Proceedings of International Joint Conference on Neural Networks, Montreal, Canada, July 31 - August 4, 2005

A variable-parameter neural network trained by improved genetic algorithm and its application

S.H. Ling
Centre of Multimedia Signal Processing,
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

H.K. Lam
Centre of Multimedia Signal Processing,
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

F.H.F. Leung
Centre of Multimedia Signal Processing,
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

Abstract - This paper presents a neural network with variable parameters. These variable parameters adapt to the changes of the input environment, and tackle different input data sets in a large domain. Each input data set is effectively handled by its corresponding set of network parameters. Thus, the proposed neural network exhibits a better learning and generalization ability than a traditional one. An improved genetic algorithm [1] is proposed to train the network parameters. An application example on hand-written pattern recognition will be presented to verify and illustrate the improvement.

I. INTRODUCTION

It is well known that a neural network can approximate any smooth and continuous nonlinear functions in a compact domain to an arbitrary accuracy [7-8]. The 3-layer feed-forward neural networks have been successfully applied in wide range of applications such as system modeling and control, prediction [2], recognition [3], etc. Thanks to its specific structure, a neural network can be used to realize a learning process [5, 8-9], which consists of two steps: designing a network structure and choosing an algorithm for the learning process. The structure of the neural network governs the non-linearity of the modelled function. The learning algorithm is used to provide a rule to optimize the weight' values within the training period. A typical neural network structure offers a fixed set of weights after the learning process. This single set of weights is used to model all input data sets. However, a fixed set of weights may not be enough to learn the data sets if the data sets are distributed in a vast domain separately and/or the number of network parameters is too small.

One of the important issues for a neural network is the learning or training process. The learning process aids to find a set of optimal network parameters. The gradient rules [7-8], such as the MRI, MRII, MRIII rules and the backpropagation techniques, adjust the network parameters based on the gradient information to reduce the mean square error over all input patterns. However, the derivative information of the optimized function is necessary (the error function is thus required to be continuous and differentiable), and the learning process is easily trapped in a local optima, especially for multimodal problems. The learning rules are also network-structure dependent. Some global search algorithms such as Tabu search [5], simulated annealing [5] and genetic algorithm [5-6] were proposed. Unlike the gradient descent based algorithms, these search algorithms are less likely to be trapped in a local optima and do not need a differentiable or even continuous error function. Thus, these search algorithms are more suitable for searching in a large, complex, non-differentiable and multimodal domain [10].

Genetic algorithm (GA) is a directed random search technique [4-6] that is widely applied in optimisation problems. It is especially useful for complex optimisation problems when the number of parameters is large and the analytical solutions are difficult to obtain. GA can help find out the optimal solution globally over a domain. It has been applied in different areas such as fuzzy control, path planning, greenhouse climate control, modelling and classification, etc.

In this paper, a variable-parameter neural network tuned by an improved GA is proposed [1]. It consists of two units, namely the rule-base (RB) neural network and the data-processing (DP) neural network as shown in Fig. 1. The RB neural network stores some rules governing how the DP neural network handles the input data. By using this proposed neural network, some cases that cannot be handled by the traditional neural networks with a limited number of parameters can now be tackled. To illustrate this point, a figure with two sets of data S1 and S2 separated in a far distance as shown in Fig. 2 is used. In general, there can be a lot of data sets separated in far distances within a large domain. If we solve this problem using a traditional neural network, the weights of the neural network are trained to minimise the error between the network output and the desired value in a mapping problem. However, for a limited number of parameters, the network may only model the data set S instead as shown in Fig. 1. In order to alleviate this problem, the architecture of the neural network shown in Fig. 1 is proposed in this paper. Referring to Fig. 1, when the input data belongs to S1, the RB neural network will follow rule set 1 to drive the DP neural network to handle the S1 data. Similarly, when the input data belongs to S2, the rules corresponding to S2 will be employed to drive the DP neural network to handle this input data. In other words, it operates like two individual neural networks handling the corresponding input data. An improved GA [1], which has good performance in multimodal problems, can be employed to train the parameters of the proposed neural network.

This paper is organized as follows. The variable parameter neural network will be presented in section II. Training the variable parameter neural network using the improved GA will be presented in section III. Application examples will be given in section IV to illustrate the applicability of the proposed approach. A conclusion will be drawn in section V.

