

Copyright Undertaking

This thesis is protected by copyright, with all rights reserved.

By reading and using the thesis, the reader understands and agrees to the following terms:

1. The reader will abide by the rules and legal ordinances governing copyright regarding the use of the thesis.
2. The reader will use the thesis for the purpose of research or private study only and not for distribution or further reproduction or any other purpose.
3. The reader agrees to indemnify and hold the University harmless from and against any loss, damage, cost, liability or expenses arising from copyright infringement or unauthorized usage.

If you have reasons to believe that any materials in this thesis are deemed not suitable to be distributed in this form, or a copyright owner having difficulty with the material being included in our database, please contact lbsys@polyu.edu.hk providing details. The Library will look into your claim and consider taking remedial action upon receipt of the written requests.

Neural Network Techniques
for
Graffiti Interpretation and Speech Recognition

by
LEUNG KOON FAI, BEng(Hons)

For the Degree of Master of Philosophy in the Department of
Electronic and Information Engineering at the Hong Kong
Polytechnic University

March 2004



Pao Yue-kong Library
PolyU • Hong Kong

ABSTRACT

This thesis explores the neural network classification techniques on an electronic book (eBook) reading device. Two areas of application are addressed: a graffiti interpreter and a Cantonese-speech recognizer. Different structures of neural networks and hybrid neural networks incorporating fuzzy sets are used to realize the applications.

An eBook reading device enhances our reading environment with interactive and multimedia features. Input for this device is possibly made using a stylus on a touch-screen or voice through a microphone; practically, the former is a pattern recognition (graffiti interpretation) problem and the latter is a speech recognition problem.

With graffiti interpretation, eBook users can take full advantage of the graffiti input to issue commands or input texts. The interpretation is done by the template matching technique. Two approaches are developed to realize the pattern recognition, which apply a self-structured neural network and a self-structure neural-fuzzy network. Improved from a 3-layer fully connected neural network/neural-fuzzy network, the self-structured network has a variable structure that adapts to the characteristics of the input patterns by incorporating link switches. By properly determining the states of the link-switches through training, the dummy links can be eliminated. Simulation results show that the self-structure network performs better than a fixed-structure network in terms of the network size.

With a speech recognizer, eBook users can use natural speech to execute some functions of the eBook and enter characters whenever necessary. Four approaches are proposed to recognize Cantonese speech. Of them, three are feed-forward neural

networks, and one is a recurrent neural network.

As the first approach, the self-structured neural-fuzzy network used for graffiti interpretation is also applied to recognize Cantonese-speech commands. Then, a neural-fuzzy network and a neural network are modified by adding associative memory to provide the network parameters. In both of these approaches, the neural-fuzzy network/neural network effectively has variable parameters that change with respect to the input patterns. Thus, the learning ability can be enhanced for the case if two feature vectors belong to the same class but sparsely distributed. Results will be given to demonstrate the improvement on recognition accuracy, network complexity and learning rate. A discussion on comparing the various approaches will also be given.

By using a recurrent neural network, the sequential properties of the double-syllable Cantonese-digit can be modeled. The fourth approach therefore involves an associative memory for a recurrent neural network. Results will be given to demonstrate the merits of the proposed approach. A discussion on the comparison between the static approaches and the dynamic approach will also be given.

In this thesis, all neural networks are trained by an improved genetic algorithm (GA). The details about this algorithm and its performance in some benchmark test functions will be given in the Appendix.

ACKNOWLEDGEMENT

First and foremost, I would like to express my deepest thanks to my chief supervisor, Dr Frank H. Leung for his advice and great supports. Without his guidance, patience, and perseverance, I would never be able to finish my study. Thanks are extended to my second supervisor, Dr Peter K.S. Tam for his motivation and helpful advice. In addition, I would like to thank my colleagues, Dr H.K. Lam and Mr Steve Ling, for their help and encouragement in my work. Last but not least, I truly appreciate the administrative help from the staff in the General Office of the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University.

The work described in this thesis was substantially supported by a grant from the Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University.

STATEMENT OF ORIGINALITY

The following contributions reported in this thesis are claimed to be original.

1. *A self-structured neural network for graffiti recognition is developed (Chapter 3, Section 3.2).* With a link switch introduced to each inter-node link, the structure of the neural network can be tuned during the training process and will be optimized after the training. The implementation cost of the network can be reduced. The self-structured neural network is applied to interpret 3 graffiti commands (Chapter 5, Section 5.3.2).
2. *A self-structured neural-fuzzy network is developed for graffiti recognition and Cantonese speech recognition (Chapter 3, Section 3.3 and Chapter 4, Section 4.3.1).* By adopting a variable structure, the neural-fuzzy network structure can be optimized and the membership functions can be generated automatically. A simpler network structure, higher recognition rate and lower implementation cost can be achieved. The self-structured neural-fuzzy network is applied to interpret 9 graffiti digits and 3 graffiti characters (Chapter 5, Section 5.3.3). It is also applied to recognize 3 Cantonese speech commands (Chapter 5, Section 5.4.1.2).
3. *A variable-parameter neural-fuzzy network and a variable-parameter neural network are proposed for mono-syllabic Cantonese speech recognition (Chapter 4, Section 4.3.2 and Section 4.3.3).* Under a specific structure, associative memory is present in the networks. The associative memory improves the searching area of the networks and thus increases the recognition rate of the Cantonese speech recognition system. The variable-parameter neural-fuzzy network is applied to recognize 10 Cantonese-digit speeches (Chapter 5, Section 5.4.1.3) and the variable-parameter neural network is applied to recognize 5 Cantonese-command

speeches (Chapter 5, Section 5.4.1.4).

4. *A dynamic variable-parameter neural network is proposed for multi-syllable Cantonese speech recognition (Chapter 4, Section 4.4).* By introducing associative memory to a recurrent neural network, the network complexity can be reduced and the learning ability can be increased. The dynamic variable-parameter neural network is applied to recognize 11 single-syllable Cantonese-digit speeches and 2 double-syllable Cantonese-number speeches (Chapter 5, Section 5.4.2.2).

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written nor material which has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____ (Signed)

_____ (Name of student)

TABLE OF CONTENTS

ABSTRACT

ACKNOWLEDGEMENT

STATEMENT OF ORIGINALITY

TABLE OF CONTENTS

AUTHOR'S PUBLICATIONS

LIST OF SYMBOLS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

<i>1</i>	<i>INTRODUCTION.....</i>	<i>1</i>
1.1	Neural Network for Graffiti Interpretation	1
1.2	Neural Network for Speech Recognition.....	2
1.3	Electronic Book.....	4
1.4	Outline of the Thesis.....	5
<i>2</i>	<i>LITERATURE REVIEW.....</i>	<i>6</i>
2.1	Neural Network for Classification	6
2.1.1	Fuzzy Logic	
2.1.2	Neural Network Topologies	
2.1.3	Evolutionary Computation	
2.2	Pattern Recognition (Handwritten Recognition).....	8
2.3	Speech Recognition.....	10
2.3.1	Cantonese Speech Recognition	
2.4	Electronic Book.....	14

