Active and Passive Elec. Comp., 1996, Vol. 19, pp. 1-12
Reprints available directly from the publisher
Photocopying permitted by license only

EFFECT OF LEARNING RATE ON THE RECOGNITION OF IMAGES

M. HAMED AND A. EL DESOUKY
Faculty of Engineering, Suez Canal University, Port Said, Egypt.

(Received Janaury 5, 1995; in final form April 3, 1995)

This paper presents a study for the effect of learning rate on an approach for texture classification and detection based on the neural network principle. This neural network consists of three layers, which are input, output, and hidden layers. The back propagation technique is considered. A computer algorithm is deduced and applied. In this work, the synthetic textures are generated. The results are taken for the modern computer of AT 486 type. The mathematical analysis is summarized in order to illustrate the effect of learning rate parameter on the exact discrimination during processing. This effect is studied through applications. The minimum consumed time for the computational time of classification in industry is correlated to correspond only the use of only 2 units in the hidden layer of a neural network for real images instead of 11 units.

KEYWORDS: *neural, computer, texture, classification, discrimination, Walsh, vision system, pattern recognition, data reduction, training, test, hidden.*

INTRODUCTION

Texture analysis is usually required for pattern recognition because it can provide exact information about the arrangement and spatial properties of image fundamental elements. Such textural information is complementary to multispectral analysis of an image in order to characterize a digital image [1]. Since the feature extraction appears to be the most important task, information about the textural characteristics of the original image can efficiently be embodied. These features should be implemented for either the description or classification of different texture images using any one of a multitude of pattern recognition techniques [2].

It is known that structural, statistical, and hybrid approaches are the major methods for a textural feature extraction. A statistical approach is suitable for the computation of a set of scalar features describing the distribution of intensities or local features ignoring their spatial interdependence while a structural approach concentrates on the spatial interaction of elementary regions, local features, or intensities [3]. The hybrid approach combines both statistical and structural approaches.

PROBLEM FORMULATION

The deduced features on the basis of the above mentioned techniques may be insufficient to indicate a complete representation for the purpose of exact analysis.

The results of these techniques depend mainly on the feature extraction, although it is difficult to extract a general complete set of features for all possible images. It should be indicated that these techniques are not easy to be parallel-implemented in addition to the fact that most of them will require a preprocessing stage for the purification of images from noise and interference, which appeared in the data of features extracted [4,5].

On the other hand, the use of neural networks helps to overcome all these problems without any human supervision. It must be noted that neural networks are valuable on several counts. Firstly, they are adaptive: they can take data and learn on it. Thus, they transfer solutions from the data presented to capture quite subtle relationships. Neural networks can reduce the development time by learning underlying relationships even if they are difficult to find and describe. They can also solve problems that lack existing solutions. Secondly, neural networks can generalize the solution: they can correctly process data that only broadly resembles the original trained data. Similarly, they can handle imperfect or incomplete data, providing a measure of fault tolerance. Generalization is useful in practical applications because real world data is noisy. Thirdly, the neural networks are nonlinear, in that they can capture complex interactions among the input variables in a system. Fourthly, neural networks are highly effective working in parallel [4,5].

There are eight different parameters that affect the learning time [6], such as: the number of units in both input and output layers; the number of hidden layers; the number of units in hidden layer; the fan in and fan out of hidden units; the number of patterns in the training set; the learning rate; the distribution of initial weights and biases; and the order of pattern presentation.

A multilayer neural network can be designed and trained on the basis of a back propagation technique to assign a texture to its corresponding class. The effect of different parameters or some of them in such a system may be studied.

THE SYSTEM

The data in texture imaging can be compressed with the help of Walsh data compressor, which has the advantage of feature compaction and data reduction to the quarter. Walsh Transform is necessary for reaching the highest possible degree of data reduction, due to the nature of the first four rectangular Walsh basis functions [7].

In the analysis used, the texture images of size (64 * 64) should be passed to a Walsh Transform based data compressor, and these samples are used for training a neural network. The effect of change of the number of units in the hidden layer on the computational time required in the training phase of the neural network is examined [8].