II. VARIABLE PARAMETER NEURAL NETWORK

The proposed 3-layer, fully-connected feed-forward variable-parameter neural network is shown in Fig. 3. Comparing with the traditional 3-layer feed-forward fully-connected neural network [7], the main differences are the variable-parameter hidden and output nodes. In the proposed network, the parameters of the activation functions vary according to some intermediate signals of the proposed network. Consequently, the proposed network can be made adaptive to the contingent changes of the environment. The learning and generalization abilities of the network are thus enhanced.

Referring to Fig. 3, $\mathbf{x}(t) = \begin{bmatrix} x_1(t) & x_2(t) & \cdots & x_{n_{in}}(t) \end{bmatrix}$ denotes the input vector, n_{in} denotes the number of input nodes; t denotes the current number of input vector, which is a non-zero integer; w_{ji}^1, j = 1, 2, …, n_h, i = 1, 2, …, n_{in}, denote the connection weights between the input layer and the hidden layer; n_h denotes the number of hidden nodes; w_{kj}^2, k = 1, 2, …, n_{out}, j = 1, 2, …, n_h, denote the connection weights between the hidden layer and the output layer; n_{out} denotes the number of output nodes. m_j^1, r_j^1, m_k^2 and r_k^2 are parameters related to a proposed activation function of the hidden and output nodes, $tf(\cdot)$. The details of the proposed network will be presented as follows.

A. Proposed Hidden Node and Output Node

Fig. 4 shows the details of the hidden and output nodes with the proposed activation function employed in the them. In this figure, $z_1(t)$ to $z_n(t)$ denote the input of the node; w_1 to w_n denote the connection weights; m and r denote the intermediate connection weights. The output of the summation block, $f_s(t)$, is given by,

$$f_s(t) = \sum_{i=1}^{n} w_i z_i(t) \quad (1)$$

The output of the node can be written as,

$$f(t) = tf\left(f_s(t), m, r\right) \quad (2)$$

where the activation function is to evaluate the fitness of the input and is defined as,

$$tf\left(f_s(t), m, r\right) = \frac{2}{1 + e^{\frac{-(f_s(t)-(m+\Delta m))}{2((r+\Delta r))^2}}} - 1 \in \begin{bmatrix} -1 & 1 \end{bmatrix} \quad (3)$$

$$\Delta m = \left(\frac{2}{1+e^{-2 f_s(t)}} - 1\right) m \quad (4)$$

$$\Delta r = \left(\frac{2}{1+e^{-2 f_s(t)}} - 1\right) r \quad (5)$$

It can be seen from (3) that the proposed activation function is characterized by the mean (m) and standard deviation (r) respectively. When Δm and Δr are both zero, the values of m (which is functionally equivalent to the static bias of the traditional neural network) and r govern the zero-crossing point and the steepness of the activation function respectively. Δm and Δr are to adjust the zero-crossing point and the steepness of the activation function according to the value of $f_s(t)$. Referring to Fig. 1, the parameters m and r form the rule sets. From (4) and (5), the values of the Δm and Δr depend on the network inputs and the parameters m and r. In other words, it operates as if the neural network handles different input data with different network parameters Δm and Δr.

B. Input-Output Relationship of Proposed Neural Network

Referring to Fig. 3, the output is defined as,

$$y_k(t) = tf\left(\sum_{j=1}^{n_h} w_{kj}^2 f_j(t), m_k^2, r_k^2\right), \; k = 1, 2, \ldots, n_{out} \quad (6)$$

where

$$f_j(t) = tf\left(\sum_{i=1}^{n_{in}} w_{ji}^1 x_i(t), m_j^1, r_j^1\right), \; j = 1, 2, \ldots, n_h \quad (7)$$

denotes the output of the j-th hidden node. In the proposed neural network, the values of the parameters w_{ji}^1, w_{kj}^2, m_j^1, r_j^1, m_k^2 and r_k^2 will be trained by the improved GA [1]. After training, the values of these parameters will be fixed during the operation. The total number of tunable parameters of the proposed neural networks is $(n_{in} + n_{out} + 2)n_h + 2n_{out}$.