3	NEURAL NETWORK FOR GRAFFITI INTERPRETATION.....	16
3.1	Introduction	16
3.2	Self-structured Neural Network.....	17
3.2.1	Network Structure	
3.2.2	Training	
3.3	Self-structured Neural-fuzzy Network	21
3.3.1	Network Structure	
3.3.2	Training and Rule Switches Tuning	
3.3.3	Graffiti Interpreter	
3.4	Comparison	26
3.5	Summary	28
4	NEURAL NETWORK FOR SPEECH RECOGNITION.....	30
4.1	Introduction	30
4.2	Cantonese Speech Recognition System.....	31
4.3	Static Pattern Classification for Cantonese Speech Recognition	32
4.3.1	Self-structured Neural-fuzzy Network	
4.3.1.1	Network structure	
4.3.2	Variable-parameter Neural-fuzzy Network	
4.3.2.1	Tuner neural-fuzzy network	
4.3.2.2	Classifier neural-fuzzy network	
4.3.3	Variable-parameter Neural Network	
4.3.3.1	Tuner neural network	
4.3.3.2	Classifier neural network	
4.3.4	Training and Classification	
4.4	Dynamic Pattern Classification for Cantonese Speech Recognition ...	41
4.4.1	Dynamic Variable-parameter Neural Network	
4.4.1.1	Tuner neural network	
4.4.1.2	Classifier recurrent neural network	
4.4.2	Training and Classification	
4.5	Comparison	48
4.6	Summary	51
5	APPLICATION TO ELECTRONIC BOOK AND RESULTS	53
5.1	Introduction	53
5.2	Electronic Book Reader	53
5.3	Graffiti Interpretation Applications	54
5.3.1	Feature Point Extraction Methodology	

5.3.2	<i>Command Interpretation by the Self-structured NN</i>	
5.3.3	<i>Graffiti Interpretation by the Self-structured NFN</i>	
5.3.4	<i>Comparison Between the Self-Structured NN and the Self-Structured NFN</i>	
5.3.5	<i>Comparison between the two proposed approaches and the commercial software package.</i>	
5.4	<i>Cantonese Speech Recognition</i>	69
5.4.1	<i>Static Pattern Classification for Cantonese Speech Recognition System</i>	
5.4.1.1	<i>Feature extraction methodology</i>	
5.4.1.2	<i>Self-structured NFN</i>	
5.4.1.3	<i>Variable-parameter NFN</i>	
5.4.1.4	<i>Variable-parameter NN</i>	
5.4.1.5	<i>Comparison between the three proposed approaches</i>	
5.4.2	<i>Dynamic Pattern Classification for Cantonese Speech Recognition System</i>	
5.4.2.1	<i>Speech feature extraction methodology</i>	
5.4.2.2	<i>Dynamic variable-parameter NN</i>	
5.4.3	<i>Comparison for Static and Dynamic Pattern Classification for Cantonese Speech Recognition</i>	
5.4.4	<i>Comparison for Static Pattern Classification, Dynamic Pattern Classification and Commercial Classification for Cantonese Speech Recognition</i>	
6	<i>CONCLUSION</i>	100
6.1	<i>Achievements</i>	100
6.2	<i>Further Research</i>	103
	<i>APPENDIX: IMPROVED GENETIC ALGORITHM</i>	104
A.1	<i>Introduction</i>	104
A.2	<i>Benchmark Tests</i>	111
	<i>REFERENCES</i>	117

AUTHOR'S PUBLICATIONS

INTERNATIONAL JOURNAL PAPER

- [1] 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.*, vol. 51, no. 2, pp. 464-471, Apr. 2004.

- [2] K.F. Leung, F.H.F. Leung, H.K. Lam and P.K.S. Tam, "Application of a modified neural-fuzzy network and an improved GA to speech recognition", *International Journal of Computational Intelligence and Applications*, (under revision).

- [3] K.F. Leung, F.H.F. Leung, H.K. Lam and P.K.S. Tam, "Cantonese speech command recognition using a neural-fuzzy network and genetic algorithm approach", *International Journal of Approximate Reasoning*, (under review).

- [4] K.F. Leung, F.H.F. Leung, S.H. Ling, and H.K. Lam, "Graffiti interpretation using a neural network with link switches and an improved genetic algorithm," *IEEE Trans. Industrial Electronics*, (under review).

INTERNATIONAL CONFERENCE PAPERS

- [1] K.F. Leung, F.H.F. Leung, H.K. Lam, and P.K.S. Tam, "Recognition of speech commands using a modified neural-fuzzy network and an improved GA," in *Proc. 12th IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE 2003)*, St. Louis, MO, 25-28 May 2003, pp. 190-195.

- [2] K.F. Leung, F.H.F. Leung, H.K. Lam, and P.K.S. Tam, "Neural-fuzzy network and genetic algorithm approach for Cantonese speech command recognition," in *Proc. 12th IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE 2003)*, St. Louis, MO, 25-28 May 2003, pp. 208-213.

- [3] K.F. Leung, H.K. Lam, F.H.F. Leung, and P.K.S. Tam, "Graffiti commands interpretation for eBooks using a self-structured neural network and genetic algorithm," in *Proc. 2002 Int. Joint Conf. Neural Networks (IJCNN 2002), World Congress on Computational Intelligence (WCCI 2002)*, Honolulu, Hawaii, May 12-17, 2002, 2487 - 2492.

- [4] H.K. Lam, K.F. Leung, S.H. Ling, F.H.F. Leung, and P.K.S. Tam, "On interpretation of graffiti digits and commands for eBooks: neural-fuzzy network and genetic algorithm approach," in *Proc. 2002 IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE 2002), World Congress on Computational Intelligence (WCCI 2002)*, Honolulu, Hawaii, May 12-17, 2002, pp. 443 - 448.

- [5] H.K. Lam, S.H. Ling, K.F. Leung, and F.H.F. Leung, "On interpretation of graffiti commands for eBooks using a neural network and an improved genetic algorithm," in *Proc. 10th IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE 2001)*, Melbourne, Australia, 2-5 December 2001, pp. 1464-1467.

LIST OF SYMBOLS

α	Number of frequency components entering the band-pass filter
b_j^1	Bias for the j -th hidden node in the 1 st layer
b_k^2	Bias for the k -th output node in the 2 nd layer
c_n^β	Mean power of a speech frame generated by the β -th band-pass filter for the n -th frame
\mathbf{D}_n	Vector of magnitude difference between two consecutive band-pass-filter outputs of the n -th speech frame
\mathbf{d}_η	Desired network output of the η -th speech group
δ_j	Mean parameter of $tf(\cdot)$ in the j -th hidden node
$\delta(\cdot)$	Unit step function
E_n	Speech energy of the n -th speech frame
err	Error of the recognition network
\mathbf{e}_w	Error weights
$fitness$	Fitness value of the recognition network
$FFT(\cdot)$	Fast Fourier transform function
$floor(\cdot)$	Floor function of a floating point number
$\mathbf{g}(\cdot)$	Non-linear function
G_η	Number of frames in the η -th group
$logsig(\cdot)$	Logarithm sigmoid function
λ_k	Weight of the k -th bias term for the output layer of the classifier NN
m_i	Number of membership functions of input variable x_i
m_n	Starting index of the n -th frame
μ_g	Grade of the membership of the g -th rule
μ_{qj}^{fb}	Weight of the link between the q -th recurrent input (fb) and the j -th hidden node at the CRNN
μ_{ij}^{ff}	Weights of the links between the i -th input and the j -th hidden

