Neural Fuzzy Network and Genetic Algorithm Approach for Cantonese Speech Command Recognition[1]

K.F. Leung, F.H.F. Leung, H.K. Lam and P.K.S. Tam
Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

Abstract: This paper presents the recognition of Cantonese speech commands using a proposed neural fuzzy network with rule switches. By introducing a switch to each rule, the optimal number of rules can be learned. An improved genetic algorithm (GA) is proposed to train the parameters of the membership functions and the optimal rule set for the proposed neural fuzzy network. An application example of Cantonese command recognition in electronic books will be given to illustrate the merits of the proposed approach.

I. INTRODUCTION

Speech recognition involves two processes: feature extraction and recognition. Feature extraction is a preprocessing stage to extract the feature parameters of each spoken word. Since spoken words are sensitive to environmental noise and human factors, a distinguished feature extraction technique have to be employed to identify each word. The commonly used approaches for feature extraction are the filter bank modeling and linear predictive coding (LPC) analysis [1]. Filter bank filtering analysis is based on a model of human ears. The bandwidths of the band-pass filters are distributed in mel-scale and bark-scale, as human ears are more sensitive to low frequency components. LPC analysis aims to mimic the human vocal tract. It approximates the current sampled speech as a linear combination of its past samples. The time-domain speech signals are first windowed into frames, and the autocorrelation coefficients between frames are obtained.

Classification is the next stage of the speech recognition system. It is used to determine the class that the feature parameters belong to. Speech classification is usually done using a pattern recognition approach or a statistical analysis approach. Artificial neural network (ANN) and hidden Markov model (HMM) are used for pattern recognition and statistical analysis respectively. ANN is distinct in discrimination; thus, matching template patterns with testing patterns can be employed to do the classification. However, the number of mathematical operations required is large if the ANN has a large number of input parameters. Thus, ANN is suitable for the recognition of short speeches only. HMM is good at statistical modeling of continuous speech signals. The states in the HMM characterize the phonemes, and the speech is formulated into a sequence of states.

[1] The work described in this paper was fully supported by a grant from the Centre for Multimedia Signal Processing, The Hong Kong Polytechnic University (project number A432).

Cantonese command recognition is a challenging because Cantonese is a nine-tonal and syllabic language [2]. Some Cantonese words are difficult to discriminate owing to the similar tones. Other human factors will introduce additional difficulties in obtaining a good performance. A good algorithm for the speech feature extraction and classification is important to achieve a high success rate of recognition.

Electronic Books (eBooks) are small handheld devices that can offer rich contents and features, such as multimedia effects, instant dictionaries and bookmark functions etc. An eBook Reader should have no keyboard or mouse. The main input device is a touch screen. As many functions are implemented in a single eBook Reader, it is not convenient to access these functions through menus and hot keys. Thus, a one-step commanding process using speech is proposed for eBooks. To realize speech recognition for commanding, a modified neural fuzzy network with rule switches trained by an improved GA is proposed in this paper. The proposed NFN incorporated rule switches to change the network structure according to different patterns. On applying the proposed NFN, the performance of speech recognition is improved, and the time of training is shortened. The proposed GA has the advantage of offering a global solution. In this paper, the proposed NFN is used to recognize three Cantonese words, and implemented in an eBook Reader practically.

This paper is organized as follows. The NFN with rule switches is presented in section II. The improved GA is presented in section III. Some benchmark functions are used to test the improved GA. The speech recognition system is presented in section IV. In section V, the application and results are detailed. A conclusion will be drawn in section VI.

II. NEURAL FUZZY NETWORK WITH RULE SWITCHES

In general, an NFN achieves learning by: (1) defining a network structure, and (2) running a learning algorithm. In this paper, we use a fuzzy associative memory (FAM) [3-4, 10] type of rule base for the NFN. An FAM is formed by partitioning the universe of discourse of each fuzzy variable according to the level of fuzzy resolution chosen for the antecedents, thereby generating grids of FAM elements. Each grid element in the FAM corresponds to a fuzzy premise. An FAM is thus interpreted as a geometric or tabular representation of a fuzzy logic rule base. Usually, the network structure is fixed for a learning process as the number of rules is fixed. For an NFN, the number of possible rules may be too large while some rules may not be needed. The implementation cost is then unnecessarily increased.

