Improved Method of Handwritten Digit Recognition

Ernst Kussul, Tatyana Baidyk
UNAM, Centro de Instrumentos,
Cd.Universitaria A.P.70-186 C.P.04510 Mexico, D.F.

ekussul@servidor.unam.mx
tbaidyk@aleph.cinstrum.unam.mx

Abstract

MNIST database serves for comparison of different methods of handwritten digit recognition. There are many data related to different classifier recognition rates among which our neural classifier had the second place [1] (recognition rate 99.21%). At present we develop improvements of neural network structure and algorithms of handwritten digit recognition. Improved classifier has recognition rate 99.37%. This result is the best from the known ones. In this paper we briefly describe the general structure of our classifier and the latest improvements.

1 Introduction

There are many applications, which need to recognize handwritten digits. For example, bank checks, custom declaration automatic reading etc.

Various methods were proposed to solve this problem [2], [3]. For estimation of the method effectiveness the most important parameter is recognition rate. This parameter show, which proportion of samples in test database is recognized correctly.

The MNIST database contains 60,000 handwritten digits in the training set and 10,000 handwritten digits in the test set. Different classifiers proved on this database by LeCun [2] had shown recognition rate from 88% till 99.3% (Table 1).

We have developed new neural classifier LIRA (LImited Receptive Area classifier) based on Rosenblatt's perceptron principles. To adapt Rosenblatt's perceptron for handwritten digit recognition problem we made some changes in perceptron structure, training and recognition algorithms.

Rosenblatt's perceptron contains three layers of neurons. The first layer *S* corresponds to retina. In technical terms it corresponds to input image. The second layer *A* called the associative layer corresponds to feature extraction subsystem. The third layer *R* corresponds to output of all the system. Each neuron of this layer corresponds to one of the output classes. In handwritten digit recognition task this layer contains 10 neurons corresponding to digits 0, ..., 9. Connections between the layers *S* and *A* are established using a random procedure and cannot be changed by perceptron training. They have the weights 0 or 1.

Table 1: Recognition rate of different classifiers

METHODS	% OF ERROR NUMBER
Linear classifier	12.0
Linear classifier (nearest neighbor-NN)	8.4
Pairwise linear classifier	7.6
K-NN, Euclidean	5.0
2-layer NN, 300 hidden units (HU) (28x28-300-10)	4.7
2-layer NN, 1000 HU (28x28-1000-10)	4.5
2-layer NN, 1000 HU, [distortions] (28x28-1000-10)	3.8
2-layer NN, 300 HU, [distortions] (28x28-300-10)	3.6
1000 RBF (Radial Basis Function) + linear classifier	3.6
40 PCA (Principal Component Analysis) + quadratic classifier	3.3
3-layer NN, 300+100 HU (28x28-300-100-10)	3.05
3-layer NN, 500+150 HU (28x28-500-150-10)	2.95
3-layer NN, 300+100 HU, [distortions] (28x28-300-100-10)	2.5
3-layer NN, 500+150 HU, [distortions] (28x28-500-150-10)	2.45
K-NN Euclidean, deslant	2.4
LeNet-1 [16x16]	1.7
2-layer NN, 300 HU, [deslant] (20x20-300-10)	1.6
K-NN, Tangent Distance, [16x16]	1.1
SVM (Support Vector Machine) poly 4	1.1
LeNet-4	1.1
LeNet-4 / K-NN	1.1
LeNet-4 / Local	1.1
Reduced Set SVM poly 5	1.0
LeNet-5	0.95
Virtual SVM poly 9 [distortions]	0.8
LeNet-5 [distortions]	0.8
Boosted LeNet-4 [distortions]	0.7
Proposed classifier LIRA	**0.63**

Connections between layers A and R are established by the principle when each neuron of A-layer is connected with all neurons of R-layer. Initially the weights are set to 0. The weights are changed during the perceptron training. The rule of weights changing corresponds to the training algorithm. We used a training algorithm slightly different from the Rosenblatt's one. We have also modified the random procedure of S-connections establishment. Our latest modifications are related to the rule of winner selection in the output R-layer.

In this paper we describe our approach and the handwritten digit recognition results.

2 Rosenblatt perceptrons

3-layer Rosenblatt perceptron contains sensor layer S, associative layer A and reaction layer R. Many investigations were dedicated to perceptrons with one neuron in layer R (R-layer) [4]. Such perceptron can recognize only two classes. If output of R neuron is higher than predetermined threshold T, the input image belongs to class 1. If it is lower than T the input image belongs to class 2. The sensor layer S (S-layer) contains two-state {–1, 1} elements. The element is set to 1 if it belongs to object image and set to –1, if it belongs to background.