Texture discrimination system (TDS) is a form of computer manipulation of imagery, in which the input is an image and the output is its description enhancement, pattern recognition, and transform coding for data compression [9]. Recently, texture discrimination systems in industry have received an increased attention due to savings in the processing time due to exact results without the need of human effort. It modifies the quality control and so improves product quality, performing 100% product inspection. This system will be of more importance because it could be integrated with production machines in the site.

The used system (as shown in fig. 1(a) consists of three stages. The first one appears to be the video data acquisition system, which includes the video camera as well as analog to digital converter. The second stage is defined as the feature extractor (neural network). The last third stage may be known as the classifier in order to assign an incoming texture image to the corresponding class [8].

ALGORITHM

Since the back propagation concept depends on an iterative gradient descent principle, the system may be designed in order to minimize the mean square error between the actual output of a multilayered feedforward network (MLFN) and the desired output. The algorithm is performed in two successive steps; the propagation and the backpropagation stages. In the propagation phase, a pattern vector at the input layer together with its desired output pattern at the output layer must be simultaneously applied to the network. The error detected at the output layer is

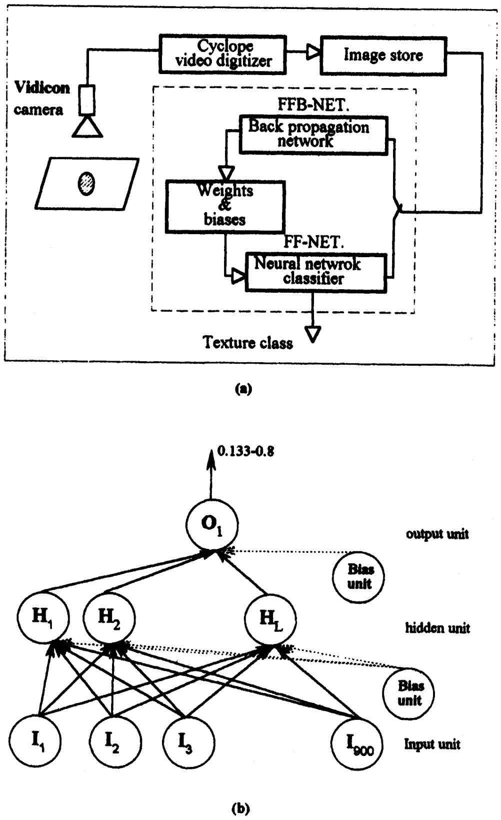

FIGURE 1 The used system (a): The experimental set up. (b): Illustration of the used neural network.

then back propagated through the network (BPN) to update the connection weights according to the generalized delta rule [10]. This process may be repeated until the average system—error goes under pre-specified limit where the procedure is terminated. As the network is learnt, it becomes capable of classifying any new input pattern vectors. The flow chart that describe the steps of back propagation network is illustrated in fig. 2.

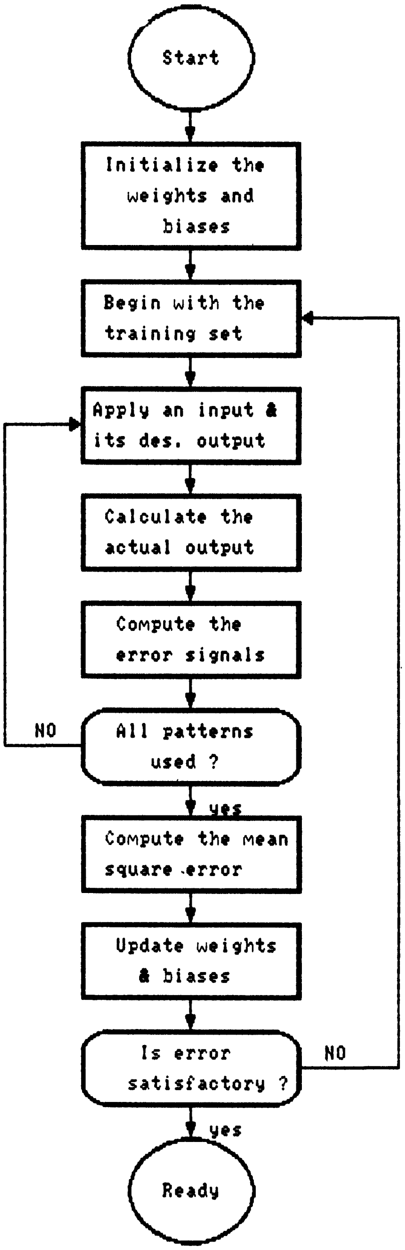
Start
Initialize the weights and biases
Begin with the training set
Apply an input & its des. output
Calculate the actual output
Compute the error signals
All patterns used ?
NO
yes
Compute the mean square error
Update weights & biases
Is error satisfactory ?
NO
yes
Ready

FIGURE 2 The flow chart of backprobagation neural network.