Thus, a multi-input-multi-output NFN that can give an optimal number of rules and membership functions is proposed. The main difference between the proposed network and the traditional network is that a unit step function is added to each rule. The unit step functions is defined as,

$$\delta(\varsigma)=\begin{cases}0 \text{ if } \varsigma \le 0\\ 1 \text{ if } \varsigma > 0\end{cases}, \varsigma \in \Re \quad (1)$$

This is equivalent to adding a switch to each rule in the NFN. The rule is used if the corresponding rule switch is closed; otherwise, the rule is removed. Referring to Fig. 1, we define the input and output variables as x_i and y_j respectively; where $i = 1, 2, \ldots, n_{in}$; n_{in} is the number of input variables; $j = 1, 2, \ldots, n_{out}$; n_{out} is the number of output variables. The behavior of y_j of the NFN is governed by m_f fuzzy rules of the following format:

R_g: IF $x_1(t)$ is $A_{1_g}(x_1(t))$ AND $x_2(t)$ is $A_{2_g}(x_2(t))$ AND ... AND $x_{n_{in}}(t)$ is $A_{n_{in}g}(x_{n_{in}}(t))$

THEN $y_j(t)$ is w_{j_g}, $g = 1, 2, \ldots, m_f$; $t = 1, 2, \ldots, n_d$ (2)

where n_d is the number of input-output data pairs; $A_{j_g}(x_j(t))$ is the fuzzy term corresponding to $x_j(t)$; w_{j_g}, $j = 1, 2, \ldots, n_{out}$, is the output singleton of the rule g; $g \in [1,2,\ldots,m_f]$. The grade of the membership of each rule is defined as,

$$\mu_g(t) = A_{1_g}(x_1(t)) \times A_{2_g}(x_2(t)) \times \ldots \times A_{n_{in}g}(x_{n_{in}}(t)) \quad (3)$$

The j-th output of the NFN, $y_j(t)$, is defined as,

$$y_j(t) = \frac{\sum_{g=1}^{m_f} \mu_g(t) w_{j_g} \delta(\varsigma_{j_g})}{\sum_{g=1}^{m_f} \mu_g(t)} \quad (4)$$

where ς_{j_g} denotes the rule switch parameter of the g-th rule.

III. IMPROVED GENETIC ALGORITHM

Genetic algorithm (GA) is a directed random search technique [5, 9] that is widely applied in complex optimization problems [5-7, 9], where the number of parameters is large and the analytical global solutions are difficult to obtain. Much research effort has been put to improve the performance of GA. Different selection schemes and genetic operations were proposed. Selection schemes such as rank-based selection, elitist strategies, steady-state election and tournament selection were reported [8]. There are two kinds of genetic operations, namely crossover and mutation. The standard GA process [5-6, 9] is shown in Fig. 2. In this paper, the standard GA is modified and new genetic operations [14] are introduced. The improved GA process is shown in Fig. 3. Its details are given as follows.

A. Initial Population

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

$$P = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{pop\_size}\} \quad (5)$$

$$\mathbf{p}_i = [p_{i_1} \quad p_{i_2} \quad \cdots \quad p_{i_j} \quad \cdots \quad p_{i_{no\_vars}}],$$
$$i = 1, 2, \ldots, pop\_size; j = 1, 2, \ldots, no\_vars \quad (6)$$

$$para^j_{\min} \le p_{i_j} \le para^j_{\max} \quad (7)$$

where *pop_size* denotes the population size; *no_vars* denotes the number of variables to be tuned; p_{i_j}'s are the parameters to be tuned; $para^j_{\min}$ and $para^j_{\max}$ are the minimum and maximum values of the parameter p_{i_j} respectively for all i. It can be seen from (5) to (7) 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,

$$fitness = f(\mathbf{p}_i) \quad (8)$$

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 [5]. 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. First, we assign a probability q_i to the chromosome $\mathbf{p}_i$:

$$q_i = \frac{f(\mathbf{p}_i)}{\sum_{k=1}^{pop\_size} f(\mathbf{p}_k)}, i = 1, 2, \ldots, pop\_size \quad (9)$$

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

$$\hat{q}_i = \sum_{k=1}^{i} q_k, i = 1, 2, \ldots, pop\_size \quad (10)$$

Then we randomly generate a nonzero floating-point number, $d \in [0 \quad 1]$. The chromosome $\mathbf{p}_i$ is selected if $\hat{q}_{i-1} < d \le \hat{q}_i$ ($\hat{q}_0 = 0$). A chromosome having a larger $f(\mathbf{p}_i)$ will therefore have a higher chance to be selected. Thus, the best chromosomes will get more offspring, the average will stay and the worst will die off. In this process, two chromosomes will be selected to undergo the genetic operations.