	node of the CRNN
N	Number of samples in a frame
no_frame	Number of frames in a speech signal
no_group	Number of groups that can be segmented from the speech
no_pat	Number of word classes
no_sample	Number of time samples of the whole speech
num	Dimension of an output vector
num_pat	Number of training patterns
n_h, h_1, h_2	Number of hidden nodes
n_{in}	Number of inputs
n_{out}	Number of outputs
p	Number of fuzzy rules
$\rho\beta$	Starting index of the β -th band-pass filter
$Re\{\cdot\}$	Real part of a frequency component
s_{ij}^1	Link switch parameter of i -th input node to the j -th hidden node in the 1 st layer
s_{jk}^2	Link switch parameter of j -th hidden node to the k -th output node in the 2 nd layer
s_j^1	Link switch parameter of the bias to the j -th hidden nodes in the 1 st layer
s_k^2	Link switch parameter of the bias to the k -th outputs in the 2 nd layer
S_i	Similarity between the NFN inputs and the NFN outputs at i -th classification NFN
Sf_n	Frequency spectrum of the n -th speech frame
s_n	n -th speech frame in time-domain
$sort(\cdot)$	Sorting function
σ	Standard deviation parameter of the hidden node transfer function $tf(\cdot)$ of the CNN
$tansig(\cdot)$	Tangent sigmoid function
$tf(\cdot)$	Hidden node activation function

$v_{ij}, \mu_{ij},$	Weights of the links between the i -th input node and the j -th hidden node
$w_{jk}, \omega_{jg}, \lambda_{jk}$	Weights of the link between the j -th hidden nodes and the k -th or g -th output nodes
w_{dig}	Parameter set of the dig -th word class
$wh(\cdot)$	Hamming window
$wt(\cdot)$	Triangular window
w_{jg}	Output singleton of rule g at output j
ω_g	Weight of the g -th bias term for the output layer of TNN
\bar{x}_{ig_i}	Mean value parameter of the g_i -th membership function of the i -th input
χ_j	Input of the j -th hidden node transfer function $tf(\cdot)$
y_k	k -th output of the recognition network
$\gamma_g(t)$	g -th output for the t -th speech frame of the TNN
\bar{y}_i	Normalized outputs of the i -th neural-fuzzy network
y_η	Output of the η -th group
$z_i(t), x_i$	i -th input of the recognition network
\bar{z}	Normalized network input vector
$z^d(t), y_d(t)$	Desired output vector of the recognition network
Z_n	Zero-crossing rate of the n -th speech frame

LIST OF FIGURES

- Fig. 3-1. Proposed 3-layer neural network with switches.
- Fig. 3-2. Proposed neural-fuzzy network.
- Fig. 3-3. Block diagram of the graffiti interpreter.
- Fig. 4-1. Block diagram of a Cantonese-digit/Cantonese-command speech recognition system.
- Fig. 4-2. Speech signal of the Cantonese-digit speech, digit “1”.
- Fig. 4-3. Block diagram of the modified neural-fuzzy network.
- Fig. 4-4. Three-layer neural-fuzzy network.
- Fig. 4-5. Architecture of the proposed neural network.
- Fig. 4-6. Structure of the proposed neural network.
- Fig. 4-7. Effects of δ_j and σ to $tf(\cdot)$.
- Fig. 4-8. Time-domain waveforms of the Cantonese digits “12” and “20”.
- Fig. 4-9. Modified recurrent neural network structure.
- Fig. 4-10. Block diagram of the proposed training and classification process.
- Fig. 4-11. (a) Two data sets in spatial domain. (b) Feature curves of the two sets.
- Fig. 5-1. Electronic book reading device.
- Fig. 5-2. Block diagram of the graffiti interpretation system.
- Fig. 5-3. Feature points capturing method.
- Fig. 5-4. Graffiti used in the eBook (the dot indicates the starting point): (a) square (b) triangle (c) straight line.
- Fig. 5-5. Output values of the two neural networks for the 24 training graffiti:
Solid line with (*): self-structured NN; Dash line with (+): traditional

NN.

- Fig. 5-6. Output values of the two neural networks for the 12 testing graffiti:
Solid line with (*): self-structured NN; Dash line with (+): traditional NN.
- Fig. 5-7. Graffiti of the numeric and control characters (the dot indicates the starting point of the graffiti).
- Fig. 5-8. Similarity values given by the 16 self-structured NFNs for the 480 testing graffiti patterns (30 for each type).
- Fig. 5-9. Block diagram of the static pattern classification for Cantonese speech recognition.
- Fig. 5-10. Comparison of convergence between the improved GA (solid-line) and the back-propagation (dotted-line).
- Fig. 5-11. Comparison of convergence between the improved GA (solid-line) and the back-propagation (dotted-line).
- Fig. 5-12. Block diagram of the dynamic pattern classification for the Cantonese-digit speech recognition system.
- Fig. 5-13. Speech acoustic feature groups of (i) a single-syllable Cantonese speech, (ii) a double-syllable Cantonese speech.
- Fig. 5-14. Comparison of convergence between the improved GA (solid-line) and the back-propagation (dotted-line).
- Fig. A-1. Procedure of the standard GA.
- Fig. A-2. Procedure of the improved GA.

LIST OF TABLES

- Table 3-1. Comparisons between the four neural networks.
- Table 3-2. Comparisons between the proposed self-structured NN and NFN.
- Table 4-1. Comparisons between the self-structured NFN, the variable-parameter NFN and the variable-parameter NN.
- Table 4-2. Comparisons between the static and dynamic pattern classification approaches.
- Table 4-3. Summary on the uses of the proposed approaches.
- Table 5-1. Results of the self-structured NN and the traditional NN for interpreting graffiti commands.
- Table 5-2. Results of the proposed self-structured NFNs for interpreting graffiti.
- Table 5-3. Results of the traditional neural-fuzzy networks for interpreting graffiti.
- Table 5-4. Results given by the self-structured NN.
- Table 5-5. Results given by the self-structured NFN.
- Table 5-6. Comparison of the performance of the proposed approaches and the PDA software tool.
- Table 5-7. Fitness values and number of parameters of the self-structured NFNs.
- Table 5-8. Number of recognition errors and fitness values for the 3 Cantonese command words (20 testing patterns for each command).
- Table 5-9. Fitness values and number of parameters of the traditional NFNs.
- Table 5-10. Number of recognition errors and fitness values for the 3 Cantonese command words using the traditional NFNs (20 testing patterns for each command).

