP. Through this technology, so as to achieve the thread level parallel acceleration. (see Figure 5)

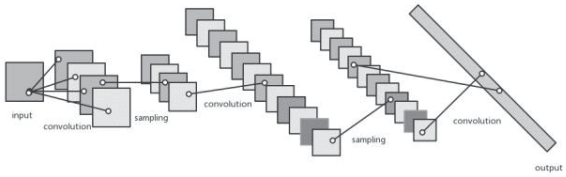

FIGURE5. PARALLEL GRAPH OF FEATURE

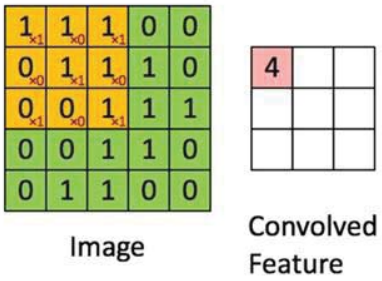

Fig. 6. Convolution window sliding parallel

3) Parallel in convolution sliding window by OpenMP

Convolution is an important operation in the process of convolution neural network training. And the convolution kernel generates the next layer feature map by sliding the feature map on the upper layer. When the convolution kernel generates the next layer of the feature map, the individual pixels are independent of each other. Since each convolution kernel is not related to generation. Therefore it can be parallel by OpenMP, and generate the convolutional kernel simultaneously in threads. It is shown in see Figure 5 and see Figure 6.

4) Calculating neuron weights in parallel

In neural network training process, there are a large number of weight calculation operations between neurons. The neurons in the same layer are only related to the neurons in the upper layer, which means the neurons in the same layer can be calculated in parallel. In the convolutional neural network, the weights correspond to the values on the convolution kernel, and the pixels of the two-layer feature map correspond to the values of the neurons between two adjacent layers.(see Figure 7)

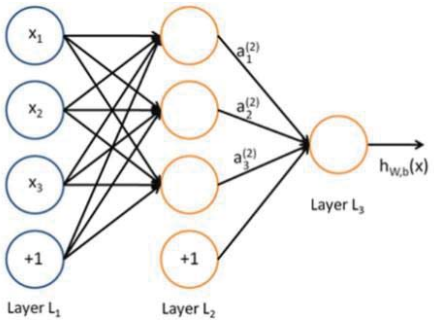

Fig. 7. Parallel calculate neuron weights

5) ReLU (Rectified Linear Units)

The original activation function in CNN algorithm was sigmoid function which cost large calculation at the back propagation stage. It need to find the deviation gradient. However, instead of sigmoid, this paper uses ReLU (Rectified Linear Units) as an activation function at output layer. This activation function is first introduced to a dynamical network by Hahnloser et al. in a 2000 paper in Nature with strong biological motivations and mathematical justifications. Based on principle of ReLU, it can save a lot of computation of source. On the other hand, in deep layers, sigmoid function will disappear its gradient when it at back propagation stage, thus it cannot complete the training in deep layers.

In the context of artificial neural networks, the rectifier is an activation function defined as

$$f(x) = max(0,x)$$

where x is the input to a neuron. This is also known as a ramp function and is analogous to half-wave rectification in electrical engineering.

V. Experimental Results And Analysis

C. Introduction of Architecture

- Many Integrated Core Architecture (MIC)

The Many Integrated Core Architecture is a heterogeneous many-core architecture introduced by Intel R. MIC can be configured in three modes, (1) Native mode, the program runs only on the MIC; (2) Offload mode, CPU need to be calculated the portion of the MIC card OFFLOAD to perform parallel computing; (3) Peer mode, procedures are applicable to more complicated process, such as MPI program in the CPU and main function MIC terminal simultaneously initiated. Concurrently, parallel Simulated Annealing Algorithm (PSAA) in our lab is implemented in Native mode. [11]

- Graphics Processing Unit

A graphics processing unit (GPU), occasionally called visual processing unit (VPU), is a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display. Modern GPUs are very efficient in manipulating computer graphics and image processing, and their highly parallel structure makes them more efficient than general purpose CPUs for algorithms where the processing of large blocks of data is done in parallel.