The relevant mathematical equations for the BPN are summarized to explain the sequence of steps during training for a single training vector pair. Therefore, applying the i/p vector X_p to the input units as a function of the image data X_{pn} to each neuron from the first 1 up to the n neuron of the input layer in thè form:

$$X_p = (x_{p1}, \quad x_{p2}, \ldots \quad , x_{pn}) \tag{1}$$

The net input values to the hidden layer units ($\text{net}_{pj}{}^h$) can be formulated as a function of the weight $W^h{}_{ii}$ on the connection from the i th input unit and the bias term $\Theta_j{}^h$ by the expression:

$$\text{net}\,^h_{pj} = \sum_{i=1}^{n} W\,^h_{ji} X_{pi} + \theta\,^h_j \tag{2}$$

It should be noted that the bias term represents a weight on a connection that has its input value (always equal 1), as well as the superscript "h" refers to quantities on the hidden layer, while the outputs i_{pj} from the hidden units can be determined via a sigmoidal function of the weighted sum of its inputs. Therefore, the net-input values $\text{net}_{pk}{}^o$ to each unit would be written in the form:

$$\text{net}\,^o_{pk} = \sum_{j=1}^{L} W\,^o_{kj} X_{pj} + \theta^o_K \tag{3}$$

where the superscript "O" refers to the quantities on the output layer.

Consequently, the outputs O_{pk} can be then calculated.

The activity at the output layer must be compared to the desired output so that the error terms $\delta_{pk}{}^o$ for the output units could be computed, accordingly, through the mathematical expression:

$$\delta^o_{pk} = (Y_{pk} - o_{pk})\, f\,^o_k\,(\text{net}^o_{pk}) \tag{4}$$

where y_{pk} is the desired output and O_{PK} is the output of the k-th unit to the p-th input/output pair at the output layer, while the error term for the hidden units ($\delta_{pj}{}^h$) will be defined, respectively.

It is noticed that the error on the hidden units are calculated before the connection weights to the output layer units have been updated. Therefore, updating weights of the output layer on the basis of the learning-rate parameter η, where it is positive and is usually less than 1 according to the formula:

$$W\,^o_{kj}\,(t+1) = W^o_{kj}(t) + \eta\delta^o_{pk}\, i_{pj} \tag{5}$$

and then, updating weights on hidden layer again before getting the final form of squared error E_p using the expression

$$w^h_{ji}\,(t+1) = w^h_{ji}\,(t) + \eta\delta\,^h_{pj}\, x_i \tag{6}$$

then finally, the measure of which is known as the error term and may be expressed in the final form as:

$$E_p = \frac{1}{2}\sum_{k=1}^{M} \delta_{pk}^{2} \qquad (6)$$

where M is the total number of neurons on the output units. When the error is acceptably small within the specified value for each of the training vector pairs, training will be stopped.

From equations (5), (6) it is seen that the learning rate affect directly the image processing and consequently, the computational time. So, the investigation in the effect of learning efficiency should be very needed.

The samples of synthetic texture (texture size 30 * 30 pixel) are generated as an array of values between zero & 1. The lines of array are concatenated (line after the other to construct single dimensional vector x(i) i = (1,.........,900), then, the input layer consists of 900 neurons. Fig. (3) presents these studied texture images.

The neural network performs non-linear transformations on its summed inputs and produce output between 0.133 & 0.8 in steps 0.1333. The weights of connections between the neurons are iteratively computed, and the training process continues until the weights give output values in the output layer equal to the target or desired outputs [8].

Since back-propagation is a time-consuming learning technique, the vital parameters affecting learning speed which are the distribution of intial weights and biases, and order of pattern presentation which have been studied in [8].