- Table 5-11. Fitness values under different combinations of numbers of membership functions in the proposed NFN.
- Table 5-12. Number of recognition errors for the Cantonese digits '0 - 9' given by the variable-parameter NFN (20 testing patterns for each digit).
- Table 5-13. Fitness values given by the traditional neural-fuzzy network.
- Table 5-14. Number of recognition errors for the Cantonese digits '0 - 9' given by the traditional neural-fuzzy network (20 testing patterns for each digit).
- Table 5-15. Results given by the variable-parameter NNs.
- Table 5-16. Results given by the traditional neural networks.
- Table 5-17. Results given by the self-structured NFNs.
- Table 5-18. Results given by the variable-parameter NFNs.
- Table 5-19. Results given by the variable-parameter NNs.
- Table 5-20. Number of recognition errors given by the self-structured NFNs.
- Table 5-21. Performance of the self-structured NFNs.
- Table 5-22. Number of recognition errors given by the variable-parameter NFNs.
- Table 5-23. Performance of the variable-parameter NFNs.
- Table 5-24. Number of recognition errors given by the variable-parameter NNs.
- Table 5-25. Performance of the variable-parameter NNs.
- Table 5-26. Recognition rate for 11 Cantonese-number speeches and 2 double-syllable Cantonese-number speeches (test data).
- Table 5-27. Overall performance on testing and training for the 13 Cantonese-number speeches.
- Table 5-28. Recognition rate for the 13 Cantonese speeches (test data).
- Table 5-29. Overall performance on testing and training for the 13 Cantonese speeches.

Table 5-30. Performance comparison between the 2 proposed approaches and the commercial approach.

Table A-1. Average fitness values obtained from the improved GA and the standard GA for the benchmark test functions.

Table A-2. Control parameters of GAs for the benchmark test functions.

ABBREVIATIONS

CNFN	Classifier Neural-Fuzzy Network
CNN	Classifier Neural Network
CRNN	Classifier Recurrent Neural Network
eBook	Electronic Book
FAM	Fuzzy Associative Memory
GA	Genetic Algorithm
NFN	Neural-Fuzzy Network
NN	Neural Network
TNN	Tuner Neural Network
TNFN	Tuner Neural-fuzzy Network

CHAPTER 1

INTRODUCTION

Neural Network (NN) and Neural-Fuzzy Network (NFN) are two topologies to mimic the biological information processing mechanism. While humans are good at symbolic recognition, it is hard for machines to learn recognizing symbolic objects. To alleviate the problem, mathematical models for symbolic objects should be created. As a result, machines can learn the features of the objects by using computational intelligence techniques, such as neural networks and neural-fuzzy networks. When expert knowledge is needed in some applications, a neural network incorporated with fuzzy rules is a possible solution. In this thesis, modifications to the neural network/neural-fuzzy network topologies will be investigated. Then, the modified networks will be applied to realize graffiti interpretations and speech recognitions in an eBook environment.

1.1 Neural Network for Graffiti Interpretation

As we know, the most common input to an eBook reading device is the stylus working on a touch-screen. The touch-screen is able to sense users' sketches made by the stylus. However, the interpretation of the sketches involves understanding the symbolic objects of input graffiti, which is a relatively difficult task for machines. One motivation of the work in this thesis is to develop an engine that can understand graffiti inputs by using computation intelligence techniques. A graffiti-command interpreter

and a graffiti-to-text translator will be developed. Two methods are proposed to realize these applications: a self-structured neural network and a self-structured neural-fuzzy network.

In a typical fixed-structure neural network, all the nodes are fully connected. For a given application, some links could be redundant that might create unnecessary disturbances to the system. We propose a self-structured neural network, which is developed from a traditional 3-layer feed-forward neural network. Switches are added into the links between the layers of nodes. The presence of these switches enables not only the network parameters but also the network structure to be tuned during the learning process. By turning off some of the link-switches, the number of links can be reduced and the network structure can be reduced after the training process. The proposed self-structured neural network will be applied to interpret three graffiti commands in an eBook reading device.

As the second method, the idea of link-switches is extended to neural-fuzzy networks. By integrating fuzzy sets into the neural networks, expert knowledge can be incorporated into the system. The inter-layer link-switches are employed so as to allow the choices of fuzzy rules during the training process. The proposed neural-fuzzy network is applied to recognize three graffiti commands and ten graffiti digits in an eBook reading device.

1.2 Neural Network for Speech Recognition

Speech is a natural communication tool for human. By using speech, users can give any commands to an eBook reading device, or input texts to the eBook content. We simply need a microphone to sense the signals for speech recognition. However, various speeches are difficult to distinguish, especially when the environment is noisy. A high accuracy for recognizing speech is often a great challenge to the computers. In

this thesis, four approaches are proposed for two speech recognition applications in the eBook environment. The two applications are a speech-to-command recognition system and a speech-to-text input system.

The first approach applies the aforementioned self-structured neural-fuzzy network to recognize Cantonese speech in order to verify that the proposed network can also be applied to other kinds of classification problems. Three Cantonese-command speeches will be recognized by the eBook reading device.

The second approach reconstructs a traditional neural-fuzzy network to a specific structure. This structure involves two neural-fuzzy networks, namely a tuner neural-fuzzy network and a classifier neural-fuzzy network. The tuner neural-fuzzy network is treated as the associative memory of the recognizer. It is employed to provide expert knowledge based on the input patterns to the classifier neural-fuzzy network. On using this proposed network, ten Cantonese-digit speeches can be recognized with fewer network parameters and better performance than the traditional approaches. Results will be given to demonstrate the merits of the proposed method.

The third approach uses two neural networks, instead of neural-fuzzy networks, interconnected together to provide associative memory for classification. This proposed network is evaluated by using it to interpret five speech commands in the eBook environment.

The above three methods are static pattern recognition approach for the mono-syllabic speech recognition problem. To provide a solution to the multi-syllable speech recognition problem, a dynamic pattern recognition approach implemented by a recurrent neural network is also proposed. The recurrent neural network is designed to model the sequential properties of the speech with multi-syllable. It is used to implement the speech-to-text input of the eBook reading device. The proposed network also comprises two sub-networks: a tuner neural network and a classifier

recurrent neural network. The tuner neural network (associative memory) provides the classifier recurrent neural network with dedicated information from the features of the input patterns. The classifier recurrent neural network is employed to handle the information variation in time domain. Owing to the associative memory in the dynamic network structure, the sequence of syllables can be determined while the size of the network can be kept small. Results will be given to show the ability of the proposed dynamic network in recognizing sequential speech (speech with multiple syllables) inputs.

1.3 Electronic Book

Books are the teaching and learning media in our daily life. Traditional books give us the easiest way of getting knowledge. With wonderful pictures and well-written texts, our learning environment can be made more interesting. Besides that, taking notes and adding bookmarks in a book can help the learning process. However, the functions mentioned above are available in a static environment. The reading materials and the functions that we can offer through books are very limited. In order to enrich the reading materials and the learning environment, traditional books are digitalized and enhanced with the state-of-the-art technologies. Tablet PCs, Personal Digital Assistants (PDA), or dedicated electronic book (eBook) reading devices are widely applied as new platforms for reading. Thanks to the digital format, more functions can be implemented inside the eBook environment. For instance, animated pictures, movies, music and speech can be added to provide multimedia and interactive effects in the eBook contents. Also, some useful functions such as annotations, bookmarks, highlights, underlines, and free-sketches can all be made available in the eBook reading device. In particular, graffiti and speech are two important inputs that can be interpreted by the eBook reading device in order to

enhance the convenience to the users.

1.4 Outline of the Thesis

This thesis is organized into six chapters. Chapter 2 describes the functions of an eBook reading device. A literature review on neural networks for classification will be given. These techniques cover the areas of fuzzy logic, neural network, and evolutionary computation. A brief account to pattern recognition (handwriting recognition) and speech recognition systems will also be provided in this chapter.