III. LEARNING WITH IMPROVED GENETIC ALGORITHM

An improved GA [1] will be employed to obtain the optimal parameters of the proposed neural network. The crossover and mutation operations of the improved GA are modified. To realize the modified genetic operations, the offspring spreads over the domain so that a higher chance of reaching the global optimum can be obtained. Let the input-output relationship of the modified neural network be described by,

$$\mathbf{y}^d(t)=\mathbf{g}\left(\mathbf{x}^d(t)\right),\ t=1,2,\ldots,n_d \quad (8)$$

where $\mathbf{x}^d(t)=\left[x_1^d(t)\quad x_2^d(t)\quad\cdots\quad x_{n_{in}}^d(t)\right]$ and $\mathbf{y}^d(t)=\left[y_1^d(t)\quad y_2^d(t)\quad\cdots\quad y_{n_{out}}^d(t)\right]$ are the given inputs and the desired outputs of an unknown nonlinear function $\mathbf{g}(\cdot)$ respectively, n_d denotes the number of input-output data pairs. The fitness function is defined as,

$$fitness=\frac{1}{1+err} \quad (9)$$

and

$$err=\sum_{k=1}^{n_{out}}\frac{\sum_{t=1}^{n_d}\left(y_k^d(t)-y_k(t)\right)^2}{n_{out}n_d} \quad (10)$$

is the mean square error (MSE). The objective is to maximize the fitness value of (9) using the improved GA by setting the chromosome to be $\left[w_{ji}^1\quad m_j^1\quad r_j^1\quad w_{kj}^2\quad m_k^2\quad r_k^2\right]$ for all i, j and k. The range of the fitness value of (9) is [0, 1]. It can be seen from (9) and (10) that a larger *fitness* implies a smaller MSE *err*.

IV. Application Example and Results

In this example, a pattern recognition problem is given to illustrate the learning and generalization abilities of the proposed neural network in a classification problem with a large number of input data sets. The proposed network is used to recognize hand-written graffiti. In this example, the digits 0 to 9 and three (control) characters (*backspace*, *carriage return* and *space*) are recognized by the modified neural network. These graffiti are shown in Fig. 5. A point of each graffiti is characterized by a number based on the x-y coordinates on a writing area. The size of the writing area is x_{max} by y_{max}. The bottom left corner is set as (0, 0). Ten uniformly sampled points of the graffiti are taken as the inputs of the interpreter. The points are taken in the following way. First, the input graffiti is divided into 9 uniformly distanced segments characterized by 10 points, including the start and the end points. Each point is labelled as (x_i, y_i), $i = 1, 2, \ldots, 10$. The first 5 points, (x_i, y_i), $i = 1, 3, 5, 7$ and 9, taken alternatively are converted to 5 numbers ρ_i respectively by using the formula $\rho_i = x_i x_{max} + y_i$. The other 5 points, (x_i, y_i), i = 2, 4, 6, 8 and 10, are converted to 5 numbers respectively by using the formula $\rho_i = y_i y_{max} + x_i$. These ten numbers, z_i, $i = 1, 2, \ldots, 10$, will be used as the inputs of proposed graffiti recognizer. The graffiti recognizer consisting of 5 modified neural networks as shown in Fig. 6 is proposed to perform the graffiti recognition. In this figure, the inputs are defined as follows,

$$\bar{\mathbf{x}}(t)=\frac{\mathbf{z}(t)}{\|\mathbf{z}(t)\|} \quad (11)$$

where $\bar{\mathbf{x}}(t)=\left[\bar{x}_1(t)\quad \bar{x}_2(t)\quad\cdots\quad \bar{x}_{10}(t)\right]$ denotes the normalized input vectors of the proposed graffiti recognizer; $\mathbf{z}(t)=\left[z_1(t)\quad z_2(t)\quad\cdots\quad z_{10}(t)\right]$ denotes the ten points in the writing area; $\|\cdot\|$ denotes the l_2 vector norm. Referring to Fig. 6, the function of the graffiti class selector is to divide the input graffiti classes into 4 sub-classes. In this example (Fig. 5), the graffiti "0(a)", "0(b)", "1" and "2" are arbitrarily assigned to class 1; the graffiti "3", "4", "5(a)" and "5(b)" are assigned arbitrarily to class 2; the graffiti "6", "7", "8(a)" and "8(b)" are assigned arbitrarily to class 3; the graffiti "9", "backspace", "carriage return" and "space" are assigned arbitrarily to class 4. To train the neural network of the graffiti class selector, a set of training pattern governing the input-output relationship will be employed. 1600 training patterns (100 patterns for each graffiti) will be used in this example. The training patterns consist of the input vectors and its corresponding expected output. The input-output relationship of the training patterns is defined such that the output $y_i(t)=1$ and others are zero when the input vector belongs to class i, i = 1, 2, 3, 4. The fitness function is given by (9), with