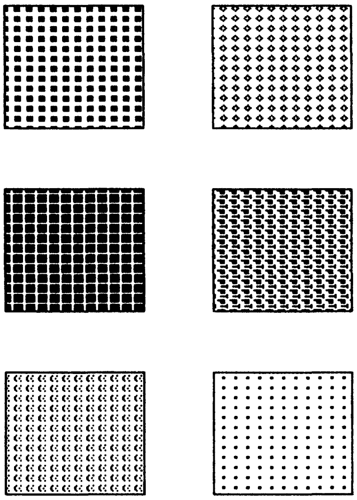

FIGURE 3 Different shapes of synthetic textures.

The effect of number of hidden units on the consumed computational time in the learning phase is studied. The different numbers of learning cycles required for the texture classification network on the type of computer DX 486 as well as the consumed computational time in the variation of the numbers of units in a hidden layer in the range 5–15 units are given, according to ref. [8]. These results have been drawn in fig. (4). It has been concluded that the number of units in the hidden layer equal to 11 takes the lowest learning cycles and consequently the minimum consumed time.

STATISTICAL STUDY

The following random number generating pascal functions are used in this study to initialize the synaptic connections using four different functions: resetl, reset2, reset3, or reset4. The order of pattern presentation to the input layer is determined by one of the following presentation functions: presentl, present2, present3, and present4.

Random generator is used but with four different principles described below. The first principle of generation in the number randomly is based on the mathematical equation a = (a * 10009 + 1) where the second random may be expressed by rand2 = 0.5 (1 − a/maxint). Then the final number can be deduced according to rand = rand2 (−sqr (random (maxint)).

Otherwise, the second generation will be based as random (a) of the available compilers but the third one may take the 3- Gaussian random numbers, which is defined by sqr (ln (1/×2) sin (2. pi. x) while the last number is generated with the formula rand2(random (maxint)) [8].

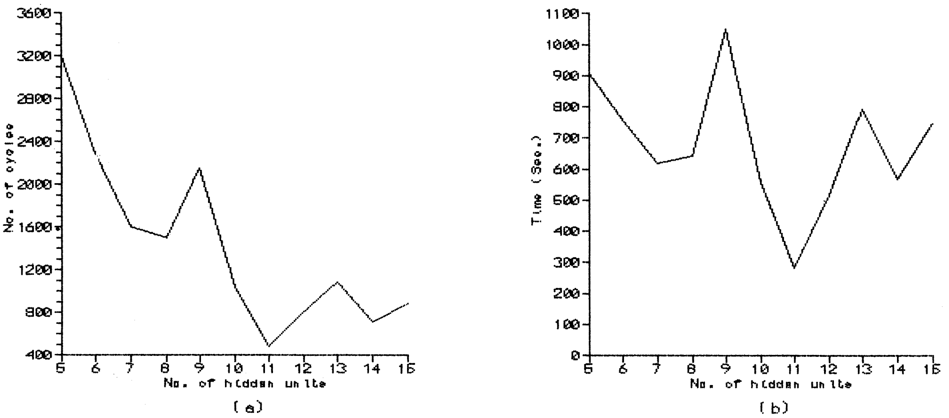

FIGURE 4 The dependence of numbers of hidden units for synthetic textures. (a): For computational time. (b): For no. of cycles.

These four different concepts for random generations are used and the results are presented. Results contain the procedures and functions that generate the random numbers against the number of training cycles and the training time of the classification with various numbers of units in the hidden layer in the range from 6 to 12. Tables 1 and 2 list the results deduced from the applications, where it is remarkable that the case of 11 hidden units (reset2 and present1) gives training speed much faster than all other random number generators. The neural network reached the steady state after a training for 580 cycles and consumed time equal to 3 minutes and 37 seconds. Also, this time is reduced from 4 minutes and 44 seconds as shown from table 1 to 3 minutes and 37 seconds in table 3 by using such a function (reset2 and present1 or present4).

Then, this may save greatly the computational time leading to speeding the processing sequence. Thus, the neural network used for classification of texture was composed of three layers of processing element whose input layer having 900 neurons with 11 neurons of hidden layers and 1 neuron of the output layer as shown in fig. 1(b).

The selection of a value for the learning rate parameter, η, may have a significant effect on the network performance. Usually, it must be a small number to ensure that the network will settle to a solution. A small value of η means that the network will have to make a large number of iterations leading to the need for increasing the size of η as learning proceeds. Increasing the decreased error will help to speed the convergence by increasing the step size since the error reaches a minimum, but the network may bounce around too far from the actual minimum value if η gets too large [10].