- Caffe

Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by the Berkeley Vision and Learning Center (BVLC) and by community contributors. Caffe integrates the frame of CNN, so it is suitable for users to study Deep Learning. And the most important is that it can be accelerated by GPU.

D. Experimental Environment

–Processor: Intel R Xeon R CPU E5-2692 v2 @ 2.10GHz;

–Memory (RAM): 63GB;

–Operating systems: Red Hat Enterprise Linux Server release 6.3 (Santiago);

– Coprocessor: Intel R Xeon Phi R coprocessor 5110P; – Coprocessor memory: 8GB.

–GPU: GTX 1080

E. Experimental results

- Based on MIC and CPU

Optimization by MPI results under Tianhe No.2 are shown in the following figure. In Figure 8, at low error rates (approximate 0.1), the cost of time is reduced (from 65 minutes to 1 minute) by the increase in the number of processes (see Figure 9). In Figure 10, the abscissa represents the number of processes and the ordinate represents the speedup. Finally, under the 75 process, the optimal speedup is 65.9 times.

- Based on GPU and CPU

In environment of GPU, the algorithm performs more efficiently benefited by the frame of Caffe (see Figure 11). In the test, it spends 0.75s with an accuracy of 0.996 (see Table 1).

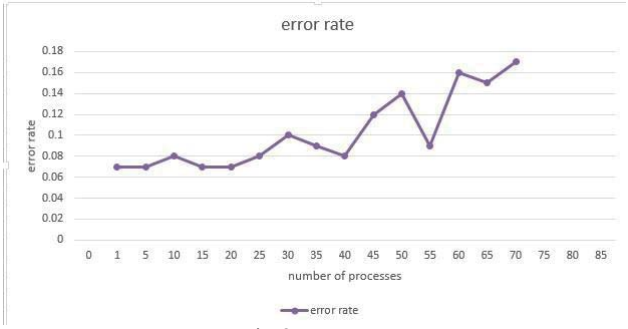

Fig. 8. Error rate

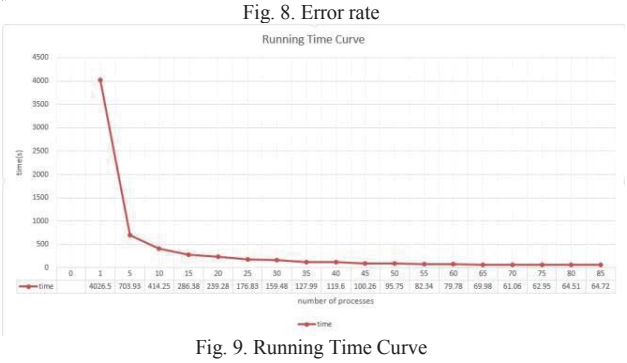

Fig. 9. Running Time Curve

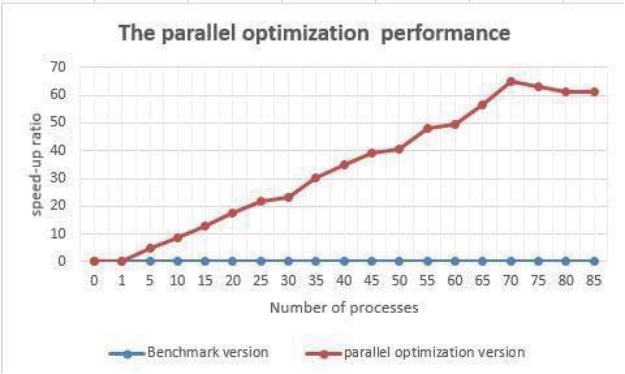

Fig. 10. Optimization performance comparison chart

TABLE 1. TEST RESULT IN GPU

	TEST	
NUMBER	TIME(S)	ACCURACY(%)
10000	0.754	0.996

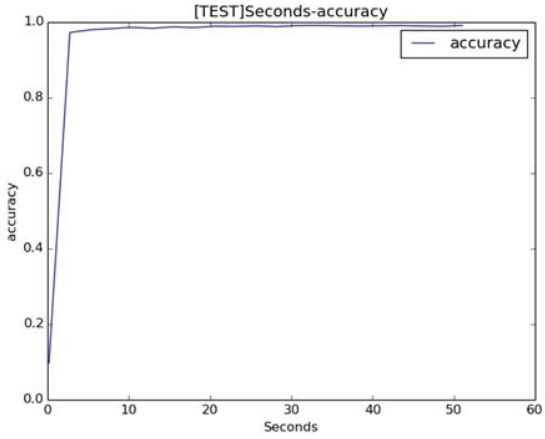

Fig. 11. Seconds-Accuracy

F. Analysis of Results

In this experiment, the acceleration ratio is measured by time. From the experimental performance comparison chart available and based on the architecture between MIC and GPU, the use of MPI multi-process parallel training samples, with the increase in the number of processes, the acceleration ratio gradually increased, while the program running time is shorter. However, with the increasednumber of processes, the process overhead will become larger. Since we use the experiment training library MNIST, the number of samples is limited. In order to ensure a certain accuracy and high performance, the best speed ratio is abtained in the 70 processes time. And in the GPU architecture, with the Caffe framework of GPU acceleration, the overall effect is more excellent.

Compared with the pre-optimization algorithm, the parallel optimized program greatly shortens the running time, and the accuracy is almost the same as the former. When the number of samples is enough, it is obviously improving the performance.