D. Genetic Operations

The genetic operations are to reproduce some new chromosomes (offspring) from their selected parents. They include the crossover and the mutation operations.

Crossover

The crossover operation exchanges information of 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 equations:

$$\mathbf{os}_c^1 = \begin{bmatrix} os_1^1 & os_2^1 & \cdots & os_{no\_vars}^1 \end{bmatrix} = \frac{\mathbf{p}_1 + \mathbf{p}_2}{2} \tag{11}$$

$$\mathbf{os}_c^2 = \begin{bmatrix} os_1^2 & os_2^2 & \cdots & os_{no\_vars}^2 \end{bmatrix} = \mathbf{p}_{\max}(1-w) + \max(\mathbf{p}_1, \mathbf{p}_2)w \tag{12}$$

$$\mathbf{os}_c^3 = \begin{bmatrix} os_1^3 & os_2^3 & \cdots & os_{no\_vars}^3 \end{bmatrix} = \mathbf{p}_{\min}(1-w) + \min(\mathbf{p}_1, \mathbf{p}_2)w \tag{13}$$

$$\mathbf{os}_c^4 = \begin{bmatrix} os_1^4 & os_2^4 & \cdots & os_{no\_vars}^4 \end{bmatrix} = \frac{(\mathbf{p}_{\max} + \mathbf{p}_{\min})(1-w) + (\mathbf{p}_1 + \mathbf{p}_2)w}{2} \tag{14}$$

$$\mathbf{p}_{\max} = \begin{bmatrix} para_{\max}^1 & para_{\max}^2 & \cdots & para_{\max}^{no\_vars} \end{bmatrix} \tag{15}$$

$$\mathbf{p}_{\min} = \begin{bmatrix} para_{\min}^1 & para_{\min}^2 & \cdots & para_{\min}^{no\_vars} \end{bmatrix} \tag{16}$$

where $w \in [0 \quad 1]$ denotes a 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 $\mathbf{p}_2$, e.g. $\max([1 \quad -2 \quad 3], [2 \quad 3 \quad 1]) = [2 \quad 3 \quad 3]$. Similarly, $\min(\mathbf{p}_1, \mathbf{p}_2)$ gives a vector by taking the minimum value, e.g. $\min([1 \quad -2 \quad 3], [2 \quad 3 \quad 1]) = [1 \quad -2 \quad 1]$. Among $\mathbf{os}_c^1$ to $\mathbf{os}_c^4$, the one with the largest fitness value is the offspring $\mathbf{os}$ of the crossover operation:

$$\mathbf{os} \equiv \begin{bmatrix} os_1 & os_2 & \cdots & os_{no\_vars} \end{bmatrix} = \mathbf{os}_c^{i_{os}} \tag{17}$$

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

If the crossover operation can give 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 used to realize the crossover operation [5, 11-13]. The offspring generated by these methods, however, may not be better than that from our approach. As seen from (11) to (14), the potential offspring from the crossover operation spreads over the domain. While (11) and (14) result in searching around the centre region of the domain (a value of w near to 1 in (14) moves $\mathbf{os}_c^4$ to be near $\frac{\mathbf{p}_1 + \mathbf{p}_2}{2}$), (12) and (13) move the potential offspring to be near the domain boundary (a small w in (12) and (13) moves $\mathbf{os}_c^2$ and $\mathbf{os}_c^3$ to be near $\mathbf{p}_{\max}$ and $\mathbf{p}_{\min}$ respectively).

Mutation

The offspring (17) will then undergo the mutation operation, which changes the genes of the chromosomes. In general, various methods like boundary mutation, uniform mutation or non-uniform mutation [5, 11-13] can be employed to realize the mutation operation. In this paper, a different process of mutation is proposed as follows.

