Digit and Command Interpretation for Electronic Book Using Neural Network and Genetic Algorithm

H. K. Lam, *Member, IEEE,* and Frank H. F. Leung, *Senior Member, IEEE*

***Abstract*—This paper presents the interpretation of digits and commands using a modified neural network and the genetic algorithm. The modified neural network exhibits a node-to-node relationship which enhances its learning and generalization abilities. A digit-and-command interpreter constructed by the modified neural networks is proposed to recognize handwritten digits and commands. A genetic algorithm is employed to train the parameters of the modified neural networks of the digit-and-command interpreter. The proposed digit-and-command interpreter is successfully realized in an electronic book. Simulation and experimental results will be presented to show the applicability and merits of the proposed approach.**

***Index Terms*—Digit and command interpretation, electronic book, genetic algorithm, neural networks.**

I. Introduction

INSPIRED by the human nervous system, the neural network was developed in the 1940s. The McCulloch–Pitts neuron, which was the foundation of the neural network development, was presented around 1943 [1]. No training algorithm was developed for this neuron. After that, a trainable adaptive linear (Adaline) element [1], [2], which is the basic building block used in many neural networks, has been proposed. A single Adaline is able to learn some of the linearly separable functions. In order to have a nonlinear separation boundary, layered networks such as multiple Adaline (Madaline) and multiplayer feedforward networks were proposed. The architecture of the Madaline network is a linear combination of some Adaline elements which form a single feedforward network structure. Based on these ideas and further exploration, different kinds of neural network were then developed, such as mapping networks, self-organizing networks, recurrent networks, radial basis function networks, etc., to fit different applications.

One of the important issues for the neural network is the learning or training process. The learning process aids to find a set of optimal network parameters. At the early stage, two major classes of learning rules, error correction and gradient rules, were proposed. The error correction rules [1], [2], such as the α-LMS algorithm, perception learning rules, and May's rule, adjust the network parameters to correct the network output error corresponding to the present input pattern. Some of the error-correction rules are only applicable to linear separable problems. The gradient rules [1]–[3], such as MRI, MRII, and MRIII rules and backpropagation techniques, adjust the network parameters based on the gradient information to reduce the mean square error over all input patterns. Different variations of backpropagation algorithms, such as backpropagation algorithms with momentum [4], backpropagation algorithms with variable learning rate [4], and conjugate gradient algorithms [5], were proposed to improve the learning time. A major weakness of gradient rules is that derivative information is necessary (the error function is thus required to be continuous and differentiable). As a result, the learning process is easily trapped in a local optimum, especially for multimodal problems and the learning rules is network-structure dependent. Some global search algorithms, such as Tabu search [6], simulated annealing [6], and genetic algorithm (GA) [6]–[8], were proposed. Unlike the gradient-descent-based algorithm, these search algorithms are less likely to be trapped in a local optimum and do not need a differentiable or even continuous error function. Thus, these search algorithms are more suitable for searching in a large, complex, nondifferentiable, and multimodal domain [9].

It is well known that a neural network can approximate any smooth and continuous nonlinear functions in a compact domain to an arbitrary accuracy [1], [2]. A three-layer feedforward neural network can successfully be applied in a wide range of applications, such as system modeling and control, prediction [10], [11], recognition [12], and so on. Owning to its specific structure, a neural network can be used to realize a learning process [1], [3], [4], [6]. Learning of the network usually consists of two steps: network structure design and learning process. The structure of the neural network defines the nonlinearity of the nonlinear equation. The learning algorithm is used to provide rules to optimize the connection weights. A typical structure of the neural network has a fixed set of connection weights after the learning process. However, a fixed set of connection weights may not be suitable to learn the information which is distributed in a vast domain separately.

In this paper, modifications have been made to the neural networks such that the parameters of the activation functions in the hidden layer are changed according to the network inputs. To achieve this, node-to-node links are introduced in the hidden layer. The node-to-node links interconnect the hidden nodes with connection weights. The modified neural network is shown in Fig. 1. Conceptually, the introduction of the node-to-node links increases the degree of freedom of the network. The modified neural network in result has a better learning and generalization abilities. The enhancements are due to the reason that the parameters in the activation functions of the hidden nodes

Manuscript received December 16, 2003; revised March 19, 2004. This work was supported by a Research Grant of the Center for Multimedia Signal Processing, The Hong Kong Polytechnic University (project number A432). This paper was recommended by Associate Editor H. Qiao.

The authors are with Centre of Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong.

Digital Object Identifier 10.1109/TSMCB.2004.834432

are allowed to change to cope with the changes of the network inputs in different operating subdomains. As a result, the modified neural network seems to have an individual neural network to handle the inputs of each operating subdomain. This characteristic is good for the modified neural network to handle problems with a large input data set in a large spatial domain.

A digit-and-command interpreter formed by the modified neural networks is proposed for recognizing handwritten digits and commands in this paper. The proposed digit-and-command interpreter will be realized practically in a prototype of the electronic book. Different methodologies for recognizing handwritten characters can be found in the literature. Generally, four different approaches [13] are used: template matching, statistical techniques, structural techniques, and neural/neural-fuzzy networks. The idea of the template-matching approach is to determine the best match between the stored templates and the inputs. Deformable models [14], [15], which are good in handling large shape variations, have been applied to the recognition of handwritten digits. During the recognition process, the patterns or shapes will be deformed to match with the input patterns. A classifier will then be employed to interpret the input patterns based on the extracted information. In [14], the deformable model was employed to represent the images in terms of contours. Dissimilarity measures will be obtained between characters and be used for classification. In [15], the deformable models were integrated into a Bayesian framework for recognition of handwritten characters. For statistical techniques, statistical decision theory is employed to determine the class to which the input belongs to. Hidden Markov modeling [16] is one of the popular statistical techniques for handwritten character recognition. On applying structural techniques, some complex patterns are represented by some simpler patterns. Based on these simpler patterns, the input can be classified. Examples of structural techniques include grammatical [17] and graphical [18] methods. In general, the neural/neural-fuzzy network approaches are model-free. The main idea of the neural/neural-fuzzy network approaches [19] is to learn the features of the training patterns in an off-line manner. The features can then be recognized using the trained neural/neural-fuzzy network. In [20], the orthogonality and information measures were employed to evaluate the features of the characters. These two measures will be taken as the inputs of the multilayer feedforward neural networks for classification. A self-organizing map approach can also be found in [21], of which the self-organizing map model were used to implement a modular classification system. An improved neocognitron approach, which is good in dealing with two-dimensional pattern recognition problems, was proposed in [22] for digit classification. However, the computational demand for the approach is high due to the complex structure of the neocognitron. A combined neural network architecture [23], which consists of two neural networks connected in cascade, can also be found to handle the recognition problems. The first neural network is for feature extraction, while the second one acts as a classifier. Furthermore, different kinds of neural and statistical classifiers were reported in [24].

A recognition system, the digit-and-command interpreter, is proposed and implemented practically in the prototype of electronic book in this paper. The digit-and-command interpreter, which consists of some modified neural networks connected in hierarchical structure, is able to recognize digits 0–9 and three (control) characters, namely *Backspace*, *Carriage Return*, and *Space*. The hierarchical structure provides expandability and flexibility to the digit-and-command interpreter. Extra graffiti can be added to the interpreter by adding an extra module without being to retrain all the networks. These properties fit for the development of different versions of electronic books required different recognition abilities. An improved GA [25] will be employed to train the parameters of the modified neural networks.