Associative layer A (A-layer) contains neurons with 2-state {0, 1} outputs. Inputs of these neurons are connected with outputs of S-layer neurons with no modifiable connections. Each connection may have the weight 1 (positive connection); or the weight –1 (negative connection). Let the threshold of such neuron equals to number of its input connections. This neuron is active only in the case if all positive connections correspond to the object and negative connections correspond to background.

The neuron R is connected with all neurons of A-layer. The weights of these connections are changed during the perceptron training. The most popular training rule is increasing the weights between active neurons of A-layer and neuron R if the object belongs to class 1. If the object belongs to the class 2 corresponding weights are decreasing. It is known that such perceptron has fast convergence and can form nonlinear discriminating surfaces. The complexity of discriminating surface depends on the number of A-layer neurons.

3 Description of the Rosenblatt perceptron modifications

We have proposed several changes to perceptron structure to create the neural classifiers for handwritten digit recognition. To examine them we used MNIST database [2]. Each black and white digit image is presented by 20*20 pixel box. The image was converted to gray level and was centered in 28*28 image by computing the center of mass of the pixels, and translating the image so as to position this point at the center of the 28*28 field. In our case we worked with the binary image.

Binary image is obtained from gray-level image by the following procedure. The threshold th is computed as:

$$th = 2 * (\sum_{i=1}^{W_S} \sum_{j=1}^{H_S} b_{ij}), \quad (1)$$

where H_S – the number of rows of the image; W_S – the number of columns of the image; b_{ij} – brightness of the pixel of gray-scale image; s_{ij} – brightness of the pixel of the resulting binary image:

$$s_{ij} = \begin{cases} 1, \text{if } b_{ij} > th, \\ -1, \text{if } b_{ij} \le th. \end{cases} \quad (2)$$

For the MNIST database $H_S = W_S = 28$.

For the first modification of simple Rosenblatt perceptron ten neurons were included into R-layer. In this case it is necessary to introduce the rule of winner selection. In the first series of experiments we used the simplest rule of winner selection. The neuron from R-layer having the highest excitation determines the class under recognition. Using this rule we obtained recognition rate 99.21%.

After that we modified winner selection rule and improved recognition rate to 99.37%. We'll describe this selection rule later.

The second modification was made in the training process. Let the neuron-winner has excitation E_w, its nearest competitor has excitation E_c. If

$$(E_w - E_c) / E_w < T_E \quad (3)$$

the competitor is considered as a winner, where T_E is the superfluous excitation of the neuron-winner.

The third modification is concerned with connections. The connections between A-layer and R-layer of Rosenblatt perceptron could be negative and positive. We used only positive connections. In this case training procedure is the following: during recognition process we obtain excitations of R-layer neurons. The excitation of neuron R_j corresponding to correct class is decreased by the factor (1 – T_E). After this the neuron having maximum excitation R_k is selected as winner.

If $j = k$, nothing to be done.

If j does not equal k,

$$w_{ij}(t + 1) = w_{ij}(t) + a_i, \quad (4)$$

where $w_{ij}(t)$ is the weight of connection between i-neuron of A-layer and j-neuron of R-layer before reinforcement, $w_{ij}(t + 1)$ is the weight after reinforcement, a_i is the output signal (0 or 1) of i-neuron of A-layer.

$$\begin{aligned} w_{ik}(t+1) &= w_{ik}(t) - a_i, && \text{if } (w_{ik}(t) > 0), \\ w_{ik}(t+1) &= 0, && \text{if } (w_{ik}(t) = 0), \end{aligned} \quad (5)$$

where $w_{ik}(t)$ is the weight of connection between i-neuron of A-layer and k-neuron of R-layer before reinforcement, $w_{ik}(t + 1)$ is the weight after reinforcement. More detailed description of the training procedure will be done further.

The perceptron with these changes is termed the LImited Receptive Area classifier (LIRA) (Figure 1). More general case of such classifier was developed and named Random Subspace Classifier (RSC) [5] – [7].

