NEURAL NETWORK SIGNAL PROCESSING FOR HANDWRITTEN CHARACTER RECOGNITION

MR.Abbas akram Khorseehd
Al mustansirya university-college of science

ABSTRACT

A simulator software program that implements the neural network multi layer perceptron trained by the Back propagation algorithm is presented. The simulator is used to learn the ten numerals handwritten characters. A database containing 1000 patterns written by 1000 different people is collected. The percentage error of the training set is 1.3% and the percentage error of the test set is 7.3%.

KEY WORDS

Signal Processing, Simulation, Character Recognition, Neural Networks.

1. Introduction

The neural network multi layer perceptron [1], [2], [3] trained by the Back propagation algorithm [3], [4], [5], [6] program for the ten numerals handwritten character recognition is presented. A simulator software program that implements the neural network has been designed. The simulator is used to learn the ten numerals handwritten characters.

2. Ten handwritten numbers database

A database containing 1000 patterns written by 1000 different people is collected. Figure 1 shows a pattern of the different 10 symbols. The database contains 1000 of each pattern. First, the people have written them on papers, and then the acquisition phase has been used to scan them and prepare them for computers.

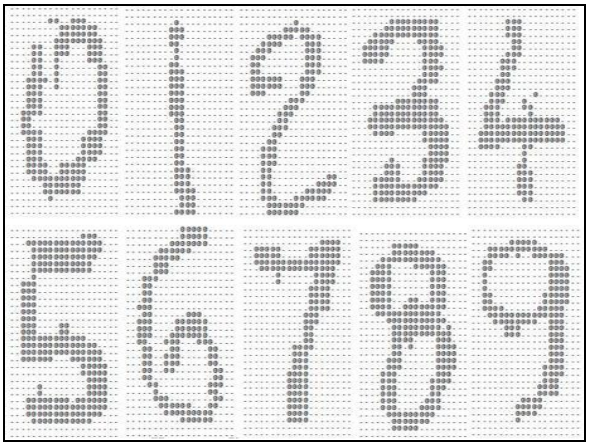

Figure 1: The ten symbols handwritten 0,1,2,3,4,5,6,7,8,9.

The ten numerals are coded by 32*20 digital pixel matrixes. Figure 2 shows an example of number eight. Figure 2 includes only 0s and 1s, where 0s are represented by "-" and 1s are represented by "@".

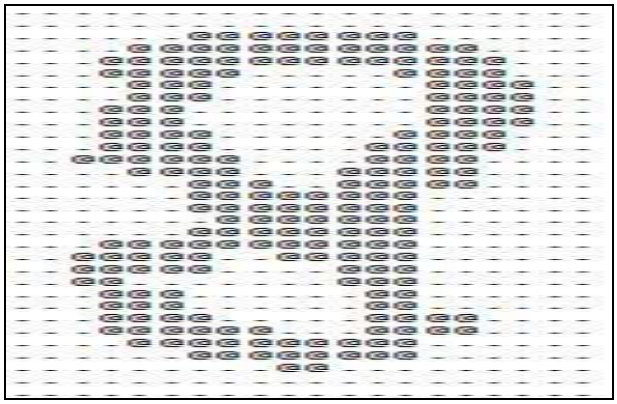

Figure 2: 32*20 digital pixel matrixes.

3. The pre-processing operations

The ten numerals (See Figure 1 and Figure 2), which are coded by 32*20 digital pixel matrixes, are transformed, through a pre-processing phase, into 28 analog values in the range [-1:1], as it follows:

1) The first and last two columns have been cut because they are always 0s;
2) The first and last two rows have been cut because they are always 0s;
3) The new figure will be composed of 28*16 digital pixel matrixes (32-4*20-4);
4) Divide the matrix into sub-matrixes; each one is composed of 4*4=16 pixels (See Figure 3);