VI. CONCLUSION

In this experiment, we use MPI multi-process and OpenMP multi-thread to optimize the handwritten digit recognition algorithm of convolution neural network by using parallel technology. It does not only greatly shorten the time of CNN training samples, but also greatly improves the efficiency of convolution neural network for handwritten numeral recognition.

With the concept of big data in recent years, as well as the growing computing power of hardware, the application of convolutional neural network is more extensive. In this paper, based on parallel computing, the optimization of the handwritten numeral algorithm for convolutional neural network does not only improve the efficiency of handwritten numeral recognition, but also provides some theoretical and practical reference for other applications of convolution neural networks.

REFERENCES

[1] Openmp: An industry standard api for shared memory programming. In: IEEE Computational Science amp; Engineering (1998)

[2] Chandra, R., Dagum, L., Kohr, D., Maydan, D., Mcdonald, J., Menon, R.: Parallel programming in openmp (2001)

[3] Cire, An, D.C., Meier, U., Masci, J., Gambardella, L.M., Schmidhuber, J., rgen: Flexible, high performance convolutional neural networks for image classification. In: IJCAI 2011, Proceedings of the International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July. pp. 1237–1242 (2011)

[4] Huanfang, M.A., Zhao, X., Zou, X., Computer, S.O.: A method for convolutional neural networks training based on mapreduce framwork. Chinese Journal of Stereology Image Analysis (2015)

[5] Hui-Na, L.I., Zhou, G.B.: The research of bp neural network algorithm based on parallel. Journal of Inner Mongolia Agricultural University (2011)

[6] Kendre, S.V., Kulkarni, D.B.: Optimized convex hull with mixed (mpi and openmp) programming on hpc. International Journal of Computer Applications 1(5), 80–84 (2010)

[7] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems 25(2), 2012 (2012)

[8] Lauer, F., Suen, C.Y., Bloch, G., rard: A trainable feature extractor for handwritten digit recognition. Pattern Recognition 40(6), 1816–1824 (2007)

[9] Lin, Y.A.: Map-reduce for machine learning on multicore. Advances in Neural Information Processing Systems 19, 281–288 (2006)

[10] TOSTQ: One of the c language versions of the convolutional neural network cnn. http://blog.csdn.net/tostq/article/details/51786265 (2016)

[11] Zhou, J., Xiao, H., Wang, H., Dai, H.N.: Parallelizing simulated annealing algorithm in many integrated core architecture. Lecture Notes in Computer Science 9787, 239–250 (2016)