Every gene of the offspring $\mathbf{o}_s$ of (17) will have a chance to mutate governed by a probability of mutation, $p_m \in [0 \quad 1]$, which is defined by the user. This probability gives an expected number ($p_m \times no\_vars$) of genes that undergo the mutation. For each gene, a random number between 0 and 1 will be generated such that if it is less than or equal to p_m, the operation of mutation will take place on that gene. The gene of the offspring of (17) is then mutated by:

$$\hat{o}_{s_k} = \begin{cases} o_{s_k} + \Delta o_{s_k}^U \text{ if } f(\mathbf{o}_s + \Delta \mathbf{o}_{s_k}^U) \geq f(\mathbf{o}_s - \Delta \mathbf{o}_{s_k}^L) \\ o_{s_k} - \Delta o_{s_k}^L \text{ if } f(\mathbf{o}_s + \Delta \mathbf{o}_{s_k}^U) < f(\mathbf{o}_s - \Delta \mathbf{o}_{s_k}^L) \end{cases},$$

$k = 1, 2, \ldots, no\_vars$ (18)

where

$$\Delta o_{s_k}^U = w_{m_k} r(para_{\max}^k - o_{s_k}) \tag{19}$$

$$\Delta o_{s_k}^L = w_{m_k} r(o_{s_k} - para_{\min}^k) \tag{20}$$

$$\Delta \mathbf{o}_{s_k}^U = \begin{bmatrix} 0 & 0 & \cdots & \Delta o_{s_k}^U & \cdots & 0 \end{bmatrix} \tag{21}$$

$$\Delta \mathbf{o}_{s_k}^L = \begin{bmatrix} 0 & 0 & \cdots & \Delta o_{s_k}^L & \cdots & 0 \end{bmatrix} \tag{22}$$

$r \in [0 \quad 1]$ is a randomly generated number; $w_{m_k} \in (0 \quad 1]$ is a weight governing the magnitudes of $\Delta o_{s_k}^U$ and $\Delta o_{s_k}^L$. The value of weight w_{m_k} depends on the value of $\frac{\tau}{T}$, where τ is the iteration number and T is the total number of iterations. The value of weight w_{m_k} should decrease as $\frac{\tau}{T}$ increases. Based on this idea, a monotonic decreasing function governing w_{m_k} is proposed as follows,

$$w_{m_k} = w_f \left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_\tau}} \geq 0 \tag{23}$$

where $w_f \in [0 \quad 1]$ and $w_\tau > 0$ determine the initial value and the decay rate respectively. Their values are chosen by the user. For a large value of w_f, it can be seen from (19) and (20) that $\Delta o_{s_k}^U \approx r(para_{\max}^k - o_{s_k})$ and $\Delta o_{s_k}^L \approx r(o_{s_k} - para_{\min}^k)$ as $\left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_\tau}} \approx 1$, which ensure a large search space. When the value of $\left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_\tau}} \approx 0$, the values of $\Delta o_{s_k}^U$ and $\Delta o_{s_k}^L$ are small to ensure a small search space for fine-tuning.

E. Reproduction

The new offspring will be evaluated using the fitness function of (8). This new offspring will replace the chromosome with the smallest fitness value among the population if a randomly generated number within 0 to 1 is smaller than $p_a \in [0\ \ 1]$, which is the probability of acceptance defined by users. Otherwise, the new offspring will replace the chromosome with the smallest fitness value only if the fitness value of the offspring is greater than the fitness value of that chromosome in the population. p_a is effectively the probability of accepting a bad offspring in order to reduce the chance of converging to a local optimum.

After the operation of selection, crossover, mutation and reproduction, 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. a defined number of iteration has been reached.

IV. SPEECH RECOGNITION SYSTEM

A speech recognition system for three Cantonese command words for an eBook is proposed. Referring to Fig. 4, the speech recognition system involves five processes: speech recording, electrical signal conversion, feature extraction, feature classification, and command execution.