TABLE 1
Procedures and functions which generate different types of random numbers against the number of training cycles of the texture classification network with 6 up to 12 units in the hidden layer.

Random Number Generator	Order of Pattern Present	No. of cycles with 6 units	No. of cycles with 7 units	No. of cycles with 8 units	No. of cycles with 9 units	No. of cycles with 10 units	No. of cycles with 11 units	No. of cycles with 12 units
Reset1	present1	13610	2920	2210	2660	660	1360	1000
	present2	14200	3140	2380	2710	730	1630	900
	present3	16560	4640	3370	4090	980	2090	1240
	present4	13830	3020	2450	2820	730	1370	970
Reset2	present1	3040	2390	2120	1230	1700	580	1280
	present2	4070	2410	2210	1100	1750	590	1120
	present3	41330	3270	3200	1730	2440	900	1620
	present4	3220	2690	2330	1160	1720	580	1270
Reset3	present1	1820	9530	3970	3030	750	760	730
	present2	2240	9490	3800	2740	650	690	600
	present3	2360	12780	5200	3680	840	1070	1020
	present4	2810	9550	3790	2570	660	660	930
Reset4	present1	6160	5290	1190	1020	1210	1020	740
	present2	6450	5570	1260	1180	1230	1250	650
	present3	8520	7280	2030	1340	1750	1400	990
	present4	6320	5290	1380	950	1240	940	690

TABLE 2
Procedures and functions which generate different types of random numbers against the training time of the texture classification network with 6 up to 12 units in the hidden layer.

Random Number Generator	Order of Pattern Present.	Computa. time with 6 units	Computa. time with 7 units	Computa. time with 8 units	Computa. time with 9 units	Computa. time with 10 units	Computa. time with 11 units	Computa. time with 12 units
Reset1	present1	0:48:54	0:12:3	0:9:58	0:13:51	0:3:45	0:8:24	0:6:43
	present2	0:50:33	0:12:58	0:10:45	0:14:5	0:4:9	0:10:4	0:6:3
	present3	0:57:9	0:19:5	0:15:8	0:21:10	0:5:33	0:12:51	0:8:17
	present4	0:49:29	0:12:26	0:11:1	0:14:36	0:4:9	0:8:28	0:6:32
Reset2	present1	0:10:47	0:9:53	0:9:35	0:6:26	0:9:37	0:3:37	0:8:35
	present2	0:16:32	0:9:57	0:9:32	0:5:45	0:9:54	0:3:40	0:7:31
	present3	1:58:46	0:13:32	0:14:29	0:9:3	0:13:45	0:5:35	0:10:50
	present4	0:11:23	0:11:10	0:10:35	0:6:6	0:9:44	0:3:38	0:8:31
Reset3	present1	0:6:26	0:35:54	0:17:56	0:15:47	0:4:15	0:4:42	0:4:54
	present2	0:7:57	0:35:45	0:16:58	0:14:13	0:3:42	0:4:16	0:4:2
	present3	0:8:21	0:47:18	0:21:35	0:15:6	0:4:45	0:6:36	0:6:49
	present4	0:9:56	0:35:57	0:16:45	0:13:22	0:3:45	0:4:5	0:6:14
Reset4	present1	0:23:39	0:21:46	0:5:24	0:5:19	0:6:52	0:6:20	0:4:59
	present2	0:24:27	0:23:4	0:5:42	0:6:11	0:6:58	0:7:45	0:4:23
	present3	0:30:22	0:28:44	0:9:8	0:7:1	0:9:53	0:8:40	0:6:39
	present4	0:24:6	0:20:16	0:6:16	0:4:58	0:7:1	0:5:50	0:4:40

The results of the above tables are considered when the value of learning rate parameter η is equal to 0.5. We studied the effect of change of the learning rate on speeding up the training rate of the neural network with the number of units in the hidden layer equal to 11 and by using the function reset2 and present 1 as illustrated in fig. 5. From fig. 5 it is noticed that as the learning rate parameter η is increased, computational time is decreased up to the value of learning rate parameter η of 0.8.

The network reached a steady state with number of trails equal to 380 cycles and computational time of 2 minutes and 26 seconds.