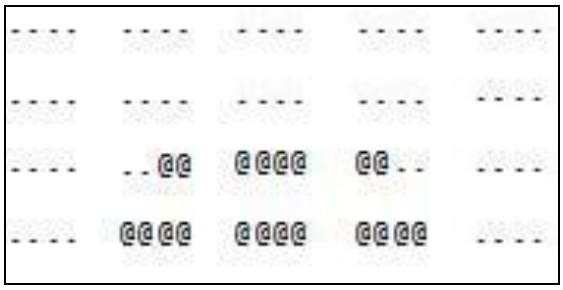

Figure 3: Sub-matrixes

5) The number of sub-matrixes equals to 28 because 28/4*16/4 = 7*4;
6) Count the 1s in each sub-matrix and compute the ratio Count of 1s divide by 16 (See Figure 3):

First Sub-matrix	0/16=0.000
Second Sub-matrix	6/16=0.375
Third Sub-matrix	8/16 =0.500
Fourth Sub-matrix	6/16 =0.375
Fifth Sub-matrix	0/16=0.000

7) The following final vector will be obtained:
   Example1:
   28 Input = -0.000 0.375 0.500 0.375 0.000 0.375 -0.500 0.000 -1.000 0.375 0.625 -0.250 -1.000 -0.750 0.875 -0.750 -1.000 -0.250 0.750 -0.500 -0.750 0.375 -0.250 -0.250 -1.000 0.250 0.750 -0.375
   Example2:
   28 Input = -1.000 -0.125 0.875 -0.625 -1.000 0.375 -0.500 0.000 -1.000 0.375 0.625 -0.250 -1.000 -0.750 0.875 -0.750 -1.000 -0.250 0.750 -0.500 -0.750 0.375 -0.250 -0.250 -1.000 0.250 0.750 -0.375

Because the problem contains ten numerals, the neural network must classify all the ten number. For that, the output of the neural network must be as follows:

0	-1 -1 -1 -1 -1 -1 -1 -1 -1 **1**
1	-1 -1 -1 -1 -1 -1 -1 -1 **1** -1
2	-1 -1 -1 -1 -1 -1 -1 **1** -1 -1
3	-1 -1 -1 -1 -1 -1 **1** -1 -1 -1
4	-1 -1 -1 -1 -1 **1** -1 -1 -1 -1
5	-1 -1 -1 -1 **1** -1 -1 -1 -1 -1
6	-1 -1 -1 **1** -1 -1 -1 -1 -1 -1
7	-1 -1 **1** -1 -1 -1 -1 -1 -1 -1
8	-1 **1** -1 -1 -1 -1 -1 -1 -1 -1
9	**1** -1 -1 -1 -1 -1 -1 -1 -1 -1

See Table 1 for details at the end of this paper.

4. The Simulation Results

We have produced a neural simulator that implements the MLP trained by the BP, in order to solve the presented problem (ten numerals).

First of all, the database is divided in three sets, a training set (500 patterns), a test set (300 patterns), a cross validation set (200 patterns). Where,

1) the training set is the set that is used to train the network;
2) the test set is the set that is used to test the generality of the network, by classifying a pattern that it is not been presented to it during the learning phase;

3) The cross validation set is the set that is used to test the generalization of the network at the over fitting point "OFP", where the number of the hidden neuron is optimum.

The neural network topology must be two-layers, as it follows:

1) The inputs number is 28 inputs because the input vector in the pre-processing phase was 28;
2) The outputs number is 10 because the ten numerals test problem and then 10 classes;
3) The hidden neurons number is unknown. It must be determined by using the simulation;
4) the optimum pattern number of the training set that can be used to obtain small test set error.

Using the developed neural simulator, We simulated the algorithm using as test problem the character recognition (ten numerals). The following results have been obtained:

1) Figure 4 shows that the optimum hidden number is 50 hidden neurons and the optimum pattern number = the maximum which is 500 patterns.
2) Figure 5 shows that the percentage error of the training set is 1.3% and the percentage error of the test set is 7.3%
3) The percentage error of the cross validation set is 9.4%.
4) Figure 6 shows that the Weight distribution, which is normal distribution.