Chapter 3 presents two proposed approaches for graffiti commands and digits interpretation. The operating principles of the two approaches will be explained. A summary of their performance will be given and comparisons will be made with respect to the traditional approaches.

Chapter 4 describes the Cantonese speech recognition system for the eBook environment. The four approaches to tackling mono-syllabic and multi-syllabic speech recognition applications will be described. A summary on the performance of the proposed approaches will be given. Comparisons between the proposed approaches and the traditional approaches will be made in order to show the merits of the proposed approaches.

Chapter 5 reports the results on implementing the proposed approaches to the eBook applications. Based on the results, the merits of the proposed approaches will be verified. Moreover, comparisons will be made to demonstrate the improvements over the conventional methodologies.

Chapter 6 gives a conclusion to the thesis. The directions for further development of the speech recognition system will also be discussed.

CHAPTER 2

LITERATURE REVIEW

In this chapter, a review on neural network for classification, pattern recognition, Cantonese speech recognition and electronic book (eBook) will be given. An account on the techniques applied for doing pattern recognition (handwriting recognition) and speech recognition, and a brief description to the inadequacy, will be given.

2.1 Neural Network for Classification

Neural network for classification has been a hot research topic for years. By applying neural networks, machines can be made more humanized. At present, it is generally accepted that neural network is one of the three areas under the umbrella of Computational Intelligence. The other two areas are fuzzy logic and evolutionary computation. Fuzzy logic facilitates neural networks with the power of reasoning with respect to linguistic rules and expert knowledge. Evolutionary computation techniques facilitate neural networks with the power of learning based on input-output data.

2.1.1 Fuzzy Logic

Fuzzy sets were proposed by Zadeh [23] and Gogien [17] in 1965. Since then, a wide range of applications has been reported. Fuzzy algebra, fuzzy subset, fuzzy

reasoning, fuzzy inference, fuzzy relation and equation, fuzzy number, fuzzy computing theory [9, 25, 28, 32] etc. are some areas under the umbrella of fuzzy logic. The main contribution of fuzzy logic is that it offers a paradigm for representing and processing linguistic or non-numeric information. It is a logic system that is much closer in spirit to human thinking and natural language than the traditional logic systems [5]. Human problems that involve linguistic or symbolic information can readily be tackled by fuzzy logic.

2.1.2 Neural Network Topologies

Neural Network is a computational tool to mimic the biological information processing mechanism. They are typically designed to perform nonlinear mapping between inputs and outputs. Neural networks are data driven self-adaptive systems that can adjust themselves to the data without any explicit specification of functional or distributional form for the underlying problem. They have been proved to be universal approximators that can estimate any nonlinear smooth function with an arbitrary accuracy [12].

The most important aspects of neural networks are learning and generalization. A typical method of realizing the learning of a neural network is to adjust the weights governing the connections between the processing elements (nodes). According to different weight sets, a neural network can be used to model different non-linear functions. In general, neural networks are advantageous over conventional approaches thanks to their generalization capability, parallelism, distributed memory, redundancy and learning. Thanks to the generalization capability, a neural network can predict the closest output based on the trained parameters and the input. However, the goodness of the network may sometimes be affected by the structural design of the network.

Many kinds of neural network topologies have been studied. Some of the popular topologies are back-propagation feed-forward networks by Rumelhart *et al* [8], self-organizing maps by Kohonen [39], and Hopfield nets by Hopfield [19]. By using different network topologies, different applications are able to achieve better performance.

2.1.3 Evolutionary Computation (EC)

Evolutionary computation (EC) refers to a collection of algorithms that work towards the solution of a certain problem based on evolution of a population. For those problems of optimizing certain multi-dimensional functions, EC is one of the excellent tools. At least three types of evolutionary computing techniques have been reported: Genetic Algorithm (GA) [42], Genetic Programming (GP) and Evolutionary Algorithm (EA). GA is a powerful random search technique that aims to handle optimization problems [10, 18, 42]. This is especially useful for complex optimization problems with a large number of parameters that make the global analytical solutions difficult to obtain. It has been widely applied in different areas such as fuzzy control [2, 6, 21, 41], path planning [14], greenhouse climate control [31], modeling and classification [27, 29, 38], etc.

2.2 Pattern Recognition (Handwriting Recognition)

Pattern recognition is a way for machines to learn the properties of an object. However, machines are only good at numerical manipulation, while the interpretation of objects could be a symbolic manipulation process. A way to convert a symbolic manipulation process to a numerical manipulation process should be found. Different methods of pattern (handwritten character) recognition can be found in the literature. In general, handwriting recognition involves 2

categories, the global approach and the analytical approach [14]. The handwritten character recognition involves segmentation, feature extraction and classification. Segmentation and feature extraction are the preprocessing for the recognition. Segmentation is to partition the connecting handwritten digits/letters into isolated units. Then feature extraction techniques, such as thinning, skeletonisation and contour extraction [7] are employed to obtain the feature vectors for the classifier. Some researchers had developed various methods for recognizing handwritten numerals [4, 7, 30, 35].

For the classification process, four different approaches [24] can be used: template matching, statistical techniques, structural techniques, and neural/neural-fuzzy networks (NN/NFN). The idea of the template matching approach is to determine the best match between the stored templates and the input template. Statistical techniques employ statistical decision theory to determine the class to which the input belongs. Hidden Markov modeling (HMM) is one of the popular statistical techniques for handwritten character recognition [7]. The sequential information were retrieved from the handwritten image and then modeled by the HMM. With a different state sequence for each handwritten pattern, classification can be done by matching the sequence properties of the testing pattern to the training templates.

On applying structural techniques, some complex patterns can be represented by some simpler patterns. Based on these simpler patterns, the input can be classified. Examples of structural techniques include grammatical [23] and graphical [17] methods. The general idea of the neural/neural-fuzzy network approaches [9] is to learn the features of the training patterns obtained from the pre-processing. The inputs with similar features can then be recognized using the trained neural/neural-fuzzy network.

One problem of using conventional NN/NFNs as the handwritten classifier is that the network structure is not necessarily optimal. As mentioned before, a fully-connected 3-layer NN/NFN is only a universal approximator for any smooth non-linear functions. For an unknown function, we are not sure whether the number of nodes is adequate and a fully-connected structure is good. Sometimes, the optimal structure might even changes with respect to the input patterns. Hence, a large fixed-structure NN/NFN might have some of its links and parameters being dummies that introduce unnecessary disturbances to the network system. Moreover, owing to the large network size, more computational power has to be used.

2.3 Speech Recognition

Speech can be modeled as energy functions of frequency. With a given frequency and energy combination, a tone can be produced. Audible speech can be regarded as mixtures of different tone sequences. In order to let machines understand speech, speech recognition techniques have to be applied. In practice, speech recognition involves the transformation of acoustic speech signals into numeric values for machine learning.