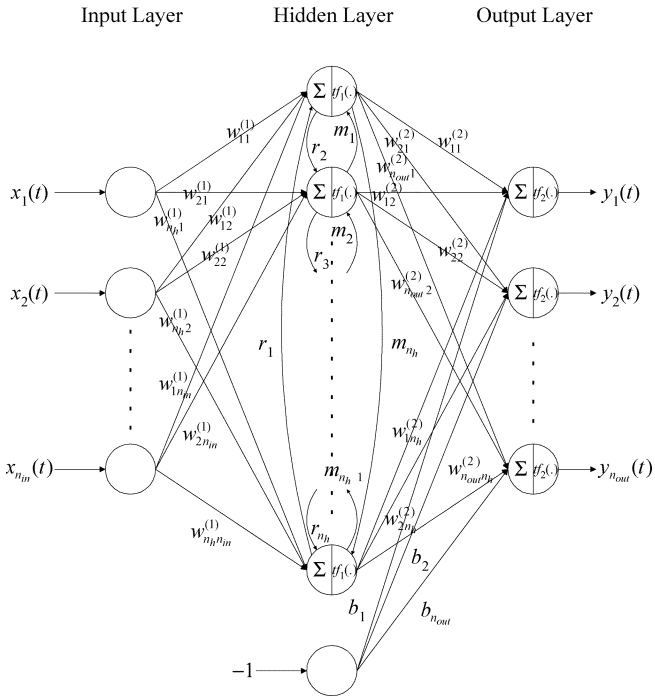

Fig. 1. A modified three-layer fully connected feedforward neural network with a node-to-node relationship.

This paper is organized as follows. The modified neural network will be presented in Section II. A digit-and-command interpreter, which is formed by the modified neural network, is proposed in Section III. The training of the parameters of the modified neural networks using an improved GA [25] will also be presented. Simulation and experimental results on interpreting handwritten digits and commands for an electronic book will be given in Section IV. A Conclusion will be drawn in Section V.

II. Modified Neural Network

A modified three-layer fully connected feedforward neural network with node-to-node relationship shown in Fig. 1 is proposed. The node-to-node connections are introduced between neighbors (the upper and the lower nodes) in the hidden node layers. It should be noted that the upper node of the first node is the last node. An inter-link is connected to the first node from the last node. Similarly, the lower node of the last node is the first node. An inter-link is connected to the last node from the first node. As a result, the total number of interconnection links

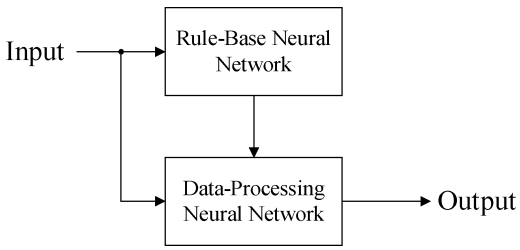

Fig. 2. Proposed architecture of the neural network.

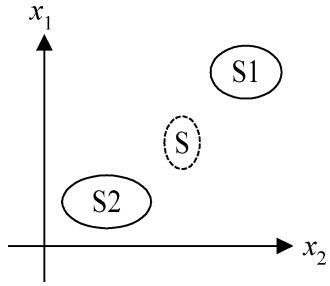

Fig. 3. Diagram showing two sets in the spatial domain.

is $2n_h$. The node-to-node relationship enhances the degree of freedom of the neural network by adapting the changes of the inputs. Consequently, the learning and the generalization abilities of the modified neural network can be enhanced.

The proposed neural network can be regarded as consisting of two units, namely the rule-base (RB) and the data -processing (DP) neural networks, as shown in Fig. 2. The RB neural network (the node-to-node part of the proposed neural network) stores some rules governing how the DP neural network (the rest part of the proposed neural-network) handles the input data. By using the modified neural network, some cases that a traditional neural network with limited number of parameters cannot provide a good performance may be solved. Fig. 3 shows an example to demonstrate the inadequacy of a traditional neural network to these cases. In this figure, S1 and S2 are the two sets of data in a spatial domain. By solving a mapping problem with a traditional neural network for instance, the weights of the neural network are trained to minimize the error between the network outputs and the desired values. However, the two data sets are separated far enough for a single neural network to model. As a result, the neural network is only able to produce S (average of S1 and S2), shown in Fig. 3. This problem might be tackled if the neural network employs a larger number of network parameters. In order to improve the learning and generalization abilities of the neural network, the architecture shown in Fig. 2 is proposed. Referring to Fig. 3, when the input data belongs to S1, the RB neural network will provide the rule (network parameters corresponding to S1) for 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 by the DP neural network to handle the input data. In other words, it operates like two individual neural networks handling their corresponding input data. Consequently, the proposed neural network is suitable to handle a large number of data. In this paper, the proposed neural network will be employed to handle the problem of handwritten-graffiti-and-command interpretation which involves lots of data.

Referring to Fig. 1, $\mathbf{x}(t) = [x_1(t)\, x_2(t)\, \cdots\, x_{n_{\text{in}}}(t)]$ denotes the input vector; n_{in} denotes the number of input nodes; t denotes the current number of input vectors, which is a nonzero integer; $w_{ij}^{(1)}, i = 1,2,\ldots,n_h, j = 1,2,\ldots,n_{\text{in}}$, denotes the connection weights between the jth node of the input layer and the ith node of the hidden layer; n_h denotes the number of hidden nodes; $w_{ki}^{(2)}$, $k = 1,2,\ldots,n_{\text{out}}$, $i = 1,2,\ldots,n_h$, denotes the connection weights between the ith node of the hidden layer and the kth node of the output layer; n_{out} denotes the number of output nodes. m_i and r_i are the connection weights of the links between hidden nodes (there are $2n_h$ inter-node links); b_k denotes the bias of the output nodes; $tf_1(\cdot)$ and $tf_2(\cdot)$ denote the activation functions of the hidden and output nodes, respectively. $\mathbf{y}(t) = [y_1(t)\, y_2(t)\, \cdots\, y_{n_{\text{out}}}(t)]$ denotes the output vector. The input–output relationship is governed by the following equation:

$$y_k(t) = tf_2\left(\sum_{i=1}^{n_h} w_{ki}^{(2)} f_{s_i}(\mathbf{x}(t)) - b_k\right), \quad k = 1,2,\ldots,n_{\text{out}} \tag{1}$$

Referring to Fig. 1

$$f_{s_i}(\mathbf{x}(t)) = tf_1\left(\sum_{j=1}^{n_{\text{in}}} w_{ij}^{(1)} x_j(t), m_i \sum_{j=1}^{n_{\text{in}}} w_{i+1j}^{(1)} x_j(t), r_i \sum_{j=1}^{n_{\text{in}}} w_{i-1j}^{(1)} x_j(t)\right), \quad i = 1,2,\ldots,n_h \tag{2}$$

which denotes the output of the ith hidden node

$$w_{i+1j}^{(1)} = \begin{cases} w_{1_j}^{(1)}, & \text{for } i = n_h \\ w_{i+1j}^{(1)}, & \text{otherwise} \end{cases} \tag{3}$$

$$w_{i-1j}^{(1)} = \begin{cases} w_{n_h j}^{(1)}, & \text{for } i = 1 \\ w_{i-1j}^{(1)}, & \text{otherwise} \end{cases} \tag{4}$$

$$tf_1\left(\sum_{j=1}^{n_{\text{in}}} w_{ij}^{(1)} x_j(t), m_i \sum_{j=1}^{n_{\text{in}}} w_{i+1j}^{(1)} x_j(t), r_i \sum_{j=1}^{n_{\text{in}}} w_{i-1j}^{(1)} x_j(t)\right) = \frac{2}{1+e^{-\frac{\left(\sum_{j=1}^{n_{\text{in}}} w_{ij}^{(1)} x_j(t) - m_i \sum_{j=1}^{n_{\text{in}}} w_{i+1j}^{(1)} x_j(t)\right)}{2\left(r_i \sum_{j=1}^{n_{\text{in}}} w_{i-1j}^{(1)} x_j(t)\right)^2}}} - 1 \in [-1\ 1]. \tag{5}$$

