The Recognition of Handwritten Digits Based on BP Neural Network and the Implementation on Android

Zhu Dan[1], Chen Xu[2]

1. College of Automation, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu, 210023, China
2. Nanjing AnFeng Technology Co.Ltd., Nanjing, Jiangsu, 210023, China
chenxupro@163.com

***Abstract*—Offline handwriting recognition has become one of the hottest directions in the field of image processing and pattern recognition. It can transform any handwriting to plain text file and has been widely used in cheque recognition, mail sorting, reading aid for the blind and so on. In this paper, we attempt to recognize handwritten digits with feature extraction by Back Propagation(BP) neural network. The MNIST database of handwritten digits is applied to train and test the neural network. In addition, we introduce the Principal Component Analysis (PCA) for feature extraction which can improve the performance of the neural network and immensely shorten the training time. On the other hand, we make comparison of the recognition rate among three methods: neural network method, thirteen features method and Fisher discriminant analysis method. Finally, we succeed in porting these algorithms to Android.**

Keywords-Image Processing;PCA;Back Propagation Nerual Network;Handwitten Character Recognition;Android

I. INTRODUCTION

In recent years, handwriting recognition has become one of the hottest and challenging directions in the field of image processing and pattern recognition[1]. New technologies and methods have been proposed continously. With the development of the smart phone operation system, the application in handwriting recognition has aroused more and more attentions from researchers.

In general, handwritten character recognition is classified into two types as offline and online recognition methods[2]. Taking the need of actual application into consideration, our study only aims at offline handwritten digits recognition.

II. DATA PREPROCESSING

In this section, we introduce the methods of normalization of the MNIST database, the acquisition of handwriting images by smart phone and data preprocessing[3-4].

A. Normalization of MNIST Database

1) MNIST database

The MNIST database of handwritten digits has a training set of 60,000 examples and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image. It can be downloaded from:

http://yann.lecun.com/exdb/mnist/.

2) Binarization

Owing to the fact that the datum from MNIST database are 8-bit gray images, binarization is necessary to make images suitable for our algorithms. Binarization divides the pixels of the gray image into two classes (0 and 1) accroding to the threshold. It will be most suitable when the threshold is fixed at 120. In fact, we only need to traverse every byte of the image, divide them by threshold and restore as a new file.

3) Image normalization

Every data in MNIST database has 28*28 pixels. We need to normalize them into the same size according to the actual borders of the characters after interception. The size of 20*20 pixels is most suitable.

B. The Acquisition of Handwritten Images and Preprocessing

1) Image acquisition

The phone model we acquire is HTC Desire. The camera of this phone has 500 million pixels at most. We take photos by phone camera to acquire bmp images, intercept the images in designate area by programming and store them in RAM.

2) Grayscaling

The images gained from the former step are 32-bit color images, so we should transform them into gray images. The gray value of every pixel can be calculated by the formula belows.

$$Y = 0.299 \times \mathrm{R} + 0.587 \times \mathrm{G} + 0.114 \times \mathrm{B}$$

3) Binarization

OTSU is used to calculate threshold, which, in fact, calculate the variances among classes. The variance indicates the uniformity of the gray level distributions. The bigger the variance is, the difference between two parts of the image will be larger. The effect of OTSU is very good，while when it comes to some "unclear" images, these images will be easy to contain noises, which may bring about difficulties to character recognition. Therefore, we have to eliminate the noises later.

4) Gradient sharpening

978-0-7695-4923-1/12 $26.00 © 2012 IEEE
DOI 10.1109/ISDEA.2012.359

The process during photographing may give rise to fuzzy images which makes recognition harder[5]. Sharpening is inevitable to make images distinct. It can also restrain the noise to some extent. The Roberts gradient is used here.

5) Noise removing

Common filter methods include: median filter, mean filter. They are not suitable for dealing with long and narrow figure characters, because they are likely to eliminate the pixels of characters as well.

6) Character segmantation

We scan by rows and columns to confirm the border of the character so as to calculate the actual area the digit exists[6].