DETECTION PROCESSING

In recent years, a remarkable progress in technology of manufacturing of solid insulating products can be observed in the field of H.V. engineering, but even advanced technology does not completely protect the insulation from the existence

TABLE 3
Results of actual defects on H.V. insulating surfaces.

Texture class	Neural network output
Texture with horizontal stroke	0.2
Texture with vertical stroke	0.4
Texture with area surface defect	0.6
Normal insulator surface	0.8

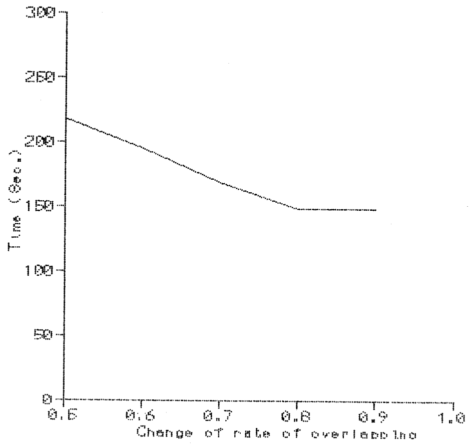

FIGURE 5 The effect of change of rate of overlapping on computational time.

of defects that may act as partial discharge sources deteriorating solid insulation and may be it leads to break down either the insulation itself or its surface resistance [11]. This problem will be more necessary if the defect is very small. So, the detection in insulating surface appears to be one of the vital examples for application of the presented method.

Defects are classified into three groups as horizontal, vertical, and area defects in addition to the fourth case representing the normal. These four samples of real

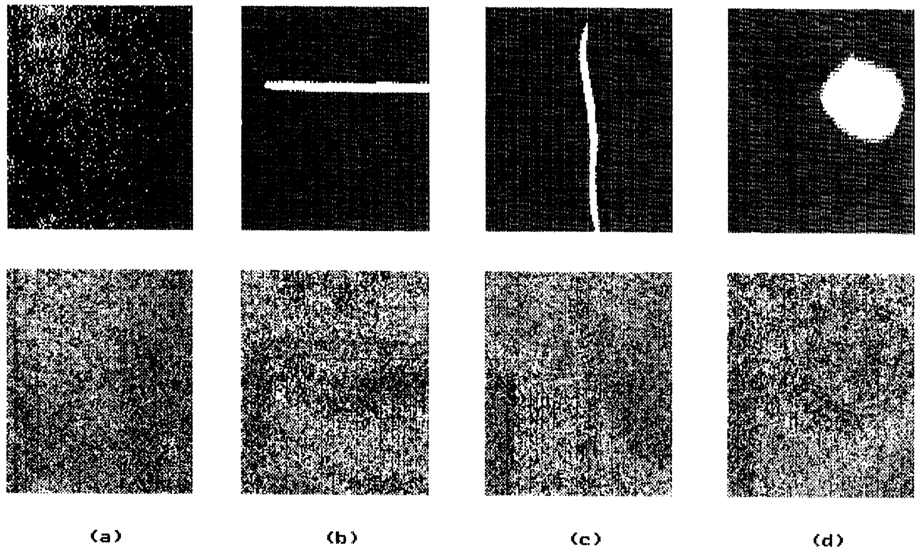

FIGURE 6 The shapes of tested images for real objects. (a): The normal binary image. (b): The binary image with horizontal defect. (c): The binary image with vertical defect. (d): The binary image with area defect.

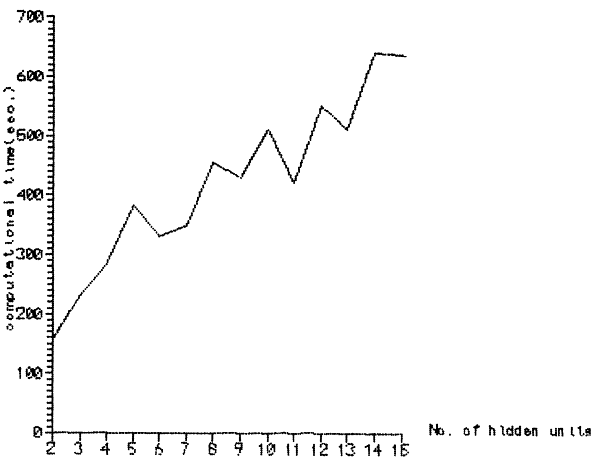

FIGURE 7 The required time in neural network classification with real images.

textures are photographed from the surface of real high voltage insulators as shown in fig. 6. The four texture images of size (64 * 64) are passed to give algorithm for the training of a neural network [8].