Cantonese words are recorded by the eBook Reader. The speech signals are sampled and fed to the feature extractor to get the feature parameters representing the spoken word. There are various kinds of parametric representations, such as short time energy, zero crossing rates and level crossing rates etc. [1]. In this paper, the filter-bank spectrum analysis modeling method is used. Speech signals are transformed from time-domain to frequency domain using Fast Fourier Transform. The frequency components of the spoken words are analyzed by a uniform filter-bank, which comprises a set of band-pass filters with the same bandwidth evenly distributed along the speech spectrum. The mean amplitude of the speech signal at the output of each band-pass filter is:

$$c_o(\alpha) = \frac{10\log\sum_{b=1}^{m} s_{\alpha b}}{m},\ \alpha = 1, 2, 3\ \ldots, k. \tag{24}$$

where c_o denotes the mean amplitude value of the frame, α denotes the speech frame number, k is the total number of frames, m is the number of frequency components within the bandwidth of each band-pass filter, $s_{\alpha b}$ denotes the amplitude of the speech signal of frame α at the b-th sampled frequency.

$$d_o(\alpha+1) = c_o(\alpha+1) - c_o(\alpha), \tag{25}$$

$$d_o(1) = c_o(1) \tag{26}$$

(24) reflects the mean energy of the speech signal in a given frequency band. The correlation between the band-pass filters given by (25) and (26) will be used as the feature parameters to represent the spoken word. These feature parameters are fed to the modified NFN, which is used to recognize the input speech command. The NFN inputs are $\mathbf{x} = \frac{\mathbf{d}_o}{\|\mathbf{d}_o\|}$,where $\mathbf{d}_o = [d_o(1)\ \ d_o(2)\ \ \cdots\ \ d_o(k)]$, $\|\|$ denotes the l_2 vector norm. The NFN outputs are given by,

$$\mathbf{y}_d = [q_1, q_2, q_3, \ldots, q_{n_{out}}] \tag{27}$$

where q_i, i =1, 2, …, n_{out}, denotes the value of target class. The desired value of q_i is either 0 or 1, reflecting the class of the speech input. The value of q_i is near to 1 if the corresponding input pattern belongs to class i; otherwise, it is near to 0. The aim of the classifier is used to maximize the fitness function as follows.

$$err = \frac{\sum_{t=1}^{num\_pat} \frac{\|\mathbf{y}_d(t) - \mathbf{y}(t)\|^2}{n_{out}}}{num\_pat} \tag{28}$$

$$fitness = \frac{1}{1+err} \tag{29}$$

where $\mathbf{y}_d(t)$ is the desired output vector; $\mathbf{y}(t)$ is the network output vector; *num_pat* is the number of training patterns. During the operation, the index of the largest element of $\mathbf{y}(t)$ indicates the possible class that the input pattern belongs to.

V. APPLICATION AND RESULTS

The speech recognition system shown in Fig. 4 is implemented in a prototyped eBook Reader. The three Cantonese command words are sampled at 11kHz, 8-bit mono and the amplitude of each sample is normalized to lie between –1 to 1. 100 training patterns and 20 testing patterns for each command word are recorded from a male speaker. The Cantonese words, "筆", "大" and "註" (3 classes), are used to represent the eBook commands of "Using Pen", "Zoom-in" and "Annotation", respectively. We use (24) to (26) to obtain 20 feature parameters from each command word. Examples of the feature parameters for the three Cantonese command words are shown in Fig. 5. A 20-input-3-output NFN is used to recognize the three command words. The membership functions of the modified NFN are defined as follows.

$$A_{ig}(x_i(t)) = e^{\frac{-(x_i(t)-\bar{x}_{ig})^2}{2\sigma_{ig}^2}} \tag{30}$$

where the parameters $\bar{x}_{ig}$ and σ_{ig} are the mean and the standard deviation of the g-th membership function respectively. The improved GA is employed to train the proposed NFN based on the training patterns to maximize the following fitness function.