$tf_2(\cdot)$ can be chosen to be any commonly used activation functions such as pure linear, hyperbolic tangent sigmoid, logarithmic sigmoid activation functions [1], [3], [4]. It can be seen from (5) that the proposed activation function characterized by the varying mean $(m_i \sum_{j=1}^{n_{\text{in}}} w_{i+1j}^{(1)} x_j(t))$ and the varying standard deviation $(r_i \sum_{j=1}^{n_{\text{in}}} w_{i-1j}^{(1)} x_j(t))$, respectively. The values of $m_i \sum_{j=1}^{n_{\text{in}}} w_{i+1j}^{(1)} x_j(t)$ (which is functionally equivalent to the bias of the traditional neural network) and $r_i \sum_{j=1}^{n_{\text{in}}} w_{i-1j}^{(1)} x_j(t)$ govern the zero-crossing point and the steepness of the activation function, respectively. These values

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

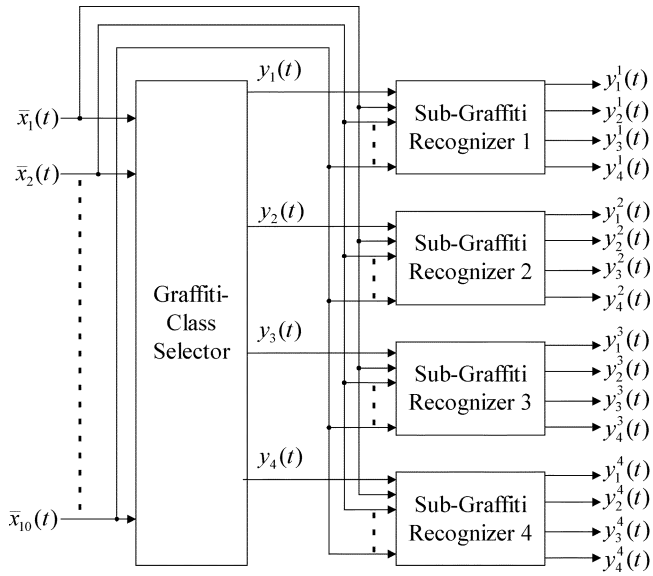

Fig. 5. Architecture of the digit-and-command interpreter.

will be changed according to the changes of the inputs of the activation function, i.e., $\sum_{j=1}^{n_{\text{in}}} w_{ij}^{(1)} x_j(t)$. In the modified neural network, the values of the parameters $w_{ij}^{(1)}$, $w_{ki}^{(2)}$, m_i, r_i and b_k will be learnt by the improved GA [25]. After training, the values of these parameters will be fixed during the operation. The total number of the tunable parameters of the proposed neural network is $(n_{\text{in}} + n_{\text{out}})n_h + n_{\text{out}} + 2n_h$.

III. Digit-and-Command Interpreter and Its Training

In this section, a digit-and-command interpreter is proposed to recognize the handwritten graffiti. The digits 0–9 and three commands (control characters: *backspace*, *carriage return* and *space*) are recognized by the proposed interpreter which is constructed by the modified neural network. The digits and commands are shown in Fig. 4. 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 $\phi_{\max}$ by $\beta_{\max}$. The bottom left corner is set as (0, 0). Ten uniformly sampled points of the graffiti will be taken as the inputs of the interpreter. The points are taken in the following way. First, the input graffiti is divided into nine uniformly distanced segments characterized by 10 points, including the start and the end points. Each point is labeled as (ϕ_i, β_i), $i = 1, 2, \ldots, 10$. The first five points, (ϕ_i, β_i), i =1, 3, 5, 7, and 9, taken alternatively are converted to five numbers ρ_i, respectively, by using the formula $\rho_i = \phi_i \phi_{\max} + \beta_i$. The other 5 points, (ϕ_i, β_i), i =2, 4, 6, 8, and 10, are converted to five numbers, respectively, by using the formula $\rho_i = \beta_i \beta_{\max} + \phi_i$. These ten numbers, ρ_i, $i = 1, 2, \ldots, 10$, will be used as the inputs of the proposed digit-and-command interpreter. The digit-and-command interpreter consisting of five modified neural networks as shown in Fig. 5 is proposed to perform the graffiti recognition. In this figure, the inputs are defined as follows:

$$\bar{\mathbf{x}}(t) = \frac{\mathbf{x}(t)}{||\mathbf{x}(t)||} \tag{6}$$

where $\overline{\mathbf{x}}(t) = [\overline{x}_1(t)\ \overline{x}_2(t)\ \cdots\ \overline{x}_{10}(t)]$ denotes the normalized input vectors of the proposed digit-and-command interpreter; $\mathbf{x}(t) = [x_1(t)\ x_2(t)\ \cdots\ x_{10}(t)] = [\rho_1(t)\ \rho_2(t)\ \cdots\ \rho_{10}(t)]$ denotes the ten points in the writing area; $||\cdot||$ denotes the l_2 vector norm. Referring to Fig. 5, the function of the graffiti-class selector is to divide the input graffiti classes into 4 subclasses. Referring to Fig. 4, the graffiti "0(a)," "0(b)," "1" and "2" are assigned to class 1; the graffiti "3," "4," "5(a)" and "5(b)" are assigned to class 2; the graffiti "6," "7," "8(a)" and "8(b)" are assigned to class 3; the graffiti "9," "backspace," "carriage return" and "space" are assigned to class 4. For better choice of the graffiti in each subclass, the characteristics of the graffiti should be considered. The graffiti with similar characteristics (e.g., the graffiti of 1, 4, and 7 look like each other) should avoid being placed in the same group.

To train the modified neural network which forms the graffiti-class selector, a set of training patterns governing the input-output relationship will be employed. 1600 training patterns (100 patterns for each graffiti) will be used. The training patterns consist of the input vectors and their corresponding expected outputs. The outputs are defined as that $y_i(t) = 1$ and others are zero when the input vector belongs to class i, i =1, 2, 3, 4. The parameters of modified neural networks will be tuned by the improved GA [25] in an offline manner. The details of the improved GA is given in the Appendix. After training, the connection weights of the neural networks will not changed during the operation. The parameter tuning process is formulated as a maximization problem with the fitness function defined as

$$\text{fitness} = \frac{1}{1 + \text{err}} \in [0\ 1] \tag{7}$$

where

$$\text{err} = \sum_{k=1}^{4} \frac{\sum_{t=1}^{1600} \left(\frac{y_k(t)}{||\mathbf{y}(t)||} - \frac{y_k^d(t)}{||\mathbf{y}^d(t)||} \right)^2}{4 \times 1600} \geq 0 \tag{8}$$

denotes the mean square error (mse); $\mathbf{y}^d(t) = [y_1^d(t)\ \ y_2^d(t)\ \ y_3^d(t)\ \ y_4^d(t)]$ denotes the expected output

vector, and $\mathbf{y}(t) = [y_1(t)\ y_2(t)\ y_3(t)\ y_4(t)]$ is the actual network output vector defined as

$$y_k(t)=tf_2\left(\sum_{i=1}^{n_h} w_{ki}^{(2)} tf_1\left(\sum_{j=1}^{10} w_{ij}^{(1)} x_j(t), m_i \sum_{j=1}^{10} w_{i+1j}^{(1)} x_l(t), r_i \sum_{j=1}^{10} w_{i-1j}^{(1)} x_j(t)\right) - b_k\right), \quad k=1,2,3,4 \quad (9)$$