7) Image normalization

Image normalization: There are bigger differences in size among characters photographed, so we normalize them into 20*20 pixels like digits in MNIST database. The unified size can higher the correct rate of recognition.

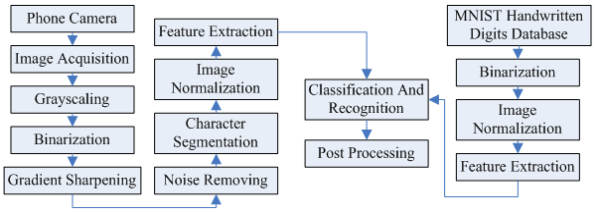

Figure 1. Flow chart of data preprocessing

Figure 2. A portion of images in MNIST database

Figure 3. Images after binarization and normalization

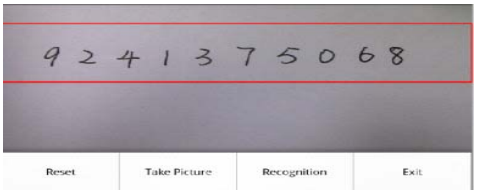

Figure 4. Image acquisition

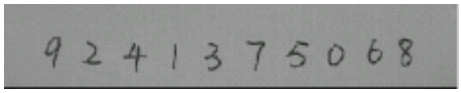

Figure 5. Grayscaling

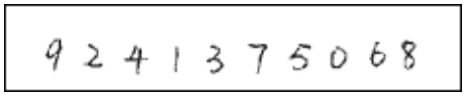

Figure 6. Binarization

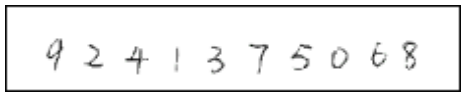

Figure 7. Gradient sharpening

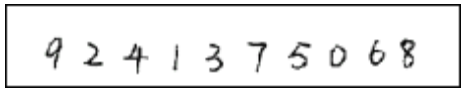

Figure 8. Noise removing

Figure 9. Image normalization

III. THE DESIGN OF BP NEURAL NETWORK

In this section, we introduce the design of BP Neural Network[7].

A. Network Layer Design

Generally, a neural network model prearranges the amount of the layers in advance. And the BP neural network may include different hidden layers. However, it has been proved in theory that the three-layer BP neural network may achieve discretional nonlinear mapping without limiting the amount of the nodes in the hidden layer. Therefore, the BP neural network model we use only includes three layers.

B. Input Layer Design

The input layer receives the external input data. Thus, the amount of the nodes lies on the dimension of the input vector. In this paper, we study the binary image which has 20*20 pixels. Therefore, the input layer includes 400 nodes.

C. Output Layer Design

The amount of the nodes in the output layer relates to the type and size of the output data. In this paper, we expect to recognize ten digits(0 to 9). The 8421 code is used to present the digits. Therefore, the output layer includes 4 nodes.

D. Hidden Layer Design

It is obvious to choose proper amount of the nodes in hidden layer. Generally, the numerous nodes may cost more time in learning process. On the contrary, the lesser nodes may result in lower recognition rate and lower fault tolerance[8]. 60 nodes in hidden layer prove to be able to obtain ideal learning speed and recognition rate after numerous experiments.

E. Target Value and Learning Rate

By testing, we conclude that when learning rate is 0.04 and the minimum mean square error is 0.02, the speed of network convergence is the fastest and the recognition rate is the highest.

F. Excitation Function

The excitation functions between input and hidden layers is Sigmoid Function which is same as that of hidden and output layers.

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

IV. Analysis of the Result of BP Neural Network

In this section, we apply the method of feature extraction based on PCA and compare the recognition rate among different algorithms[9].