2.3.1 Cantonese Speech Recognition

Cantonese speech is a chain of mono-syllabic sound [37]. Each Cantonese-character speech is a combined tone and syllable unit. There are nine similar tones for a Cantonese syllable. Some Cantonese characters share the same vowel, e.g. the Cantonese character of “1” (/jat1/) and the Cantonese character of “7” (/cat1/) share the same vowel of /a/ [43]. Thus, it is a difficult task to recognize Cantonese characters, which requires not only the accurate discrimination of

characters with different syllable but also the different tones of the same syllable.

Speech recognition [24] involves two steps: speech preprocessing and classification. The preprocessing stage involves segmentation and feature extraction. Segmentation is used to define the boundary of the temporal speech segments which represents the basic phonetic components of a speech. Then, the stationary properties of the individual segment can be modeled by static classification approaches. The dynamic properties of the temporal segments, on the other hand, can be modeled by using the dynamic classification approaches.

Feature extraction is a technique to find a good representation to a speech signal. Normally, the time-domain speech signals will be windowed into speech frames. For each speech frame, Fast Fourier Transform is applied to obtain the frequency spectrum. Based on the frequency spectrum, digital signal analysis techniques can be applied to obtain the cepstral coefficients, which describe the features of the speech. Filter-bank analysis [24], which is a technique for modeling the human ears, is used to analyze the speech features. By distributing different band-pass filters in the mel-scale of frequency, which models the characteristics of the human ears, the frames of speech feature coefficients can be obtained. Using the feature coefficients, we can perform the second step of classification in order to recognize a speech.

Speech classification techniques can be categorized into 3 types: template matching, acoustic-phonetic recognition and stochastic processing. For the template matching technique, reference speech units called templates are used to perform the matching process between the testing speech units and the templates. Thus, the testing speech units that closely match with the reference templates can be identified. The acoustic-phonetic recognition is a phoneme level speech recognition approach. By using this approach, the acoustic similarity among the

phonemes combination in a speech will be used to identify the input speech. The stochastic process for speech recognition is similar to the template matching process that requires the reference speech units for identifying the input speech. The main difference between the stochastic process and the template matching process is that the stochastic process performs a statistical and probabilistic analysis matching process, which is not a direct mapping to the reference templates.

Two popular Cantonese speech recognition techniques are those using the Hidden Markov Model (HMM) and Neural Networks (NN) [20, 24, 36]. HMM is one well-known stochastic processing technique for speech recognition [17]. It is a statistical state-sequence recognizing approach. The states in an HMM structure represent the stochastic process, and the directional links connected between states are the transitions which indicate the flow from one state to another. With the different states-and-transitions structure of each speech, recognition can be realized by matching the testing speech unit to the reference models. The Baum-Welch maximum-likelihood algorithm can be employed to compare the probability score between the testing and reference models as the likelihood index. Another verification process for HMM speech recognition can be done by the Viterbi algorithm, which is a method to determine if the nodes and sequence of the testing speech unit is closely matched with the reference models.

By using the connectionist modeling technique, non-linear functions can be modeled by neural networks (NNs). Thanks to the capability of being a universal approximator through training, an NN can be trained to become a classifier for a certain input speech patterns. We can adopt a static pattern classification method or a dynamic pattern classification method. The static pattern classification method uses a conventional 3-layer feed-forward neural network to model each single-character Cantonese speech. By using a conventional feed-forward NN as

the speech classifier, the static properties of the speech frames can be analyzed. This conventional feed-forward NN's input vector contains no acoustic feature, and it is only suitable for recognizing speech with a single syllable.

When the dynamic properties between speech frames are important for recognizing the speech, we should consider using recurrent neural networks (RNNs) as the classifiers, and learn the relationships between speech frames through the training process. A recurrent neural network (RNN) is a network which is commonly used to tackle complex dynamic sequential problems, such as time series prediction, dynamic system identification, and grammatical inference. Thanks to the specific structure of a recurrent network, the temporal information can be brought to the next time frame. As a result, the network not only models the static information but also the time-varying information. Elman network, Jordan network [1], etc. are commonly used RNN structures. They are constructed as a closed-loop system where the hidden-node outputs or the network outputs are fed back to the network inputs. In practice, an RNN is suitable to classify speech with multiple syllables. However, an RNN consumes more computing power than a feed-forward NN. Owing to the recurrent information fed back, RNN needs a large number of network parameters so as to cope with the speech frame updates and the recurrent information updates for each speech. As the number of parameters is large, a longer training time is needed for an RNN to converge.

It should be noted that one neural network, feed-forward or recurrent, is needed to effectively model one speech. If we have to recognize a large number of recognition vocabularies, a standard structured NN may be difficult to achieve a good performance. It is because the trained network parameters are used to model the features of the input patterns. When the number of input patterns is large and they are not closely clustered into the same class, more network parameters are

needed to provide a fine resolution for modeling them. A complicated network structure will then result. This will require higher computational power and decrease the convergence rate.

2.4 Electronic Book (eBook)

Books are the major media for knowledge exchange. Typically, we gain information from the texts and pictures inside books. However, with the growth of technologies, people prefer multimedia contents so that information can be presented in a vivid, efficient and interactive manner. Electronic devices such as Personal Digital Assistants (PDA), Notebook Computers, and Tablet PCs are developing rapidly. These devices take advantage of the computational power to display multimedia materials. However, these general-purpose devices are not dedicated for manipulating book contents, and some of the device features might not be needed by eBook applications. In order to reduce the size and cost, dedicated electronic book reading software has been developed such as Microsoft Reader [45] and Acrobat Reader [44]. However, this software has no readily integrated graffiti/speech recognition tools offered to help the interactions with the readers.

In a dedicated electronic book reading device, our usual actions made in the traditional book environment, such as highlighting, book-marking, free-sketching, and annotating all become build-in device features. The results offered by these features can easily be changed or removed. In addition, features like multimedia book contents, page magnification, content searching, text reading, etc. that cannot be done by ordinary books can now easily be realized in an eBook reading device [44, 45].

To further enhance the features of an eBook reading device, computational intelligence techniques can be applied so that the device can become more

user-friendly and intelligent. In this thesis, we look at an eBook reading device as a platform to realize the proposed neural network techniques. They will be employed to tackle the problems of pattern recognition (graffiti input interpretation by the eBook) and speech recognition (Cantonese input speech recognition by the eBook).

CHAPTER 3

NEURAL NETWORK FOR GRAFFITI

INTERPRETATION

3.1 Introduction

In this thesis, graffiti refer to sketches made by a stylus on the screen of an eBook reading device. Thanks to the touch-screen input device, users' sketches can be transformed into numeric data corresponding to the coordinates of the sketched trajectory. Taking the graffiti symbol as the input of the eBook reader, we can develop a recognizer to interpret graffiti as some user-defined meanings. Neural network techniques are proposed to realize graffiti command and graffiti text recognition, which are to be implemented in an eBook environment.

This chapter reports two approaches for graffiti interpretation. One uses a self-structured neural network, and the other uses a self-structured neural-fuzzy network. The two proposed approaches are developed based on the idea of the pruning techniques [11, 35]. The main difference between the two proposed approaches to the well-known pruning techniques is that the deduced links/nodes for the pruning techniques can not be reborn. However, in the proposed approaches, the inter-node link switches can be re-opened/re-closed during the training process.