The index of the maximum output of the graffiti-class selector indicates the possible subclass of the input vector belonging to. This output will be used to select the corresponding subclass recognizer to perform the second phase of recognition process. Hence, only one subclass recognizer activates at each time accordingly. Referring to Fig. 5, four subclass recognizers are employed to classify its input patterns. Each subclass recognizer implemented by the modified neural network has ten inputs and four outputs. It can be seen from this approach that the subclass recognizer limits the classes of the input patterns into four subclasses. Thus, the loading of learning process can be shared by the four subgraffiti recognizers which can be trained separately. Similar to the training process of the graffiti-class selector, a set of training patterns will be employed to train the subclass recognizer. 400 training patterns (100 for each graffiti) from those of the graffiti-class selector are employed to train each subclass recognizer. The outputs are defined as $y_i^{\alpha}(t) = 1$ and others are zero when the input vector belongs to class i, i =1, 2, 3, 4 where α =1, 2, 3, 4, numbers the subclass recognizer. The subclass recognizers will also be trained by the improved GA [25] with the fitness function given by (7) with

$$\text{err} = \sum_{k=1}^{4} \frac{\sum_{t=1}^{400} \left(\frac{y_k^{\alpha}(t)}{\|\mathbf{y}(t)\|} - \frac{y_k^{\alpha^d}(t)}{\|\mathbf{y}^d(t)\|} \right)^2}{4 \times 400}, \quad \alpha = 1,2,3,4 \quad (10)$$

where $\mathbf{y}_k^{\alpha^d}(t) = [y_1^{\alpha^d}(t)\ y_2^{\alpha^d}(t)\ y_3^{\alpha^d}(t)\ y_4^{\alpha^d}(t)]$ denotes the expected output vector and $\mathbf{y}_k^{\alpha}(t) = [y_1^{\alpha}(t)\ y_2^{\alpha}(t)\ y_3^{\alpha}(t)\ y_4^{\alpha}(t)]$ is the actual network output vector of the αth subclass recognizer. $\mathbf{y}_k^j(t)$ is the output of the modified neural network governed by (1). The final recognized class of the input pattern is indicated by the maximum outputs of the subclass recognizer activated by the graffiti-class selector. For instance, in the first recognition phase, the maximum output of the graffiti-class selector is from $y_1(t)$. Then, the first subclass recognizer will be employed to classify the input pattern. For instance, in the second recognition phase, $y_2^1(t)$ of the first subclass recognizer produces the maximum output. From these outputs, it can be concluded that the possible input pattern is "0(b)."

IV. Simulation and Experimental Results

The parameters of the modified neural networks will be tuned by the improved GA [25]. Different number of hidden nodes, i.e., n_h =10, 15, 20 and 25, for the modified neural networks will be employed. Hyperbolic tangent sigmoid function is employed as the output activation functions of the modified neural networks. For all training processes, the initial values of the chromosomes of the GA process which are formed by the parameters of the modified neural networks are randomly generated. The lower and upper bounds of the modified neural networks for all genes (network parameters) are -3 and 3, respectively, except that the lower bounds of all genes representing r_i are 0.001. The numbers of iteration to train the modified neural networks are 15 000 for the graffiti-class selector and 3000 for the subclasses recognizer, respectively. The control parameters of the improved GA [25], i.e., the probability of acceptance (p_a) and the weight of crossover (w) of the improved GA are 0.1 and 0.5, respectively. The population size for the GA process is 20. The training process is carried out in a personal computer with 1.4-GHz CPU and 512 MB RAM.

For comparison purposes, the digit-and-command interpreter constructed by the traditional three-layer fully connected feedforward neural network [1], [3], [4] will be trained by the improved GA. Logarithmic sigmoid and pure linear functions are employed as the activation functions of the hidden and output nodes, respectively. The training environment is the same as that of the modified neural networks except that the control parameters of the improve GA, i.e., $p_a = 0.1$ and $w = 0.7$, are different. In order to have similar number of tunable parameters, the number of hidden nodes (n_h) of the traditional neural networks is chosen to be 12, 18, 24, and 30. The GA training process for each neural network will be carried out 30 times. Tables I and II show the training and testing results of the modified and traditional neural networks. For each class of graffiti, 30 testing patterns are employed for the testing process. Hence, 480 $(4\ \text{classes} \times 4\ \text{graffiti per class} \times 30\ \text{patterns per graffiti})$ and 120 (4 graffiti $\times$ 30 patterns per graffiti) testing patterns are employed for the graffiti-class selector and subgraffiti recognizers, respectively. In these tables, it can be observed that the modified neural networks is superior than the traditional neural networks in terms of maximum fitness value (the largest fitness value among the 30 runs), mean and standard deviation of the fitness values, and the recognition rate (which is obtained by feeding the training patterns to each neural network individually). Furthermore, referring to Tables I and II, it can be seen that the mean and testing fitness values of the subgraffiti recognizersimplemented by the modified neural networks are slightly better than those of the subgraffiti recognizers implemented by the traditional three-layer fully connected feedforward neural network. However, when large numbers of training and testing data are used, the better learning and generalization abilities of the modified neural network can be demonstrated. Referring to Table I and Table II, the mean and the testing fitness values of graffiti-class selector implemented by the modified neural networks are better than those of the subgraffiti recognizers implemented by the traditional three-layer fully connected feedforward neural network. It can be shown that the modified neural network is good in handling large number of data. On average, the training times of each node for the proposed and traditional neural networks are 1.0816×10^{-3} and 0.9499×10^{-3} seconds, respectively, for each iteration.

Table III shows the overall recognition rate for the digit and command recognizer implemented by the trained modified

TABLE I
TRAINING RESULTS OF THE MODIFIED AND TRADITIONAL NEURAL NETWORKS

	n_h (number of parameter)	Maximum Fitness Value	Mean Fitness Value	Standard Deviation	Recognition Rate of the Best Neural Network (%) for the training patterns			
					Class 1	Class 2	Class 3	Class 4
Graffiti Class Selector	10 (164)	0.9787	0.9748	0.0123	97.5	98.75	98.25	96.25
	15 (244)	0.9816	0.9768	0.0043	99.5	98.75	97.5	97
	20 (324)	0.9882	0.9765	0.0016	99	99.25	98	98.5
	25 (404)	0.9898	0.9786	0.0018	99	99.5	99	99
Sub-Graffiti Recognizer 1	10 (164)	0.9985	0.9961	0.0020	100	100	100	100
	15 (244)	0.9988	0.9965	0.0018	100	100	100	100
	20 (324)	0.9982	0.9965	0.0013	100	100	100	100
	25 (404)	0.9983	0.9931	0.0179	100	100	100	100
Sub-Graffiti Recognizer 2	10 (164)	0.9959	0.9879	0.0137	99	100	100	100
	15 (244)	0.9958	0.9905	0.0042	99	100	100	100
	20 (324)	0.9957	0.9844	0.0272	100	100	100	100
	25 (404)	0.9951	0.9921	0.0019	99	100	100	100
Sub-Graffiti Recognizer 3	10 (164)	0.9977	0.9898	00078	100	100	100	100
	15 (244)	0.9982	0.9923	0.0027	100	100	100	100
	20 (324)	0.9956	0.9921	0.0026	100	100	100	100
	25 (404)	0.9969	0.9888	0.0194	100	100	100	100
Sub-Graffiti Recognizer 4	10 (164)	0.9992	0.9978	0.0010	100	100	100	100
	15 (244)	0.9991	0.9956	0.0128	100	100	100	100
	20 (324)	0.9992	0.9965	0.0065	100	100	100	100
	25 (404)	0.9991	0.9978	0.0007	100	100	100	100