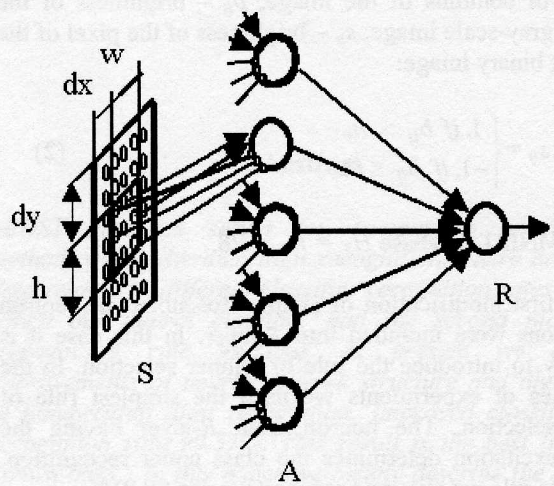

Figure 1: LImited Receptive Area classifier (LIRA)

Each A-layer neuron of LIRA has random connections with S-layer. To install these connections it is necessary to enumerate all elements of S-layer. Let the number of these elements equals to N_S. To determine the connection of the A-layer neuron we select the random number uniformly distributed in the range $[1, N_S]$. This number determines S-layer neuron, which will be connected with the mentioned A-layer neuron. The same rule is used to determine all connections between A-layer neurons and S-layer neurons. Frank Rosenblatt proposed this rule [4]. Our experience shows that it is possible to improve the perceptron performance by modification of this rule.

The fourth modification is the following. We connect A-layer neuron with S-layer neurons randomly selected not from the entire S-layer, but rather from the rectangle $(h * w)$, which is located in S-layer (Figure 1).

The distances dx and dy are random numbers selected from the ranges: dx from $[0, W_S - w]$ and dy from $[0, H_S - h]$, where W_S, H_S stand for width and height of S-layer.

3.1. Mask design

The associative neuron mask is the number of positive and negative connections of A-layer neuron with retina. The procedure of random selection of connections is used to design the mask. This procedure begins from choice of the upper left corner of the rectangle in which all positive and negative connections of the associative neuron are located. The next formulas are used:

$$dx_i = random_i(W_S - w),$$
$$dy_i = random_i(H_S - h), \quad (6)$$

where i – the position of a neuron in associative layer A; $random_i\,(z)$ – the random number which is uniformly distributed in the range $[0, z]$. After that the each positive and negative connection position within the rectangle is defined by couple of numbers:

$$x_{ij} = random_{ij}(w),$$
$$y_{ij} = random_{ij}(h), \quad (7)$$

where j – the number of i-th neuron connection with retina. Absolute coordinates of the connection on the retina are defined by couple of the numbers:

$$X_{ij} = x_{ij} + dx_i,$$
$$Y_{ij} = y_{ij} + dy_i., \quad (8)$$

3.2. Image coding

Any input image defines the activities of A-layer neurons in one-to-one correspondence. The binary vector which corresponds to the activity of associative neurons is termed the image binary code $A= a_1, \ldots, a_n$, (where n – the number of the neurons in A-layer). The procedure, which transforms the input image to binary vector A, is termed the image coding.

In our system i-th neuron of A-layer is active only if all the positive connections with retina correspond to object and all negative connections correspond to the background. In this case $a_i = 1$, in opposite case $a_i = 0$. From the experience of the work with such systems it is known that the active neuron number m in A-layer must be many times less than whole neuron number n of this layer. In our works we usually use next expression $m = c\sqrt{n}$, where c – constant, which belongs to the range from 1 till 5. This relation corresponds to neurophysiological facts. The number of active neurons in the cerebral cortex is hundreds times less than the total number of neurons.

Taking into account the little number of active neurons it is convenient to represent the binary vector A not explicitly but as a list of numbers of active neurons. Let, for example, the vector A is:

$$A = 00010000100000010000.$$

The corresponding list of the numbers of active neurons will be 4, 9, and 16. This list is used for save the image codes in compact form and for the fast calculation of activity of the neurons of output layer. Thus after execution of the coding procedure every image has corresponding list of numbers of active neurons.

3.3. Training procedure

Before training all the weights of connections between neurons of A-layer and R-layer are set to zero.

1. The training procedure begins from the presentation of the first image to the perceptron. The image is coded and the R-layer neuron excitation E_i is computed. E_i is defined as:

$$E_i = \sum_{j=1}^{n} a_j * w_{ji} \qquad (9)$$

where E_i – the excitation of the i-th neuron of the R-layer; a_j - the excitation of the j-th neuron of A-layer; w_{ji} – weight of the connection between j-th neuron of A-layer and i-th neuron of R-layer.