A self-structure neural network is defined as a network with inter-node link switches with trainable on/off states. The first approach is based on a new neural network structure, which employs a switch in each link between the layers of a

conventional 3-layer neural network. The states (on or off) of the link switches are tuned by using the reference patterns during the training process. In effect, a traditional 3-layer feed-forward neural network is modified into a network with a tunable structure. Using this approach, a smaller network structure than a 3-layer feed-forward neural network with fixed connections can be obtained.

The second approach is a modification to the first approach. Fuzzy rules are incorporated into the neural network so that expert knowledge can be used to enhance the learning ability of the resulting neural-fuzzy network. Besides that, in order to obtain the most useful fuzzy rules, rule switches are introduced to the network. Hence, the size of the network and the number of rules used in the network can be reduced after the training process.

This chapter is organized as follows. A detailed description to the network structures and the training aspects of the two proposed approaches will be provided in Sections 3.2 and 3.3 respectively. In Section 3.4, the performance of the two approaches, and the factors of considerations for choosing the network will be discussed. A chapter summary will be given in the final section of this chapter.

3.2 Self-structured Neural Network

As a result of its specific structure, a neural network can be used to realize a learning process [31]. In general, learning is carried out in two steps: 1) a network structure is defined, and 2) an algorithm is chosen for realizing the learning process. Usually, the structure of a neural network is fixed for a learning process. However, this fixed structure may not provide the best performance within a given training time for different training patterns. If the neural network structure is too complicated, the training time will be long. In this section, a neural network with link switches is presented. By introducing a switch to each link, not only the

parameters but also the structure of the neural network can be tuned. The proposed neural network will be employed to interpret graffiti commands for eBooks.

3.2.1 Network Structure

The proposed three-layer network is shown in Fig. 3-1.

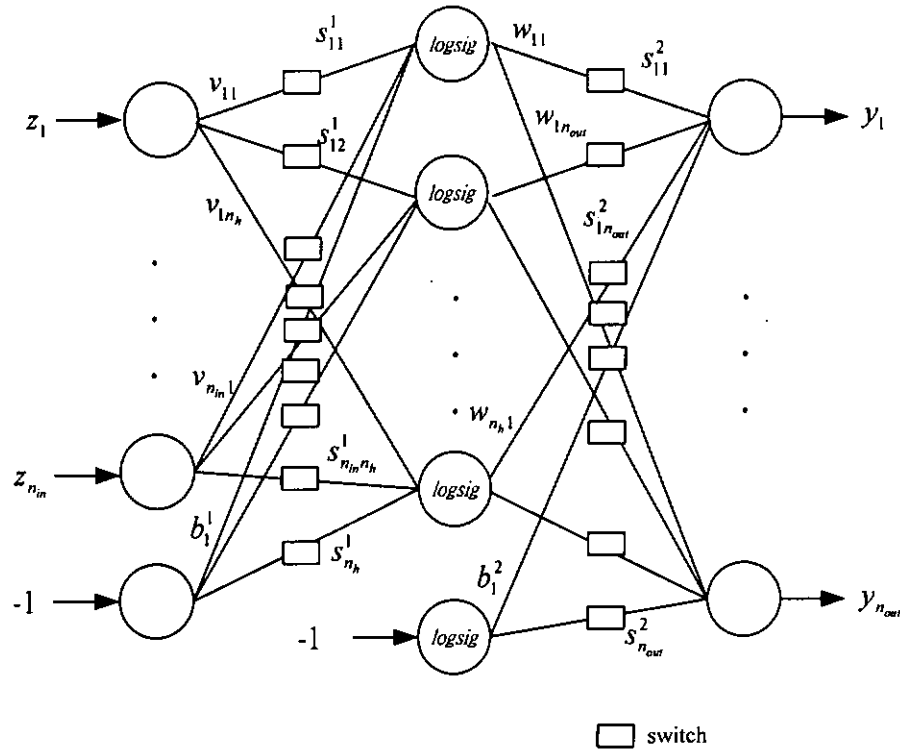


Fig. 3-1. Proposed 3-layer neural network with switches.

The important point is that a unit-step function is introduced to each link. Such a unit-step function is defined as,

$$\delta(\alpha) = \begin{cases} 0 & \text{if } \alpha < 0 \\ 1 & \text{if } \alpha \geq 0 \end{cases}, \quad \alpha \in \mathbb{R}. \quad (3.1)$$

This is equivalent to adding a switch to each link of the neural network. Referring to Fig. 3-1, the input-output relationship of the proposed

multiple-input-multiple-output three-layer neural network is as follows,

$$y_k(t) = \sum_{j=1}^{n_h} \delta(s_{jk}^2) w_{jk} \text{logsig} \left[\sum_{i=1}^{n_{in}} (\delta(s_{ij}^1) v_{ij} z_i(t)) - \delta(s_j^1) b_j^1 \right] - \delta(s_k^2) b_k^2, \\ k = 1, 2, \dots, n_{out}. \quad (3.2)$$

$z_i(t)$, $i = 1, 2, \dots, n_{in}$, are the inputs which are functions of a variable t ; n_{in} denotes the number of inputs; v_{ij} , $i = 1, 2, \dots, n_{in}$; $j = 1, 2, \dots, n_h$, denotes the weight of the link between the i -th input and the j -th hidden node, n_h denotes the number of hidden nodes (excluding the bias node); s_{ij}^1 , $i = 1, 2, \dots, n_{in}$; $j = 1, 2, \dots, n_h$, denotes the parameter of the link switch from the i -th input to the j -th hidden node. w_{jk} , $j = 1, 2, \dots, n_h$; $k = 1, 2, \dots, n_{out}$, denotes the weight of the link between the j -th hidden node and the k -th output. s_{jk}^2 , $j = 1, 2, \dots, n_h$; $k = 1, 2, \dots, n_{out}$, denotes the parameter of the link switch from the j -th hidden node to the k -th output, n_{out} denotes the number of outputs of the proposed neural network. b_j^1 and b_k^2 denote the biases for the hidden nodes and output nodes respectively. s_j^1 and s_k^2 denote the parameters of the link switches of the biases to the hidden and output layers respectively. $\text{logsig}(\cdot)$ is the logarithmic sigmoid function defined as follows,

$$\text{logsig}(\alpha) = \frac{1}{1 + e^{-\alpha}}, \quad \alpha \in \mathbb{R}. \quad (3.3)$$

$y_k(t)$, $k = 1, 2, \dots, n_{out}$, is the k -th output of the proposed neural network. Because of the presence the switches, not only the weights but also the switch states

can be tuned. It can be seen that the weights of the links govern the input-output relationship of the neural network while the switches of the links govern the structure of the neural network.