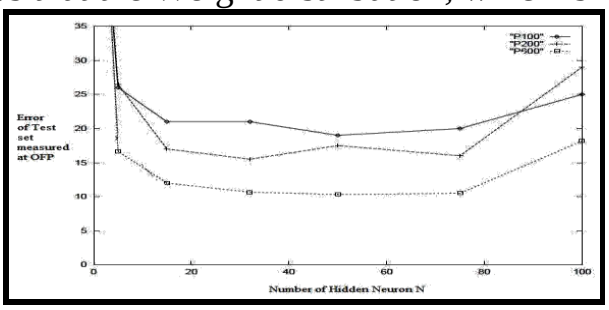

Figure 4:The percentage test set error versus the hidden number.

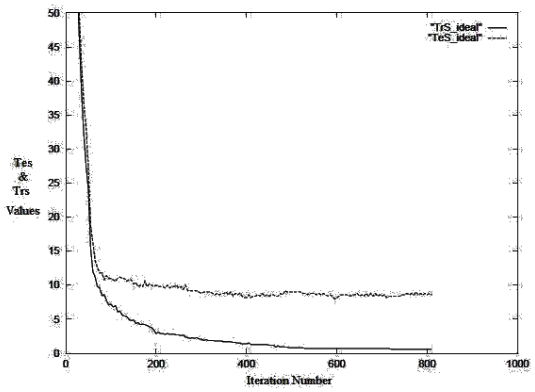

Figure 5:Behaviour of the error for the training set.

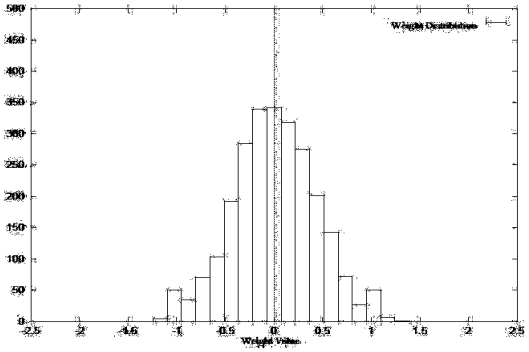

Figure 6: Weight distribution.

5. Conclusion

In this paper, a simulator software program that implements the neural network multi layer perceptron trained by the Back propagation algorithm is presented. The simulator is used to learn the ten numerals handwritten characters. The database containing 1000 patterns written by 1000 different people is collected. The ten numerals handwritten characters have been processed from paper to matrix to vector. The percentage error of the training set is 1.3% and the percentage error of the test set is 7.3%.

Table 1: The ten handwritten numbers after the preprocessing operations 28 inputs and the 10 classifications output.