A. Feature Extraction Based on PCA

Karhunen Loeve Transformation (K-L Transformation) is a kind of Principal Component Analysis (PCA) which is easy to achieve and understand. By this linear transformation, we can make use of the correlation among every dimensionality of unrecognized samples to shorten the dimensionality. And the scale of the classification machine can be minished. We use all the 60,000 examples of MNIST database as training set. The formulas of K-L Transformation are[10]:

$$\Psi = \frac{1}{M}\sum_{n=1}^{m}\Gamma_n$$
$$\Phi_i = \Gamma_i - \Psi$$
$$C = \frac{1}{M}\sum_{n=1}^{m}\Phi_n\Phi_n^T$$
$$X = \{u_1, u_2, \cdots, u_m\}^T$$
$$\Omega_i = (\Gamma_i - \Psi)\times X$$

where Γ_i is a sample of training set, Ψ is the mean of training set, C is a covariance matrix, U_i is an eigen vector of C, m is the number of samples acquired and Ω is the eigen vector of Γ_i.

Table 1. The percentage of the eigen values

Range of Eigen Values	Percentage
0	9.11%
0~1	17.16%
0~4	32.40%
0~7	42.14%
0~11	50.22%
0~19	60.52%
0~33	70.36%
0~63	79.89%
0~149	90.04%

Table 1 indicates: 0 to 63 dimensionality eigen vectors contain nearly 80 percent of original information. Therefore, the input number of nodes can be reduced at 64, which may shorten the training time and the amount of calculation.

B. The Recognition Rate of Different Algorithms

Not only gaining the BP neural network and its best recognition rate, we also compare it with other two methods: thirteen features method and Fisher discriminant analysis method. These methods take the same 60,000 training samples and 1,000 testing samples as BP neural network.

The parameters of each algorithm:

1) BP neural network

The number of nodes is 400 in input layer, 60 in hidden layer and 4 in output layer. The learning rate is 0.04. The minimum mean squared error is 0.02.

2) PCA+BP neural network

The number of nodes is 64 in input layer, 35 in hidden layer and 4 in output layer. The learning rate is 0.04. The minimum mean squared error is 0.02.

3) Thirteen features

The dimensionality of eigen vectors is 13.

4) Fisher discriminant analysis

The dimensionality of eigen vectors is 80.

Table 2. The recognition rate of different digits by different methods

Digits/ Method	Fisher	Thirteen Feature	PCA+BP	BP
0 (%)	95.83%	98.82%	95.29%	95.29%
1 (%)	60.98%	96.03%	92.06%	93.65%
2 (%)	59.38%	86.21%	97.41%	91.38%
3 (%)	66.67%	82.24%	90.65%	96.26%
4 (%)	91.89%	81.82%	85.45%	87.27%
5 (%)	89.66%	80.46%	88.51%	81.61%
6 (%)	87.50%	96.55%	91.95%	90.80%
7 (%)	76.47%	83.84%	90.91%	79.80%
8 (%)	76.19%	76.40%	87.64%	76.40%
9 (%)	79.41%	81.91%	91.49%	87.23%
Over all (%)	77.67%	86.50%	91.20%	88.30%

Table 2 indicates: With the same training set and testing set: The overall recognition rate of each algorithm is: Fisher<Thirteen Feature<BP<PCA+BP; The recognition rate of eight is very low; The digit 8 is easily mistaken as others, so we can use its architectural feature like hole number to increase the rate; The recognition rates of PCA+BP neural network and BP neural network are nearly the same; The training time and calculation amount of BP neural network are immensely reduced.

V. The Implement of Different Algorithms in Android

In this section, we succeed in implement the algorithms mentioned above in Android.

A. Android NDK

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The algorithms of image processing and neural

network applied in the software are implemented by C and C++, which can not directly run at Android. By Java, the amount of work is huge and the efficiency is not high. So, it is necessary to use Android NDK. The NDK toolkit itself provided by Google do not support C++ exception, RTTI and standard C++ library, but some of our algorithms are developed by C++. CrystaX Android NDK toolkit helps to solve the problem. This toolkit can be downloaded from:

http://www.crystax.net/android/ndk.php.

In this paper, we adopt android-ndk-r4-crystax which is modified on the basis of NDK r4. Notice that we should add: LOCAL_CPP_EXTENSION: = .cpp to Android. mk for compiler to recognize the .cpp C++ source file.