$$err=\sum_{k=1}^{4}\frac{\sum_{t=1}^{1600}\left(\frac{y_k(t)}{\|\mathbf{y}(t)\|}-\frac{y_k^d(t)}{\|\mathbf{y}^d(t)\|}\right)^2}{4\times 1600} \quad (12)$$

where $\mathbf{y}^d(t)=\left[y_1^d(t)\quad y_2^d(t)\quad y_3^d(t)\quad y_4^d(t)\right]$ denotes the expected output vector and $\mathbf{y}(t)=\left[y_1(t)\quad y_2(t)\quad y_3(t)\quad y_4(t)\right]$ is the actual network output defined as,

$$y_k(t)=tf\left(\sum_{j=1}^{n_h}w_{kj}^2 tf\left(\sum_{i=1}^{10}w_{ji}^1\bar{x}_i(t),m_j^1,r_j^1\right),m_k^2,r_k^2\right),\ k=1,2,3,4. \quad (13)$$

The index of the maximum output of the graffiti-class selector indicates the possible sub-class number of the input vector. This index is used to select the corresponding sub-class recognizer to perform the pattern recognition. As a result, only one sub-class recognizer activates each time. Each sub-class recognizer implemented by the proposed neural network has 10 inputs and 4-outputs, which is activated by the class selector. It can be seen from this approach that the sub-class recognizer limits the number of input patterns into 4 classes. In this way, the learning process is shared by the 4 sub-graffiti recognizers, which can be trained separately. Similar to the training process of the graffiti-class selector, 400 training patterns (100 for each graffiti) are employed to train each sub-class recognizer.

The desired network output $y_i^{\alpha}(t)=1$ (and others are zero) when the input vector belongs to class α, α = 1, 2, 3, 4 and input pattern i, i = 1, 2, 3 , 4. The fitness function for each network is given by (9), with

$$err=\sum_{k=1}^{4}\frac{\sum_{t=1}^{400}\left(\frac{y_k^{\alpha}(t)}{\|\mathbf{y}(t)\|}-\frac{y_k^{\alpha^d}(t)}{\|\mathbf{y}^d(t)\|}\right)^2}{4\times 400} \quad (14)$$

where

$\mathbf{y}_k^{\alpha^d}(t)=\left[y_1^{\alpha^d}(t) \quad y_2^{\alpha^d}(t) \quad y_3^{\alpha^d}(t) \quad y_4^{\alpha^d}(t)\right]$ denotes the expected output vector and $\mathbf{y}_k^{\alpha}(t)=\left[y_1^{\alpha}(t) \quad y_2^{\alpha}(t) \quad y_3^{\alpha}(t) \quad y_4^{\alpha}(t)\right]$ is the actual network output of the α-th sub-class recognizer defined as,

$$y_k^{\alpha}(t)=tf\left(\sum_{j=1}^{n_h}w_{kj}^2 tf\left(\sum_{i=1}^{10}w_{ji}^1\bar{x}_i(t),m_j^1,r_j^1\right),m_k^2,r_k^2\right),\ k=1,2,3,4 \quad (15)$$

The actual recognized class of the input pattern is indicated by the maximum output of the sub-class recognizer activated by the graffiti-class selector. For instance, the maximum output of the graffiti class selector is from $y_1(t)$. Then, the first sub-class recognizer will be employed to classify the input pattern. If $y_2^1(t)$ has the maximum output value, it can be concluded that the possible input pattern is "0(b)".

For comparison purpose, traditional 3-layer fully connected feed-forward neural networks [7] trained by the improved GA [1] are also used to replace the proposed networks in this example. For all cases, the initial values of the parameters of the neural network are randomly generated. The number of iteration to train the neural networks is 100000 for the graffiti class selector and 10000 for sub-class recognizer in both approaches. For the improved GA, the probability of acceptance (p_a) is set at 0.1 for both networks. The probability of mutation (p_m), the weight of crossover (w), the parameters of w_τ and w_f of the improved GA are 0.01, 0.5, 0.5 and 0.5 respectively for both networks. The population size is 10. All the results are the averaged ones out of 10 runs. The simulation results of both the approaches with different numbers of hidden nodes (numbers of parameters) are tabulated in Table I. For a large number of training sets, it can be seen from Table I that the graffiti-class selector implemented by the proposed approach outperforms that of the traditional approach in terms of training and testing fitness values. For a small number of training sets, it can also be seen from Table I that the sub-graffiti recognizers realized by the proposed and traditional approaches provide similar results in terms of training and testing fitness values.