2. We require recognition to be robust. After calculation of the all neuron excitations of R-layer the correct name of presented image is read from mark file of the MNIST database. The excitation E of corresponding neuron is recalculated according to the formula:

$$E_k^* = E_k * (1 - T_E). \qquad (10)$$

After that we find the neuron (winner) with the maximal activity. This neuron presents the recognized handwritten digit.

3. Denote the neuron-winner number as i_w, and the number of neuron, which really corresponds to the input image, as i_c. If $i_w = i_c$ nothing to be done. If $i_w \neq i_c$

$$\begin{aligned} &(\forall j)\left(w_{ji_c}(t+1) = w_{ji_c}(t) + a_j\right) \\ &(\forall j)\left(w_{ji_w}(t+1) = w_{ji_w}(t) - a_j\right) \qquad (11) \\ &\text{if } (w_{ji_w}(t+1) < 0) \;\; w_{ji_w}(t+1) = 0. \end{aligned}$$

where $w_{ji}(t)$ is the weight of connection between j-neuron of A-layer and i-neuron of R-layer before reinforcement, $w_{ji}(t+1)$ is the weight after reinforcement.

The training process is carried out iteratively. After representation of all the images from training subset the total number of training errors is calculated. If this number is higher than one percent of total number of images then the next training cycle is doing. If the error number is less than one percent the training process is stopped. The training process is also stopped when the cycle number is more than before prescribed value. In previous experiments this value was 10 cycles, and in final ones – 40 cycles.

It is obvious that in every new training cycle the image coding procedure is repeated and gives the same results as in previous cycles. Therefore in final experiments we performed the coding process of images only once and recorded the lists of active neuron numbers for each image on hard drive. Later for all cycles we used not the images but corresponding lists of active neurons. Due to this procedure training process was accelerated approximately for an order of magnitude.

It is known [2], that recognition rate of handwritten symbols may be increased essentially if during the training cycle represent the images not only in initial state but also with shifting and with changing the image inclination (so called distortions). In final experiments we used besides the initial images 16 variants of each image with distortions.

Distortion models can be used to increase the effective size of a data set without actually requiring collecting more data. We used 16 distortion variants (Table 2): 12 shifts and 4 skewing.

Table 2: Input image distortions (shifts)

X	–1	0	1	0	–1	–1	1	1	–2	0	2	0
Y	0	–1	0	1	–1	1	–1	1	0	–2	0	2

The skewing angles were selected –26°, –13°, 13° and 26°.

3.4. Recognition procedure

To examine the recognition rate the test set of MNIST database was used. This test set contains 10000 images. Coding and calculation of neuron activity were made by the same rules as by training, but the value T_E (reserve of robustness) was 0.

The recognition process for the new classifier differs from the previous ones. In this version we use distortions in recognition process too. There is the difference between implementation of distortions during the training session and recognition session. In the training session each new position of initial image produced by distortions is considered as a new image, which is independent from other image distortions. In recognition session it is necessary to introduce a rule of decision-making. All the recognition results of one image and its distortions must be used for receiving of one result, which gives the class name of image under recognition. We have developed two rules of decision-making.

Rule 1. According to this rule all the excitations of R-layer neurons are sum for all the distortions.

$$E_i = \sum_{k=1}^{d} \sum_{j=1}^{n} a_{kj} * w_{ji}\,, \qquad (12)$$

where E_i – the excitation of the i-th neuron of the R-layer; a_{kj} – the excitation of the j-th neuron of A-layer in k-th distortion; w_{ji} – weight of the connection between j-th neuron of A-layer and i-th neuron of R-layer. And after that the neuron-winner is selected as result of recognition.

Rule 2. The second rule consists in calculations of R-layer neurons excitations and selection of neuron-winner and its nearest competitor for each distortion. For k-th distortion the relation r_k of the neuron-winner excitation E_{wk} to its nearest competitor excitation E_{ck} is calculated.

$$r_k = \frac{E_{wk}}{Eck}. \qquad (13)$$

After that we select distortion with the maximal r_k. The neuron-winner of this distortion is considered to be the result of recognition.

4 Handwritten digit recognition results

We carried out preliminary experiments to estimate the performance of our classifiers. On the base of preliminary experiments we selected the best classifiers and carried out final experiments to obtain maximal recognition rate. In the preliminary experiments we changed the A-layer neuron number from 1 000 to 128 000 (Table 3). These experiments showed that recognition error number has been decreased approximately by the factor 8 with increasing of A-layer neuron number. Disadvantages of very big A-layer are the increasing of train and recognition time and memory capacity.