(a) Modified Neural Network.

	n_h (number of parameter)	Maximum Fitness Value	Mean Fitness Value	Standard Deviation	Recognition Rate of the Best Neural Network (%) for the training patterns			
					Class 1	Class 2	Class 3	Class 4
Graffiti Class Selector	12 (184)	0.9489	0.9360	0.0074	90	98	75.5	93.5
	18 (274)	0.9568	0.9464	0.0062	91	97.5	94	96.25
	24 (364)	0.9654	0.9528	0.0079	93	99	96.5	96.5
	30 (454)	0.9666	0.9591	0.0088	93	98.5	96	96
Sub-Graffiti Recognizer 1	12 (184)	0.9914	0.9865	0.0102	100	100	99	100
	18 (274)	0.9920	0.9765	0.0295	99	100	100	100
	24 (364)	0.9924	0.9570	0.0531	99	100	100	100
	30 (454)	0.9925	0.9616	0.0381	99	100	99	100
Sub-Graffiti Recognizer 2	12 (184)	0.9801	0.9730	0.0054	99	100	100	99
	18 (274)	0.9812	0.9756	0.0046	99	100	100	100
	24 (364)	0.9823	0.9738	0.0144	99	100	100	99
	30 (454)	0.9817	0.9643	0.0314	99	100	100	99
Sub-Graffiti Recognizer 3	12 (184)	0.9845	0.9731	0.0054	100	100	100	100
	18 (274)	0.9856	0.9748	0.0071	100	100	100	99
	24 (364)	0.9843	0.9723	0.0135	100	100	100	99
	30 (454)	0.9868	0.9671	0.0252	100	100	100	100
Sub-Graffiti Recognizer 4	12 (184)	0.9967	0.9923	0.0142	100	100	100	100
	18 (274)	0.9968	0.9855	0.0237	100	100	100	100
	24 (364)	0.9971	0.9773	0.0368	100	100	100	100
	30 (454)	0.9969	0.9615	0.0551	100	100	100	100

(b) Traditional Neural Network.

and traditional neural networks. 1600 training and 480 testing patterns are fed to the digit and command recognizer for interpretation. Referring to Table III, it can be seen that the digit and command recognizer implemented by the modified neural networks performs better in terms of recognition rate. The proposed digit and command recognizer has been realized practically in a prototype of an electronic book which is shown in Fig. 6. Fig. 7 shows the input of digit "9" to the annotation window using the graffiti pad of the electronic book. The proposed approach will be compared with the interpreter constructed by the traditional neural networks, and the approaches in [15] and [20]–[23]. Table IV tabulates the best overall testing results of different approaches. It can be seen that the proposed approach provides are more or less the same performance as others.

The digit-and-command interpreter shown in Fig. 5 which is formed by the proposed neural networks connected in hierarchical structure. The graffiti-class selector is to perform the first-phase classification which divides the input graffiti into different subclasses. Once the subclass of the input graffiti is iden-

TABLE II
TESTING RESULTS OF THE BEST NEURAL NETWORKS FOR BOTH APPROACHES

	n_h (number of parameter)	Testing Fitness Value	Recognition Rate of the Best Neural Network (%) for the testing patterns			
			Class 1	Class 2	Class 3	Class 4
Graffiti Class Selector	10 (164)	0.9695	98.3333	97.5	91.6667	99.1667
	15 (244)	0.9722	100	96.6667	91.6667	98.3333
	20 (324)	0.9768	100	98.3333	93.33335	98.3333
	25 (404)	0.9793	100	98.3333	96.6667	100
Sub-Graffiti Recognizer 1	10 (164)	0.7285	100	100	100	100
	15 (244)	0.9981	100	100	100	100
	20 (324)	0.9979	100	100	100	100
	25 (404)	0.9983	100	100	100	100
Sub-Graffiti Recognizer 2	10 (164)	0.7258	100	100	100	100
	15 (244)	0.9854	93.3333	100	100	96.6667
	20 (324)	0.9913	100	100	100	100
	25 (404)	0.9925	100	100	100	100
Sub-Graffiti Recognizer 3	10 (164)	0.7391	96.6667	100	96.6667	86.6667
	15 (244)	0.9867	93.3333	100	96.6667	100
	20 (324)	0.9719	100	100	93.3333	96.6667
	25 (404)	0.9775	100	100	96.6667	96.6667
Sub-Graffiti Recognizer 4	10 (164)	0.7327	100	96.6667	100	100
	15 (244)	0.9924	96.6667	96.6667	100	100
	20 (324)	0.9942	100	96.6667	100	100
	25 (404)	0.9940	100	96.6667	100	100

(a) Modified Neural Network.

	n_h (number of parameter)	Testing Fitness Value	Recognition Rate of the Best Neural Network (%) for the testing patterns			
			Class 1	Class 2	Class 3	Class 4
Graffiti Class Selector	12 (184)	0.9483	94.1667	97.5	76.6667	98.3333
	18 (274)	0.9509	94.1667	98.3333	90.8333	98.3333
	24 (364)	0.9598	96.6667	98.3333	91.6667	98.3333
	30 (454)	0.9612	98.3333	98.3333	96.6667	99.1667
Sub-Graffiti Recognizer 1	12 (184)	0.7343	100	100	100	100
	18 (274)	0.9920	100	100	100	100
	24 (364)	0.9930	100	100	100	100
	30 (454)	0.9933	100	100	100	100
Sub-Graffiti Recognizer 2	12 (184)	0.7219	100	100	100	100
	18 (274)	0.9729	96.6667	100	100	100
	24 (364)	0.9793	96.6667	100	100	100
	30 (454)	0.9782	96.6667	100	100	100
Sub-Graffiti Recognizer 3	12 (184)	0.7162	100	100	93.3333	90
	18 (274)	0.9513	100	100	93.3333	80
	24 (364)	0.9509	100	100	96.6667	90
	30 (454)	0.9483	100	100	96.6667	96.6667
Sub-Graffiti Recognizer 4	12 (184)	0.7386	100	96.6667	100	100
	18 (274)	0.9919	100	96.6667	100	100
	24 (364)	0.9925	100	96.6667	100	100
	30 (454)	0.9921	100	96.6667	100	100

(b) Traditional Neural Network.

tified, the corresponding subgraffiti recognizer will be employed to carry out the second-phase recognition process to identify the input graffiti. If extra graffiti are added to the digit-and-command interpreter, the graffiti-class selector is needed to be re-trained to include the information of the extra graffiti. Extra subgraffiti recognizers will be added to recognize the extra graffiti. However, all the existing subgraffiti recognizers are not necessary to be retained. It can be seen that the hierarchical structure provides expandability and flexibility to the digit-and-command interpreter. These properties fit for the development of different versions of electronic books required different recognition abilities. Furthermore, for adding large number of graffiti, the two-level hierarchical structure of the interpreter can be extended to a multiple-level one.