B. Generation of Dynamic Link Library

We compile partly code of the image preprocessing into libSegmentation.so and partly code of pattern recognition into libPatternRecognition.so.

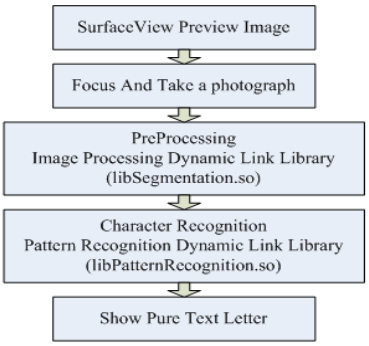

Figure 10. Flow chart of software

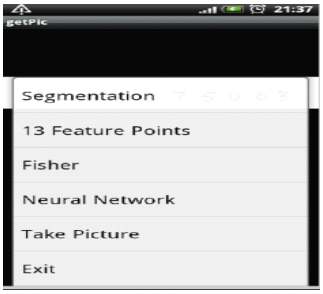

Figure 11. Recognition result

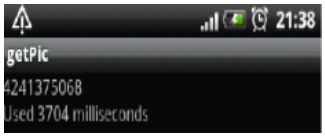

Figure 12. Recognition result

VI. CONCLUSIONS

In this paper, BP neural network based on offline handwritten digits recognition system has been introduced. We also implement BP neural network, PCA+BP neural network, thirteen features and Fisher discriminant analysis methods in Android successfully and compare the recognition rates of them.

- After comparisons, Fisher discriminant analysis methods has the lowest recognition rate and proved to be time-consuming which is not suitable for smart phone application as is showed in table II.
- BP neural network itself has too many nodes, which costs a lot of time in training. We should train them beforehand, and then export the weight of every layer for recognition.
- The thirteen features method need to calculate the minimum of the weighted distance from unclassified sample to all the training samples, so it occupies much more RAM.
- In the actual testing of smart phone software, we respectively recognized 1,000 handwritten digits from 10 writers. The overall recognition rate is about 75 percent, which accord with the demand and have definite practical applicability.
- In order to increase the recognition rate, we acquire the real-time-learning method to adapt to the written style of the specific software user.

Eventually, for better learning and communicating, we upload all the source code in this paper to:

http://sourceforge.net/projects/andocrrecog/files/ .

REFERENCES

[1] J. Pradeep, E. Srinivasan, S. Himavathi. Neural Network based handwritten character recognition system without feature extraction[C], International Conference on Computer, Communication and Electrical Technology－ICCCET 2011, 18th & 19th, March 2011.

[2] W. Wu, Y. Bao. Online handwriting Mongolia words recognition based on multiple classifiers[C], 2009.

[3] G. Sun, Y. Li, P. Wang, J. Yang. An adaptive normalization method for handwritten character recognition based on Neural Networks[J], Pattern Recognition and Artificial Intelligence, vol. 3, 2005.

[4] L. Gao. Handwritten English letters recognition based on BP Neural Network[D], April 2009.

[5] W. Feng, M. Tang, B. He. Digital image pattern recognition programming using Visual C++, China Machine Press[M], September 2010.

[6] L. Li, G. Feng. The license plate recognition system based on fuzzy theory and BP Neural Network[C], 2011 Fourth International Conference on Intelligent Computation Technology and Automation, 2011.

[7] Y. Wang. An improved handwritten numeral recognition method based on the BP Neural Network[J], Computer Engineering & Science, vol. 4, 2008.

[8] Y. Fan, Y. Ma, C. Xue. Tank Unit Combat Formation Recognition Model Based on BP Neural Network in Virtual Simulation Training[C], 2011 Fourth International Conference on Intelligent Computation Technology and Automation, 2011.

[9] Y. Wang. Research on key technologies in handwritten numeral recognition[D], BUPT, 2009.

[10] Y. Yang. An artificial neural networks based OCR system and its hardware implementation[D], April 2006.