In order to test the generalization ability of the proposed neural networks, a set of testing patterns consisting of 480 input patterns (30 patterns for each graffiti) is used. The recognition rates of the proposed and traditional neural networks validated by the training and testing patterns for each graffiti with different n_h are tabulated in Table II. Based on the results of Table I, the networks with the best fitness values will be used to realized the graffiti recognizer. For the proposed neural network, the graffiti-class selector with n_h = 25; sub-class recognizer 1 with n_h = 7; sub-class recognizer 2 with n_h = 15, sub-class recognizer 3 with n_h = 15; and sub-class recognizer 4 with n_h = 15 are employed; For the traditional neural networks, graffiti-class selector with n_h = 27; sub-class recognizer 1 with n_h = 16; sub-class recognizer 2 with n_h = 16; sub-class recognizer 3 with n_h = 16 and sub-class recognizer 4 with n_h = 11 are employed. In general, it can be seen from Table II that the recognition rate of the graffiti recognizer realized by the proposed neural networks for each graffiti is over 93%. The recognition rate of the graffiti recognizer realized by the traditional neural networks is not acceptable for the digit "6" (only 67%).

V. CONCLUSION

A variable parameter neural network has been proposed in this paper. The parameters of the proposed neural network are trained by the improved GA. Thanks to the variable parameters, the learning and generalization abilities of the proposed network have been increased. An application example on pattern recognition has been given to illustrate the merits of the proposed approach.

ACKNOWLEDGMENT

The work described in this paper was substantially supported by a grant from the Hong Kong Polytechnic University (Project No. G-YX31).

REFERENCES

[1] H.K. Lam, S.H. Ling, F.H.F. Leung, and P.K.S. Tam, "Function estimation using a neural-fuzzy network and an improved genetic algorithm," *International Journal of Approximate Reasoning*, vol. 30, no. 3, pp. 243-260, Jul. 2004.

[2] S.H. Ling, F.H.F. Leung, H.K. Lam, Y.S. Lee, and P.K.S. Tam, "A novel GA-based neural network for short-term load forecasting," *IEEE Trans. Industrial Electronics*, vol. 50, no. 4, pp.793-799, Aug. 2003.

[3] K.F. Leung, F.H.F. Leung, H.K. Lam, and S.H. Ling "On interpretation of graffiti digits and characters for eBooks: Neural-fuzzy network and genetic algorithm approach," *IEEE Trans. Industrial Electronics*, vol. 51, no. 2, pp.464-471, Apr. 2004.

[4] F.H.F. Leung, H.K. Lam, S.H. Ling, and P.K.S. Tam, "Tuning of the structure and parameters of neural network using an improved genetic algorithm," *IEEE Trans. Neural Networks*, vol. 14, no. 1, pp. 79-88, Jan. 2003.
[5] D.T. Pham and D. Karaboga, *Intelligent Optimization Techniques, Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks*. Springer, 2000.
[6] Z. Michalewicz, *Genetic Algorithm + Data Structures = Evolution Programs,* 2nd extended ed. Springer-Verlag, 1994.
[7] B. Widrow and M.A. Lehr, "30 years of adaptive neural networks: Perceptron, madaline, and backpropagation," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1415-1442, Sept. 1990.
[8] F.M. Ham and I. Kostanic, *Principles of Neurocomputing for Science & Engineering*. McGraw Hill, 2001.
[9] S. Haykin, *Neural Network: A Comprehensive Foundation*. Prentice Hall, 1999.
[10] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423-1447, Sept. 1999.