	28 INPUT	10 OUTPUT
0	-1.000 -0.250 0.875 -0.125 0.000 -0.250 -0.750 0.375 0.375 -0.500 -1.000 0.000 0.500 -1.000 -1.000 0.000 0.500 -1.000 -0.750 0.500 0.500 -0.500 0.375 -0.375 -0.750 0.625 0.250 -1.000	-1 -1 -1 -1 -1 -1 -1 -1 -1 [1]
1	-1.000 -1.000 0.250 -1.000 -1.000 -1.000 0.250 -1.000 -1.000 -0.625 0.875 -1.000 -1.000 0.125 0.875 -0.750 -0.750 -0.500 0.375 -0.500 -1.000 -1.000 0.500 -1.000 -1.000 -1.000 0.375 -0.750	-1 -1 -1 -1 -1 -1 -1 -1 [1] -1
2	-0.750 0.375 0.625 -1.000 -0.625 -0.125 -0.125 -0.250 0.625 -0.375 0.125 -0.750 -0.250-0.250- 0.375 -1.000 -1.000 0.500 -0.625 -1.000 -0.375 -0.125 -1.000 -1.000 0.750 0.875 0.875 0.750	-1 -1 -1 -1 -1 -1 -1 [1] -1 -1
3	-0.750 0.500 0.750 -0.875 -1.000 -1.000 0.250 -1.000 -1.000 -0.375 1.000 -0.750 -0.750 0.750 0.500 0.875 -1.000 -1.000 -1.000 0.500 -1.000 -1.000 -1.000 0.625 0.500 0.875 0.250 -0.625	-1 -1 -1 -1 -1 -1 [1] -1 -1 -1
4	-1.000 0.250 -0.125 -1.000 -0.875 0.875 0.000 -1.000 -0.250 1.000 0.000 -1.000 0.875 -0.500 0.000 -0.750 0.875 0.250 0.750 -0.125 0.000 0.500 0.625 -1.000 -1.000 -1.000 0.375 -1.000	-1 -1 -1 -1 -1 [1] -1 -1 -1 -1
5	0.000 1.000 0.875 0.625 -0.375 -0.500 -0.500 -0.250 0.750 -0.750 -1.000 -1.000 0.250 0.750 0.625 -0.750 -1.000 -1.000 0.000 0.250 -1.000 0.000 -0.750 -0.250 -1.000 0.000 0.875 -0.500	-1 -1 -1 -1 [1] -1 -1 -1 -1 -1
6	-1.000 0.000 0.375 -0.750 -1.000 0.250 -1.000 -1.000 -0.500 0.125 -1.000 -1.000 0.125 -0.375 -1.000 -1.000 0.500 -0.750 0.000 -0.375 0.750 0.250 -1.000 0.250 0.375 -0.125 -0.375 0.375	-1 -1 -1 [1] -1 -1 -1 -1 -1 -1
7	-1.000 -1.000 -0.375 0.375 -0.125 0.500 0.750 -0.875 -1.000 -0.750 -0.250 -1.000 -1.000 -0.375 0.250 -0.625 -0.125 0.875 0.750 0.875 -1.000 -0.375 -0.875 -1.000 -1.000 -0.250 -0.750 -1.000	-1 -1 [1] -1 -1 -1 -1 -1 -1 -1
8	-1.000 -0.125 0.875 -0.625 -1.000 0.375 -0.500 0.000 -1.000 0.375 0.625 -0.250 -1.000 -0.750 0.875 -0.750 -1.000 -0.250 0.750 -0.500 -0.750 0.375 -0.250 -0.250 -1.000 0.250 0.750 -0.375	-1 [1] -1 -1 -1 -1 -1 -1 -1 -1
9	-0.125 -0.375 0.875 -0.625 0.875 -0.750 1.000 -0.500 0.625 -0.500 0.875 -0.500 0.250 0.500 0.625 -0.250 -1.000 -1.000 -1.000 0.750 -1.000 -1.000 -1.000 0.875 -0.750 0.625 0.500 0.000	[1] -1 -1 -1 -1 -1 -1 -1 -1 -1

References

[1] D.E. Rumelhart and J.L. McClelland. Parallel distribuited processing. MIT Press, Cambridge, Mass., 1986.

[2] H. Chibl_e, M. Valle and D.D. Caviglia. A modi_ed vogl-based back propagation algorithm for VLSI hardware implementation. Proccedings of the _fteenth IASTED International Conference on Applied Informatics, pp. 83{86, 1997.

[3] J. Hertz, A. Kough and R.G. Palmer. Introduction to the Theory of Neural Computation. Addison-Wesley, Reading, 1991.

[4] M. Valle, D.D. Caviglia, G. Donzellini, A. Mussi, Oddone F. and G.M. Bisio. A neural computer based on an analog VLSI neural network. Proceedings of ICANN'94, Sorrento, Italy, vol. 2:pp. 1339{1342, 26- 29 May 1994.

[5] R. P. Lippmann. An introduction to computing with neural nets. IEEE ASSP Magazine, pp. 4{22, April, 1987.

[6] T. P. Vogl, J. K. Mangis, A. K. Rigler, W.T. Zink and D.L. Alkon. Accelerating the convergence of the back-propagation method. Biological Cybernetics, pp. 257{263, 1988.

الخلاصه

يقدم هذا البحث برنامج تطبيقي للمحاكاة مستخدما الشبكه العصبية متعددة الطبقات والمدربة بواسطة خوارزمية التغذيه الارجاعيه "العكسيه".عملية المحاكاة perceptron استخدمت لتعليم عشرة ارقام كتبت بخط اليد. قاعدة البيانات تحتوي على 1000 نموذج كتبت من قبل 1000 شخص مختلفين.نسبة الخطأ التي ظهرت اثناء فترة التعلم هي 1.3% ونسبة الخطأ اثناء التجربه هو 7.3% .