$$err = \frac{\sum_{t=1}^{300} \frac{\|\mathbf{y}_d(t) - \mathbf{y}(t)\|^2}{3}}{300} \tag{31}$$

$$fitness = \frac{1}{1+err} \tag{32}$$

The number of training patterns for the modified NFN is 300. The first 100 training patterns are the feature

parameters of the Cantonese word "筆". Thus, the first 100 vectors of y_d are [1 0 0]. The next 100 training patterns are the feature parameters of the Cantonese word "大", and the corresponding 100 vectors of y_d are [0 1 0]. The last 100 training patterns are the feature parameters of the Cantonese word "註", and the corresponding 100 vectors of y_d are [0 0 1]. The number of membership functions, m_f, for the modified NFN is changed from 4 to 10. The learning process is done by a personal computer with a P4 1.4GHz CPU and 256 MB RAM. The number of iterations for training is 3000; $w = 0.5$, $p_m = 0.01$; $w_f = 0.5$, $w_\tau = 5$, $p_a = 0.01$ for all cases. After the training, 60 testing patterns (3 Cantonese words × 20) are used to test the performance of the modified NFN. The fitness value from training and the number of the parameters used in the training are tabulated in Table I. The fitness value for testing and the testing error of each spoken word are tabulated in Table II.

For comparison, a traditional NFN trained by GA with arithmetic crossover and non-uniform mutation [9] is also used to do the same job. The iteration number is 3000, the shape parameter, the probability of crossover and the probability of mutation [9] are chosen to be 1, 0.8 and 0.01 respectively. The results are tabulated in Tables III and IV.

Referring to tables I and III, the proposed network uses fewer parameters than the traditional network to get a better result. Taking the recognition error as a reference, we observe from tables II and IV that the best result occurs when $m_f = 8$ (3/60 errors) for the proposed NFN, and $m_f = 9$ (9/60 errors) for the traditional NFN. This indicates that the proposed NFN performs better.

VI. CONCLUSION

A modified neural-fuzzy network with rule switches has been proposed. An improved genetic algorithm has been used to train the membership function parameters and the rule set of the modified NFN. The trained NFN has been applied to recognize Cantonese speech commands, and implemented in an eBook Reader experimentally.

REFERENCES

[1] L. Rabiner and B.H. Juang, *Fundamentals of Speech Recognition.* Englewood Cliffs, NJ: Prentice Hall, 1993.

[2] T. Lee, W.K. Lo, P.C. Ching, and H. Meng, "Spoken language resources for Cantonese speech processing," *Speech Communication*, vol. 36, Issues 3-4, pp. 327-342, March 2002.

[3] B. Kosko, *Neural Networks and Fuzzy System: A Dynamical Systems Approach to Machine Intelligence.* Englewood Cliffs, NJ: Prentice Hall, 1991.

[4] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd Ed. Prentice Hall, 1999.

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

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

[7] Y. Hanaki, T. Hashiyama, and S. Okuma, "Accelerated evolutionary computation using fitness estimation," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, 1999, vol. 1, pp. 643-648.

[8] L. Davis, *Handbook of Genetic Algorithms.* NY: Van Nostrand Reinhold, 1991.

[9] Z. Michalewicz, *Genetic Algorithm + Data Structures = Evolution Programs*, 2nd extended ed. Springer-Verlag, 1994.

[10] M. Brown and C. Harris, *Neuralfuzzy Adaptive Modeling and Control.* Prentice Hall, 1994.

[11] M. Srinivas and L.M. Patnaik, "Genetic algorithms: a survey," *IEEE Computer*, vol. 27, issue 6, pp. 17-26, June 1994.

[12] X. Wang and M. Elbuluk, "Neural network control of induction machines using genetic algorithm training," in *Conf. Record 31st IAS Annual Meeting*, vol. 3, 1996, pp. 1733-1740.

[13] L. Davis, *Handbook of Genetic Algorithms.* NY: Van Nostrand Reinhold, 1991.

[14] 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.