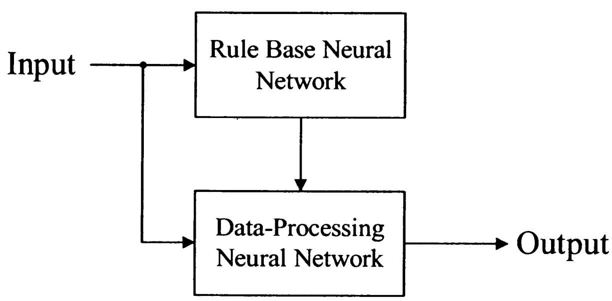

Fig. 1. Proposed architecture of the neural network.

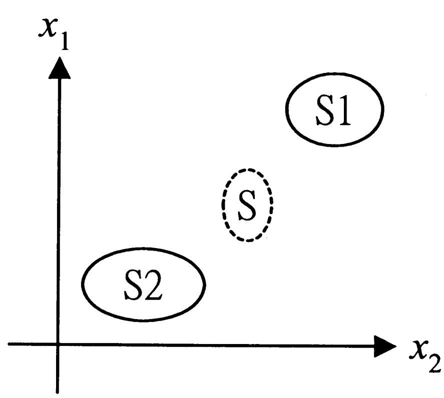

Fig. 2. Diagram showing two sets of data in a spatial domain.

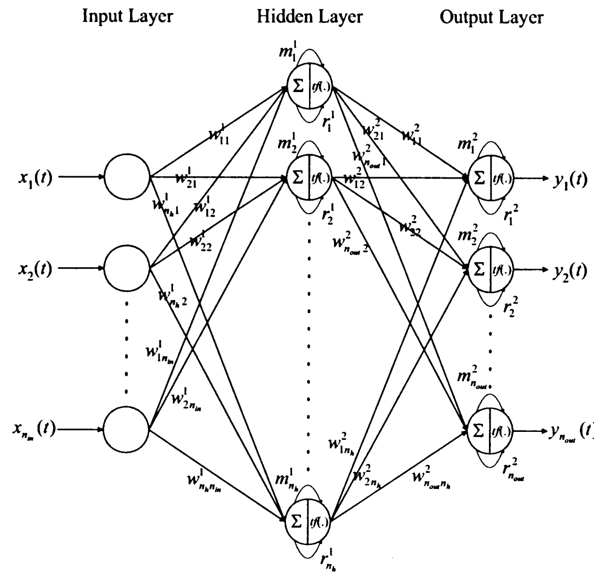

Fig. 3. Proposed variable parameter neural network.

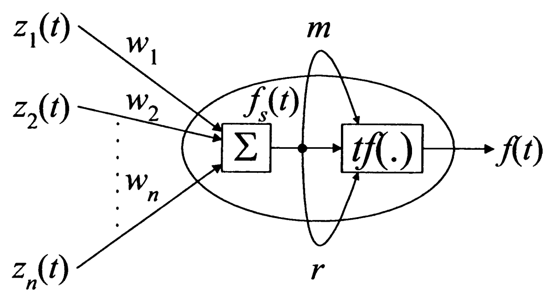

Fig. 4. Proposed hidden and output nodes.

Digits or Characters	Strokes	Digits or Characters	Strokes
0(a)		6	
0(b)		7	
1		8(a)	
2		8(b)	
3		9	
4		Backspace	
5(a)		Carriage Return	
5(b)		Space	

Fig. 5. Graffiti digits and characters (with the dot indicating the starting point of the graffiti).

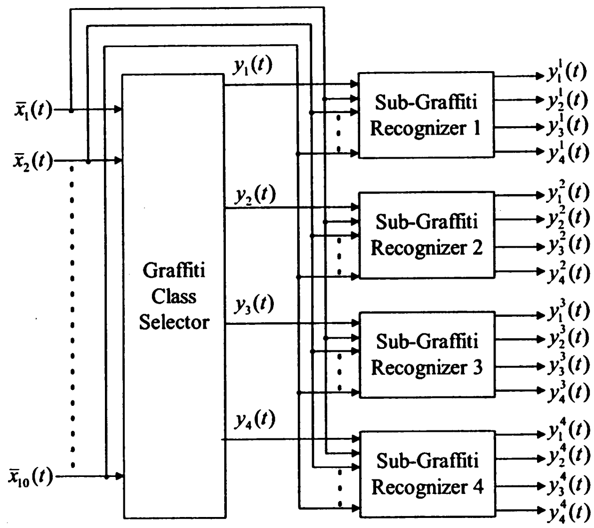

Fig. 6. Architecture of the graffiti recognizer.