The effect of change of the number of hidden units in the hidden layer on speeding up the training phase of the neural network was examined in [8] and concluded that the best number of hidden units is 11 hidden units. The network reached a stable state with 11 hidden units after 8228 trails and after 7 minutes and 2 seconds.

In this paper, after the purification of the computer program from any impurities that may consume a time, the effect of change of hidden units from 2 up to 15 with computational time is retried to check the published number of units in the hidden layer. Fig. (7) shows the relation of number of hidden units of real textures according to the computational time. It is seen that only 2 hidden units in the hidden layer is quite enough; although it takes a higher number of training cycles of 12440 trials, it takes minimum computational time of 2 minutes and 38 seconds. Thus, the architecture of the network for classification consists of three layers (fig. 1 (b)). The number of neuron in the first layer are 128 (equal to the number of Walsh Transform coefficients of the real texture) while the hidden layer includes only 2 neurons. The single element in the output layer provides the response of the network to be as given in table 3. Thus, the back propagation training technique is suitable for the defect classification task.

CONCLUSIONS

From this investigation it is concluded that:

- Three layer neural network is still quite enough for exact discrimination.
- The number of hidden units affect on speeding up the training of a neural network.

- **Only 11 units in the hidden layer of a neural network are for exact discrimination in synthetic texture while only 2 units are enough for real image analysis.**

- **The given technique can be effectively used for surface detection in industry with minimum consumed time and a great reduction in the human need.**

- **The presented network concept reaches the steady state in a learning rate of 0.8 with its minimum consumed computational time.**

REFERENCES

1. D. Chen and L. Wang (1990): Texture Features Based On Texture Spectrum centerd applications et de recherches en teledetection (cartel), Universite de sherbrooke (Quebec) Canada JIK 2R1.
2. R.M. Pickett, B.S. Lipkin and A. Rosenfeld (1970): Visual Analysis Of Texure In The Detection And Recognition Of Object picture processing and psychopictorics, NEW YORK AND LONDON, ACADEMIC PRESS, 289–308.
3. D. Chetverikov (1984): Measuring The Degree Of Texture Regularity, computer and automation institute Hungarian Acadamy of Sciences, Budapest, POB 63- H-1502 Hungary.
4. M. Hamed, I.A. Amin, A. Alam Eldin, I.F. El. Nehary and H.A. Nour Eldin (1994): A New Concept of Shape Recognition Using Neural Networks, Bull. of Scientific Engineering, Port Said, Egypt, Vol.6, 81–95.
5. M. Hamed, I.A. Amin, A. Alam Eldin, I.F. El. Nehary and H.A. Nour Eldin (1994): A New Method Of Colour Image Classification Using Parallel Connected Strips, Bull. of Scientific Engineering, Port Said, Egypt, Vol.6, 15–27.
6. A.S. Tolba (1993): A Neural Network For Arabic Character Recognition, Mansoura Engineering Journal (MEJ) Vol.18, No. 4, March, 61–71.
7. I.I. Ibrahim and A.S. Tolba (1993): A Neural Network For Recognition Of Handwritten Arabic Characters, Ein—shams engineering Journal, 1–12.
8. A. El-Desouky (1995): Texture Analysis And Its Application On High Voltage Insulating Surfaces. A Thesis for M.SC, Suez Canal University, Egypt.
9. M. Hamed, A. Tolba and H. El Hendy (1994): Pattern recognition of plane objects, Syria, Damascus, 34th Sience WeekConf., Nov. 5–10, 1994.
10. J.A. Freeman and D.M. Skapura (1991): Neural Networks: Algorithms, Applications, Arogramming Techniques, Addision-Wesley Publishing Company.
11. N. Farrag and Et al (1995): Partial Discharges Responsible For Degradation And Breakdown In Solid Syntatic Insulations, AMSE, MODELLING, MEASUREMENTS & CONTROL, A, VOL. 60, No. 1, 23–29.