We also changed the ratio $p = w/W_S = h/H_S$ from 0.2 to 0.8. The parameter T_E was 0.1. In these experiments we did not use distortions in either training or recognition sessions.

For each set of parameters we made 10 training cycles on MNIST training set. After that we estimated the recognition rate on MNIST test set. The recognition rates obtained in the preliminary experiments are presented in Table 3.

In the preliminary experiments we created 3 positive and 3 negative connections for each A-layer neuron. In the final experiments we created 3 positive and 5 negative connections (Table 4). The number of A-layer neurons was 256000. Windows parameters were w=10 and h=10 and retina size was 28x28. Number of training cycles was 40.

Coding time was 20 hours and training time 45 hours. Recognition time (for 10 000 samples) was 30 minutes without distortions, 60 minutes for 4 distortions and 120 minutes for 8 distortions. We made different experiments with different numbers of distortions in recognition session (4 and 8). We created distortions only with shifting (the first four or eight cases in the Table 2). For comparison we made experiments without distortions in recognition session.

Table 3: The recognition rates of classifier in preliminary experiments

A-layer neuron number	Error number				
	p = 0.2	p=0.4	p=0.6	p=0.8	p=1
1000	3461	1333	1297	1355	1864
2000	1705	772	772	827	1027
4000	828	452	491	532	622
8000	482	338	335	388	451
16000	330	249	247	288	337
32000	245	205	207	246	270
64000	218	186	171	190	217
128000	207	170	168	190	195

Table 4: Error numbers in the final experiments

Number of experiments	Without distortions	Rule 1		Rule 2	
		4 dist.	8 dist.	4 dist.	8 dist.
1	72	68	63	75	75
2	86	65	65	72	67
3	83	66	62	68	72
Mean value	80	66	**63**	72	71

The error number 63 corresponds to 99.37% of recognition rate.

5 Conclusions

New classifier based on perceptron of Frank Rosenblatt is proposed for handwritten digit recognition. Experiments on MNIST database showed that this classifier has the best recognition rate (99.37 %) among other classifiers proved on this database. Training time (65 hours) is reasonably good. The main drawback of this classifier is relatively low speed of recognition (1-3 images per second on the Pentium 500 MHz). For this reason the classifier is to be used when the speed the recognition is not critical.

New efforts are necessary to improve the recognition speed of proposed classifier.

With the recent advances in computer hardware, it can be believed that this (speed) problem will be resolved.

Acknowledgement

We are grateful to Yan LeCun for his database MNIST which can be used by anyone who wants to test his classifier.

This research has originated at Glushkov Institute of Cybernetics, Kiev, Ukraine, in collaboration with Lora Kasatkina and Vladimir Lukovich. This work was funded by the projects CONACYT 33944-U, DGAPA IN-118799 and MEC 0104.

References

[1] Kussul, E., Baidyk, T., Kasatkina, L., Lukovich, V "Rosenblatt Perceptrons for Handwritten Digit Recognition". *Proceedings of International Joint Conference on Neural Networks IJCNN, 2001*, V.2, 2001, pp. 1516 –1520.

[2] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., "Gradient-based Learning Applied to Document Recognition", *Proceedings of the IEEE,* v. 86, N 11, November 1998, pp. 2278-2344.

[3] Hoque M.S., Fairhurst M.C. "A Moving Window Classifier for Off-line Character Recognition". *Proceedings of the 7-th International Workshop on Frontiers in Handwriting Recognition*, 2000, Amsterdam, pp.595-600.

[4] Rosenblatt, F., Principles of Neurodynamics, Spartan books, New York, 1962.

[5] Kussul, E., Baidyk, T., Lukovitch, V., Rachkovskij, D., "Adaptive High Performance Classifier Based on Random Threshold Neurons", *Cybernetics and Systems'94, World Scientific Publishing Co.Pte.Ltd*, Singapore, pp. 1687-1695.

[6] E. Kussul, T. Baidyk, "Neural Random Threshold Classifier in OCR Application", *Proc. of the Second All-Ukrainian Intern. Conf. "UkrOBRAZ'94*, Ukraine, December 1994, pp. 154-157.

[7] E. Kussul, L. Kasatkina, D. Rachkovskij, D. Wunsch, "Application of Random Threshold Neural Networks for Diagnostics of Micro Machine Tool Condition", *International Joint Conference on Neural Networks,* NJ Alaska, 1998, v. 1, pp. 241-244.