TABLE. I. SIMULATION RESULTS OF BOTH APPROACHES FOR HAND-WRITTEN GRAFFITI RECOGNITION.

TABLE. I(A). PROPOSED NEURAL NETWORK.

	n_h (number of parameters)	Fitness value	Training error	Testing error
Graffiti Class Selector	10 (168)	0.993194	0.006852	0.022748
	15 (248)	0.994311	0.005722	0.013656
	20 (328)	0.995481	0.004540	0.018550
	25 (408)	0.997622	0.002384	0.011136
Sub-Graffiti Recognizer 1	5 (88)	0.995445	0.004576	0.004685
	7 (120)	0.999660	0.000340	0.000862
	10 (168)	0.996938	0.003072	0.005798
	15 (248)	0.996835	0.003175	0.005561
Sub-Graffiti Recognizer 2	5 (88)	0.992152	0.007910	0.016251
	7 (120)	0.991734	0.008335	0.017311
	10 (168)	0.994312	0.005720	0.015005
	15 (248)	0.994676	0.005352	0.011817
Sub-Graffiti Recognizer 3	5 (88)	0.990333	0.009761	0.049566
	7 (120)	0.993056	0.006993	0.027096
	10 (168)	0.994664	0.005365	0.042821
	15 (248)	0.995843	0.004175	0.032889
Sub-Graffiti Recognizer 4	5 (88)	0.996169	0.003845	0.008233
	7 (120)	0.995208	0.004815	0.009562
	10 (168)	0.997302	0.002706	0.008834
	15 (248)	0.999145	0.000855	0.005120

TABLE. I(B). TRADITIONAL NEURAL NETWORK.

	n_h (number of parameter)	Fitness value	Training error	Testing error
Graffiti Class Selector	11 (169)	0.972083	0.028719	0.039160
	16 (244)	0.984667	0.015571	0.024011
	22 (334)	0.986700	0.013480	0.032367
	27 (409)	0.988324	0.011814	0.025062
Sub-Graffiti Recognizer 1	5 (79)	0.997621	0.002385	0.002371
	8 (124)	0.998419	0.001583	0.001502
	11 (169)	0.998833	0.001168	0.001067
	16 (244)	0.999275	0.000725	0.000901
Sub-Graffiti Recognizer 2	5 (79)	0.992110	0.007953	0.014470
	8 (124)	0.994964	0.005061	0.006394
	11 (169)	0.995409	0.004612	0.008528
	16 (244)	0.996251	0.003763	0.004985
Sub-Graffiti Recognizer 3	5 (79)	0.992217	0.007844	0.030851
	8 (124)	0.994101	0.005934	0.035875
	11 (169)	0.997153	0.002855	0.020290
	16 (244)	0.997500	0.002507	0.022223
Sub-Graffiti Recognizer 4	5 (79)	0.998220	0.001783	0.006770
	8 (124)	0.999143	0.000858	0.005955
	11 (169)	0.999694	0.000306	0.004761
	16 (244)	0.999610	0.000390	0.005342

TABLE II. RECOGNITION ACCURACY RATE OF THE BEST PROPOSED NEURAL NETWORK AND TRADITIONAL NEURAL NETWORK WITH DIFFERENT NETWORK SETTING.

	Proposed network		Traditional network	
	Recognition accuracy rate (%) of the best network for the training patterns	Recognition accuracy rate (%) of the best network for the testing patterns	Recognition accuracy rate (%) of the best network for the training patterns	Recognition accuracy rate (%) of the best network for the testing patterns
0(a)	100.00	93.33	99.00	100.00
0(b)	100.00	100.00	100.00	100.00
1	100.00	100.00	98.00	100.00
2	99.00	96.67	100.00	100.00
3	99.00	100.00	100.00	100.00
4	100.00	96.67	98.00	93.33
5(a)	100.00	100.00	100.00	100.00
5(b)	100.00	100.00	100.00	100.00
6	100.00	100.00	98.00	66.67
7	98.00	100.00	99.00	93.33
8(a)	100.00	93.33	99.00	93.33
8(b)	100.00	93.33	100.00	86.67
9	99.00	100.00	99.00	100.00
Back Space	100.00	96.67	100.00	96.67
Return	100.00	100.00	100.00	100.00
Space	100.00	100.00	100.00	100.00