V. CONCLUSION

A modified neural network has been presented in this paper. The modified neural network has been employed to implement a digit-and-command interpreter for recognizing handwritten digits and commands. The proposed digit-and-command interpreter has been successfully implemented in an electronic book practically. Simulation and experimental results have been given.

TABLE III
OVERALL RECOGNITION RATE OF THE DIGIT-AND-COMMAND INTERPRETER IMPLEMENTED BY THE MODIFIED AND TRADITIONAL NEURAL NETWORKS

	Modified Neural Network		Traditional Neural Network	
	Recognition rate (%) of the best network for the training patterns	Recognition rate (%) of the best network for the testing patterns	Recognition rate (%) of the best network for the training patterns	Recognition rate (%) of the best network for the testing patterns
0(a)	100	100	99	96.6667
0(b)	100	100	100	100
1	98	100	74	96.6667
2	98	100	99	93.3333
3	99	96.6667	99	96.6667
4	99	96.667	99	96.6667
5(a)	100	100	100	96.6667
5(b)	99	100	96	100
6	100	100	100	90
7	94	90	90	86.6667
8(a)	98	90	96	90
8(b)	100	83.3333	99	90
9	96	93.3333	86	93.3333
Back Space	98	96.6667	100	96.6667
Return	100	100	100	100
Space	100	100	100	100

(c). $n_h = 20$.

	Modified Neural Network		Traditional Neural Network	
	Recognition rate (%) of the best network for the training patterns	Recognition rate (%) of the best network for the testing patterns	Recognition rate (%) of the best network for the training patterns	Recognition rate (%) of the best network for the testing patterns
0(a)	98	100	98	100
0(b)	99	100	100	100
1	100	100	76	93.3333
2	100	100	98	100
3	98	96.6667	99	96.6667
4	100	100	98	96.6667
5(a)	100	96.6667	99	96.6667
5(b)	99	100	96	100
6	99	100	99	100
7	98	93.3333	90	86.6667
8(a)	98	96.6667	95	96.6667
8(b)	100	96.6667	100	96.6667
9	97	100	84	96.6667
Back Space	99	96.6667	100	96.6667
Return	100	100	100	100
Space	100	100	100	100

(d). $n_h = 25$.

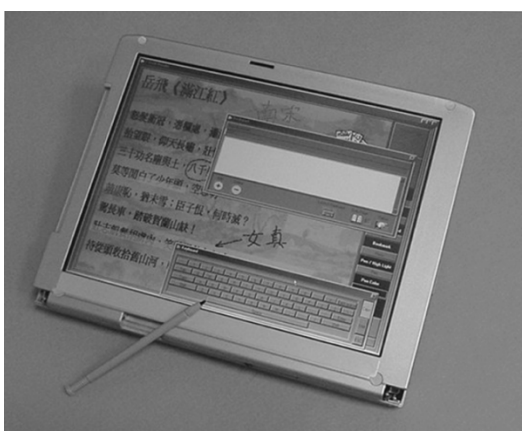

Fig. 6. Prototype of an electronic book.

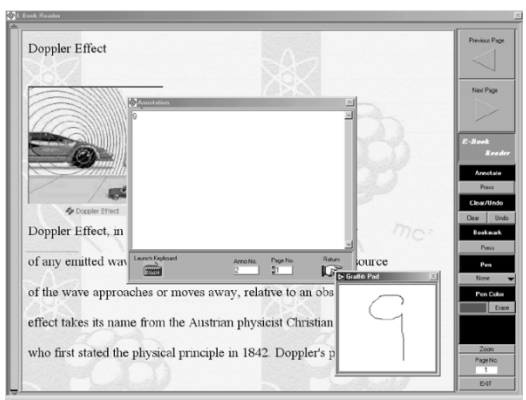

Fig. 7. Input of digit "9" to the annotation window using the graffiti pad of the electronic book.

TABLE IV
BEST TESTING RESULTS OF DIFFERENT APPROACHES

Approach	Overall Testing Results (%)
Digit-and-command interpreter implemented by the modified neural networks	98.54
Digit-and-command interpreter implemented by the traditional neural networks	97.29
[15]	94.7
[20]	98
[21]	97.87
[22]	96.43
[23]	90.8

```
Procedure of the improved GA
begin
        τ → 0    // τ: number of iteration
        initialize P(τ)                  //P(τ): population for iteration τ
        evaluate f(P(τ))   //f(P(τ)):fitness function
while (not termination condition) do
        begin
        τ → τ+1
        select 2 parents p1 and p2 from P(τ − 1)
        perform crossover operation according to equations (A7) to (A13)
          perform mutation operation according to equation (A14) to generate three
          offspring nos1, nos2 and nos3
        // reproduce a new P(τ)
              if random number < pa   // pa: probability of acceptance
                  The one among nos1, nos2 and nos3 with the largest fitness value
                  replaces the chromosome with the smallest fitness value in the
                  population
                else begin
                  if f(nos1) > smallest fitness value in the P(τ − 1)
                      nos1 replaces the chromosome with the smallest fitness value
                  end
                  if f(nos2) > smallest fitness value in the updated P(τ − 1)
                      nos2 replaces the chromosome with the smallest fitness value
                  end
                  if f(nos3) > smallest fitness value in the updated P(τ − 1)
                      nos3 replaces the chromosome with the smallest fitness value
                  end
                end
                evaluate f(P(τ))
        end
end
```

Fig. 8. Procedure of the improved GA.

APPENDIX

The details of the improved GA [25] will be given in this Appendix. The improved GA process is shown in Fig. 8. First, a population of chromosomes is created. Second, the chromosomes are evaluated by a defined fitness function. Third, some of the chromosomes are selected for performing genetic operations. Forth, genetic operations of crossover and mutation are performed. The reproduced offspring replace their parents in the initial population. The details of the improved GA are presented as follows.

A. Initial Population

The initial population is a potential solution set P. The first set of the population is usually generated randomly

$$P = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{\text{pop\_size}}\} \tag{A1}$$

$$\mathbf{p}_i = \begin{bmatrix} p_{i_1} & p_{i_2} & \cdots & p_{i_j} & \cdots & p_{i_{\text{no\_vars}}} \end{bmatrix} \quad i = 1, 2, \ldots, \text{pop\_size} \quad j = 1, 2, \ldots, \text{no\_vars} \tag{A2}$$

$$\text{para}^j_{\min} \le p_{i_j} \le \text{para}^j_{\max} \tag{A3}$$

where pop_size denotes the population size; no_vars denotes the number of variables to be tuned; p_{i_j}, $i = 1, 2, \ldots, \text{pop\_size}$; $j = 1, 2, \ldots, \text{no\_vars}$, are the parameters to be tuned; $\text{para}^j_{\min}$ and $\text{para}^j_{\max}$ are the minimum and maximum values of the parameter p_{i_j}, respectively, for all i. It can be seen from (A1) to (A3) that the potential solution set P contains some candidate solutions $\mathbf{p}_i$ (chromosomes). The chromosome $\mathbf{p}_i$ contains some variables p_{i_j} (genes).

B. Evaluation

Each chromosome in the population will be evaluated by a defined fitness function. The better chromosomes will return higher values in this process. The fitness function to evaluate a chromosome in the population can be written as

$$\text{fitness} = f(\mathbf{p}_i). \tag{A4}$$

The form of the fitness function depends on the application.