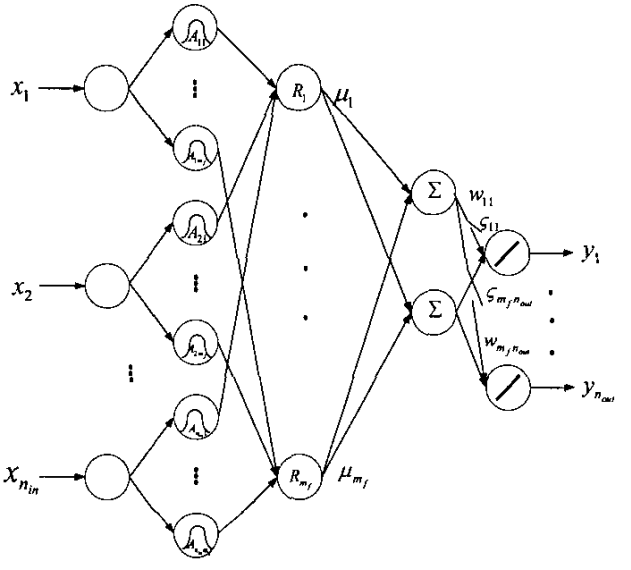

Fig. 1. Proposed neural fuzzy network with rule switches.

```
Procedure of the standard 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 genetic operations (crossover and mutation)
      reproduce a new P(τ)
      evaluate f(P(τ))
    end
end
```

Fig. 2. Procedure of the standard GA.

```
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 (11) - (17)
    perform mutation operation according to equations
    (18) - (23)
    to generate the offspring os
    // reproduce a new P(τ)
    if random number < pa  // pa: probability of acceptance
        os replaces the chromosome with the smallest fitness
        value in the population
    else if f(os) > smallest fitness value in the P(τ-1)
        os replaces the chromosome with the smallest
        fitness value
    end
        evaluate f(P(τ))
    end
end
```

Fig. 3. Procedure of the improved GA.

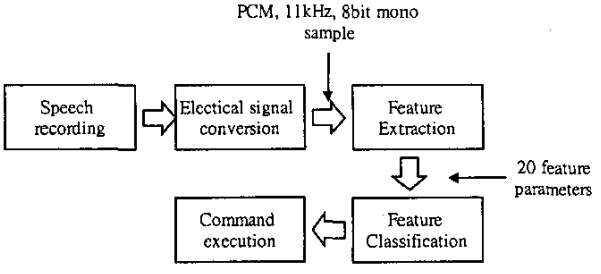

Fig. 4. Block diagram of Cantonese command recognition system.

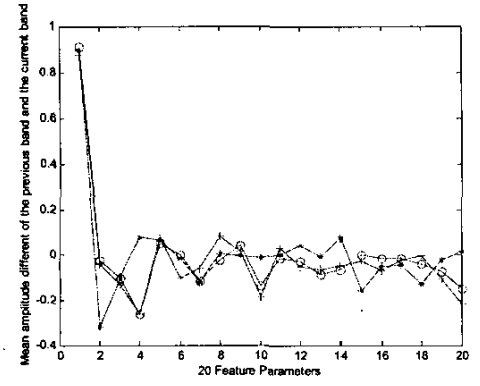

Fig. 5. 20 speech feature parameters of the first training template (+: 筆, o: 大, *: 註).

m_f	4	5	6	7	8	9	10
Fitness	0.9997	0.9997	0.9968	0.9991	0.9978	0.9992	0.9994
No. of parameters	164	206	245	286	326	368	408

Table I. Fitness value and number of parameters used in the proposed NFN trained by the improved GA. (100 training patterns for each command)

m_f	4	5	6	7	8	9	10
fitness	0.8865	0.8765	0.9050	0.9008	0.9686	0.9611	0.8929
筆	0	0	0	0	0	0	0
大	15	15	14	15	3	4	16
註	0	0	0	0	0	0	0

Table II. Number of recognition error and the fitness value for testing the 3 Cantonese command words using the proposed NFN with rule switches trained by the improved GA (20 testing patterns for each command).

m_f	4	5	6	7	8	9	10
Fitness	0.9682	0.9504	0.9937	0.9903	0.9821	0.9922	0.9806
No. of parameters	173	216	259	302	345	388	431

Table III. Fitness value and number of parameters used in the traditional NFN trained by GA with arithmetic crossover and non-uniform mutation (100 training patterns for each command).

m_f	4	5	6	7	8	9	10
fitness	0.8588	0.8844	0.8546	0.9144	0.9044	0.9333	0.9208
筆	0	0	0	0	1	1	0
大	20	14	20	10	9	8	9
註	0	2	0	0	0	0	2

Table IV. Number of recognition error and the fitness value for testing the 3 Cantonese command words using the traditional NFN trained by GA with arithmetic crossover and non-uniform mutation (20 testing patterns for each command).