C. Selection

Two chromosomes in the population will be selected to undergo genetic operations for reproduction by the method of spinning the roulette wheel [6]–[8]. It is believed that high potential parents will produce better offspring (survival of the best ones). The chromosome having a higher fitness value should therefore have a higher chance to be selected. The selection can be done by assigning a probability q_i to the chromosome $\mathbf{p}_i$ as follows:

$$q_i = \frac{f(\mathbf{p}_i)}{\sum_{k=1}^{\text{pop\_size}} f(\mathbf{p}_k)}, \quad i = 1, 2, \ldots, \text{pop\_size}. \tag{A5}$$

The cumulative probability $\hat{q}_i$ for the chromosome $\mathbf{p}_i$ is defined as

$$\hat{q}_i = \sum_{k=1}^{i} q_k, \quad i = 1, 2, \ldots, \text{pop\_size}. \tag{A6}$$

The selection process starts by randomly generating a nonzero floating-point number $d \in [0\ 1]$. Then, the chromosome $\mathbf{p}_i$ is chosen if $\hat{q}_{i-1} < d \le \hat{q}_i$ ($\hat{q}_0 = 0$). It can be observed from this selection process that a chromosome having a larger $f(\mathbf{p}_i)$ will have a higher chance to be selected. Consequently, the best chromosomes will get more offspring, the average will stay and the worst will die off. In the selection process, only two chromosomes will be selected to undergo the genetic operations.

D. Genetic Operations

The genetic operations are to generate some new chromosomes (offspring) from their parents after the selection process. They include the crossover and the mutation operations.

1) Crossover: The crossover operation is mainly for exchanging information from the two parents, chromosomes $\mathbf{p}_1$ and $\mathbf{p}_2$, obtained in the selection process. The two parents will produce one offspring. First, four chromosomes will be generated according to the following mechanisms:

$$\mathbf{os}^1_c = \begin{bmatrix} \text{os}^1_1 & \text{os}^1_2 & \cdots & \text{os}^1_{\text{no\_vars}} \end{bmatrix} = \frac{\mathbf{p}_1 + \mathbf{p}_2}{2} \tag{A7}$$

$$\mathbf{os}_c^2 = \begin{bmatrix} \mathrm{os}_1^2 & \mathrm{os}_2^2 & \cdots & \mathrm{os}_{\mathrm{no\_vars}}^2 \end{bmatrix} = \mathbf{p}_{\max}(1-w) + \max(\mathbf{p}_1, \mathbf{p}_2)w \quad \text{(A8)}$$

$$\mathbf{os}_c^3 = \begin{bmatrix} \mathrm{os}_1^3 & \mathrm{os}_2^3 & \cdots & \mathrm{os}_{\mathrm{no\_vars}}^3 \end{bmatrix} = \mathbf{p}_{\min}(1-w) + \min(\mathbf{p}_1, \mathbf{p}_2)w \quad \text{(A9)}$$

$$\mathbf{os}_c^4 = \begin{bmatrix} \mathrm{os}_1^4 & \mathrm{os}_2^4 & \cdots & \mathrm{os}_{\mathrm{no\_vars}}^4 \end{bmatrix} = \frac{(\mathbf{p}_{\max} + \mathbf{p}_{\min})(1-w) + (\mathbf{p}_1 + \mathbf{p}_2)w}{2} \quad \text{(A10)}$$

$$\mathbf{p}_{\max} = \begin{bmatrix} \mathrm{para}_{\max}^1 & \mathrm{para}_{\max}^2 & \cdots & \mathrm{para}_{\max}^{\mathrm{no\_vars}} \end{bmatrix} \quad \text{(A11)}$$

$$\mathbf{p}_{\min} = \begin{bmatrix} \mathrm{para}_{\min}^1 & \mathrm{para}_{\min}^2 & \cdots & \mathrm{para}_{\min}^{\mathrm{no\_vars}} \end{bmatrix} \quad \text{(A12)}$$

where $w \in [0\ 1]$ denotes the weight to be determined by users, $\max(\mathbf{p}_1, \mathbf{p}_2)$ denotes the vector with each element obtained by taking the maximum among the corresponding element of $\mathbf{p}_1$ and $\mathbf{p}_2$. For instance, $\max([1\ \ -2\ \ 3], [2\ \ 3\ \ 1]) = [2\ \ 3\ \ 3]$. Similarly, $\min(\mathbf{p}_1, \mathbf{p}_2)$ gives a vector by taking the minimum value. For instance, $\min([1\ -2\ 3], [2\ 3\ 1]) = [1\ -2\ 1]$. Among $\mathbf{os}_c^1$ to $\mathbf{os}_c^4$, the one with the largest fitness value is used as the offspring of the crossover operation. The offspring is defined as

$$\mathbf{os} \equiv \begin{bmatrix} \mathrm{os}_1 & \mathrm{os}_2 & \cdots & \mathrm{os}_{\mathrm{no\_vars}} \end{bmatrix} = \mathbf{os}_c^{i_{\mathrm{OS}}} \quad \text{(A13)}$$

where i_{OS} denotes the index i which gives a maximum value of $f(\mathbf{os}_c^i)$, i =1, 2, 3, 4.

If the crossover operation can provide a good offspring, a higher fitness value can be reached in less iteration. In general, two-point crossover, multipoint crossover, arithmetic crossover or heuristic crossover can be employed to realize the crossover operation [7], [26]–[28]. The offspring generated by these methods, however, may not be better than that from our approach. As seen from (A7) to (A10), the potential offspring spreads over the domain. While (A7) and (A10) result in searching around the centre region of the domain (a value of w near to 1 in (A10) can move $\mathbf{os}_c^4$ to be near $(\mathbf{p}_1 + \mathbf{p}_2)/2$), (A8) and (A9) move the potential offspring to be near the domain boundary (a large value of w in (A8) and (A9) can move $\mathbf{os}_c^2$ and $\mathbf{os}_c^3$ to be near $\mathbf{p}_{\max}$ and $\mathbf{p}_{\min}$, respectively).

2) Mutation: The offspring (A13) will then undergo the mutation operation. The mutation operation is to change the genes of the chromosomes. Consequently, the features of the chromosomes inherited from their parents can be changed. Three new offspring will be generated by the mutation operation

$$\mathbf{nos}_j = \begin{bmatrix} \mathrm{os}_1 & \mathrm{os}_2 & \cdots & \mathrm{os}_{\mathrm{no\_vars}} \end{bmatrix} + \begin{bmatrix} b_1 \Delta \mathrm{nos}_1 & b_2 \Delta \mathrm{nos}_2 & \cdots & b_{\mathrm{no\_vars}} \Delta \mathrm{nos}_{\mathrm{no\_vars}} \end{bmatrix}, \quad j = 1, 2, 3 \quad \text{(A14)}$$

where $b_i, i = 1, 2, \ldots,$ no_vars can only take the value of 0 or 1, $\Delta nos_i, i = 1, 2, \ldots,$ no_vars, are randomly generated numbers such that $\mathrm{para}_{\min}^i \leq \mathrm{os}_i + \Delta \mathrm{nos}_i \leq \mathrm{para}_{\max}^i$. The first new offspring $(j = 1)$ is obtained according to (A14) with that only one b_i (i being randomly generated within the range) is allowed to be 1 and all the others are zeros. The second new offspring is obtained according to (A14) with that some b_i randomly chosen are set to be 1 and others are zero. The third new offspring is obtained according to (A14) with all $b_i = 1$. These three new offspring will then be evaluated using the fitness function of (A4). A real number will be generated randomly and compared with a user-defined number $p_a \in [0\ 1]$. If the real number is smaller than p_a, the one with the largest fitness value among the three new offspring will replace the chromosome with the smallest fitness f_s in the population. If the real number is larger than p_a, the first offspring $\mathbf{nos}_1$ will replace the chromosome with the smallest fitness value f_s in the population if $f(\mathbf{nos}_1) > f_s$; the second and the third offspring will do the same. p_a is effectively the probability of accepting a bad offspring in order to reduce the chance of converging to a local optimum. Hence, the possibility of reaching the global optimum is kept.

In general, various methods like boundary mutation, uniform mutation or nonuniform mutation [7], [27], [28] can be employed to realize the mutation operation. Boundary mutation is to change the value of a randomly selected gene to its upper or lower bound. Uniform mutation is to change the value of a randomly selected gene to a value between its upper and lower bounds. Non-uniform mutation is capable of fine-tuning the parameters by increasing or decreasing the value of a randomly selected gene by a weighted random number. The weight is usually a monotonic decreasing function of the number of iteration. In our approach, we have three offspring generated in the mutation process. From (A14), the first mutation is in fact the uniform mutation. The second mutation allows some randomly selected genes to change simultaneously. The third mutation changes all genes simultaneously. The second and the third mutations allow multiple genes to be changed. Hence, the searching domain is larger than that formed by changing a single gene. The genes will have a larger space for improving when the fitness values are small. On the contrary, when the fitness values are nearly the same, changing the value of a single gene (the first mutation) will give a higher probability of improving the fitness value as the searching domain is smaller and some genes may have reached their optimal values.

After the operation of selection, crossover, and mutation, a new population is generated. This new population will repeat the same process. Such an iterative process can be terminated when the result reaches a defined condition, e.g., the change of the fitness values between the current and the previous iteration is less than 0.001, or a defined number of iteration has been reached.

REFERENCES

[1] F. M. Ham and I. Kostanic, *Principles of Neurocomputing for Science & Engineering*. New York: McGraw-Hill, 2001.

[2] B. Widrow and M. A. Lehr, "30 years of adaptive neural networks: perceptron, madaline, and backpropagation," *Proc. IEEE*, vol. 78, no. 9, pp. 1415–1442, Sept. 1990.

[3] J. M. Zurada, *Introduction to Artificial Neural Systems*. St. Paul: West Info Access, 1992.

[4] S. Haykin, *Neural Network: A Comprehensive Foundation*. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[5] M. F. Moller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks*, vol. 6, no. 4, pp. 525–533, 1993.

[6] D. T. Pham and D. Karaboga, *Intelligent Optimization Techniques, Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks*. New York: Springer, 2000.

[7] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.

[8] Z. Michalewicz, *Genetic Algorithm + Data Structures = Evolution Programs*, 2nd Michalewicz, Ed. New York: Springer-Verlag, 1994.

[9] X. Yao, "Evolving artificial neural networks," *Proc. IEEE*, vol. 87, pp. 1423–1447, Sept. 1999.

[10] M. Li, K. Mechrotra, C. Mohan, and S. Ranka, "Sunspot numbers forecasting using neural network," in *Proc. 5th IEEE Int. Symp. Intelligent Control*, 1990, pp. 524–528.
[11] 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. Ind. Electron.*, vol. 50, pp. 793–799, Aug. 2003.
[12] 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. Ind. Electron*., to be published.
[13] N. Arica and F. T. Yarman-Vural, "An overview of character recognition focused on off-line handwriting," *IEEE Trans. Syst., Man, Cybern. C*, vol. 31, pp. 216–233, May 2001.
[14] A. K. Jain and D. Zongker, "Representation and recognition of handwritten digits using deformable templates," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 1386–1391, Dec. 1997.
[15] K. W. Cheung, D. Y. Yeung, and R. T. Chin, "A Bayesian framework for deformable pattern recognition with application to handwritten character recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 1382–1388, Dec. 1998.
[16] M. A. Mohamed and P. Gader, "Generalized hidden Markov models—part II: application to handwritten word recognition," *IEEE Trans. Fuzzy Syst.*, vol. 8, pp. 82–95, Feb. 2000.
[17] M. Shridhar and F. Kimura, "High accuracy syntactic recognition algorithm for handwritten numerals," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, pp. 152–158, Jan. 1985.
[18] H. Y. Kim and J. H. Kim, "Handwritten Korean character recognition based on hierarchical random graph modeling," in *Proc. Int. Workshop Frontiers Handwriting Recognition*, 1998, pp. 577–586.
[19] R. Buse, Z. Q. Liu, and J. Bezdek, "Word recognition using fuzzy logic," *IEEE Trans. Fuzzy Syst.*, vol. 10, pp. 65–76, Feb. 2002.
[20] P. D. Gader and M. A. Khabou, "Automatic feature generation for handwritten digit recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 1256–1261, Dec. 1996.
[21] B. Zhang, M. Fu, H. Yan, and M. A. Jabri, "Handwritten digit recognition by adaptive-subspace self-organizing map (ASSOM)," *IEEE Trans. Neural Networks*, vol. 10, pp. 939–945, July 1997.
[22] D. R. Lovell, T. Downs, and A. C. Tsoi, "An evaluation of the neocognitron," *IEEE Trans. Neural Networks*, vol. 8, pp. 1090–1105, Sept. 1997.
[23] C. A. Perez, C. A. Salinas, P. A. Estévez, and P. M. Valenzuela, "Genetic design of biologically inspired receptive fields for neural pattern recognition," *IEEE Trans. Syst., Man, Cybern. B*, vol. 33, pp. 258–270, Apr. 2003.
[24] L. Holmström, P. Koistinen, J. Laaksonen, and E. Oja, "Neural and statistical classifiers—taxonomy and two case studies," *IEEE Trans. Neural Networks*, vol. 8, pp. 5–17, Jan. 1997.
[25] 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, pp. 79–88, Jan. 2003.
[26] X. Wang and M. Elbuluk, "Neural network control of induction machines using genetic algorithm training," in *Conf. Record 31st IAS Ann. Meeting (IAS '96)*, vol. 3, 1996, pp. 1733–1740.
[27] L. Davis, *Handbook of Genetic Algorithms*. New York: Van Nostrand, 1991.
[28] M. Srinivas and L. M. Patnaik, "Genetic algorithms: a survey," *IEEE Computer*, vol. 27, pp. 17–26, June 1994.

H. K. Lam received the B.Eng. (Hons.) and Ph.D. degrees from the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Kowloon, in 1995 and 2000, respectively.

He is currently a post-doctoral Fellow in the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University. His research interests include intelligent control and systems and computational intelligence.

Frank H. F. Leung (M'92) was born in Hong Kong in 1964. He received the B.Eng. and Ph.D. degrees in electronic engineering from the Hong Kong Polytechnic University, Kowloon, in 1988 and 1992 respectively.

He joined the Hong Kong Polytechnic University in 1992 and is now an Associate Professor in the Department of Electronic and Information Engineering. He has published over 120 research papers on intelligent control and power electronics. At present, he is actively involved in the research on intelligent multimedia home, and eBooks. He is a Reviewer for many international journals and has had helped in the organization of many international conferences. He is a Chartered Engineer and a corporate member of the Institution of Electrical Engineers, U.K

Dr. Leung is an Executive Committee member of the IEEE Hong Kong Chapter of Signal Processing and the Joint Chapters of CAS/COM.