3.2.2 Training

The proposed neural network can be employed to learn the input-output relationship of an application using some optimization algorithm such as genetic algorithm (GA). Because of the presence of switches in the inter-node links, the neural network is effectively a discontinuous system during its training phase. Hence, the derivatives of the network function cannot be calculated. GA is one of the appropriate training methods for this type of neural network systems. In this thesis, all neural networks are tuned by an improved GA, which is detailed in the Appendix. For the training purpose, the input-output relationship of the proposed neural network can be described by,

$$\mathbf{y}^d(t) = \mathbf{g}(\mathbf{z}^d(t)), t = 1, 2, \dots, n_d \quad (3.4)$$

where $\mathbf{y}^d(t) = [y_1^d(t) \ y_2^d(t) \ \dots \ y_{n_{out}}^d(t)]$ and $\mathbf{z}^d(t) = [z_1^d(t) \ z_2^d(t) \ \dots \ z_{n_{in}}^d(t)]$ are the desired outputs and given inputs respectively of an unknown nonlinear function $\mathbf{g}(\cdot)$, n_d denotes the number of input-output data pairs. The fitness function used by the GA can be defined as,

$$fitness = \frac{1}{1 + err}, \quad (3.5)$$

$$err = \sum_{k=1}^{n_{out}} \frac{\sum_{t=1}^{n_d} |y_k^d(t) - y_k(t)|}{n_d \times n_{out}}. \quad (3.6)$$

The objective is to maximize the fitness value of (3.5) using the improved GA by setting the chromosome to be $[s_{jk}^2 \ w_{jk} \ s_{ij}^1 \ v_{ij} \ s_j^1 \ b_j^1 \ s_k^2 \ b_k^2]$ for all i, j, k . It can be seen from (3.5) and (3.6) that a larger fitness value implies a smaller error value.

3.3 Self-structured Neural-fuzzy Network

Expert knowledge and experience in the form of fuzzy rules can be incorporated into a neural network to realize a neural-fuzzy network (NFN). Like a neural network, an NFN can be used to realize a learning process [31]. Similar to the argument in the former section, a fixed-structured NFN may not provide the best performance within a given iteration number. If the structure of the NFN is too complicated and the training patterns are not sufficient, the number of iteration, or the number of floating point operations, will be large during operation. In this light, an NFN that allows both tuning the membership functions and finding the number of rules is proposed. The number of rules can be found by introducing switches for activating the fuzzy rules, which can be realized as switches in some links of the NFN.

3.3.1 Network Structure

A fuzzy associative memory (FAM) [10] is used as the rule base of the NFN. The FAM is formed by partitioning the universe of discourse of each fuzzy variable according to the level of fuzzy resolution chosen for the antecedents. A grid of FAM elements is then generated. The entry at each grid element in the FAM corresponds to a fuzzy premise. Hence, an FAM can be interpreted as a geometric or tabular representation of a fuzzy logic rule base. As discussed previously, the

number of possible rules in an NFN might be too large. Thus, a multi-input-multi-output NFN that can have a selectable number of rules and membership functions is proposed. The main difference between the proposed network and the traditional network is that the unit step function of (3.1) is introduced for activating each rule. The structure of the proposed NFN is shown in Fig. 3-2. Referring to this figure, the input and output variables are defined as x_i and y_j respectively; where $i = 1, 2, \dots, n_{in}$ and n_{in} is the number of input variables, $j = 1, 2, \dots, n_{out}$ and n_{out} is the number of output variables. The behavior of the NFN is governed by p fuzzy rules of the following format:

$$\begin{aligned}
 R_g: \quad & \text{IF } x_1(t) \text{ is } A_{1g_1}(x_1(t)) \text{ AND } x_2(t) \text{ is } A_{2g_2}(x_2(t)) \text{ AND } \dots \text{ AND } x_{n_{in}}(t) \\
 & \text{is } A_{n_{in}g_{n_{in}}}(x_{n_{in}}(t)) \\
 & \text{THEN } y_j(t) \text{ is } w_{gj}, t = 1, 2, \dots, u
 \end{aligned} \tag{3.7}$$

where u denotes the number of input-output data pairs; $g = 1, 2, \dots, p$, is the rule number; w_{gj} is the output singleton of rule g for the j -th output. From Fig. 3-2,

$$p = \prod_{i=1}^{n_{in}} m_i \tag{3.8}$$

where m_i is the number of membership functions of input variable x_i , and in (3.7), $g_i \in [1, \dots, m_i]$, $i = 1, \dots, n_{in}$. In this network, the membership function is a bell-shaped function given by,

$$A_{i g_i}(x_i(t)) = e^{\frac{-(x_i(t) - \bar{x}_{ig_i})^2}{2\sigma_{ig_i}^2}} \tag{3.9}$$

where the parameters \bar{x}_{ig_i} and σ_{ig_i} are the mean value and the standard deviation of the membership function respectively. The grade of the membership of each rule is defined as,

$$\mu_g(t) = A_{1g_1}(x_1(t)) \cdot A_{2g_2}(x_2(t)) \cdot \dots \cdot A_{n_{in}g_{n_{in}}}(x_{n_{in}}(t)) \quad (3.10)$$

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

$$y_j(t) = \frac{\sum_{g=1}^p \mu_g(t) w_{gj} \delta(\varsigma_{gj})}{\sum_{g=1}^p \mu_g(t)} \quad (3.11)$$

where ς_{gj} denotes the rule switch parameter of the g -th rule for the j -th output.

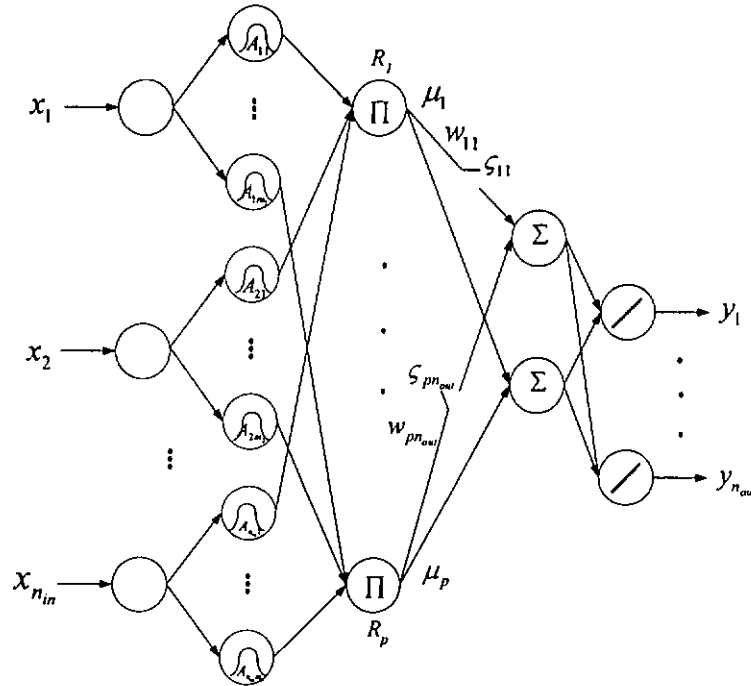


Fig. 3-2. Proposed neural-fuzzy network.

3.3.2 Training and Rule Switches Tuning

The proposed NFN can be employed to learn the input-output relationship of an application by using the improved GA. The desired input-output relationship can be described by,

$$\mathbf{y}^d(t) = \mathbf{g}(\mathbf{z}^d(t)), t = 1, 2, \dots, u \quad (3.12)$$

where $\mathbf{y}^d(t) = [y_1^d(t) \ y_2^d(t) \ \dots \ y_{n_{out}}^d(t)]$ is the desired output corresponding to the input vector $\mathbf{z}^d(t) = [z_1^d(t) \ z_2^d(t) \ \dots \ z_{n_{in}}^d(t)]$, and $\mathbf{g}(\cdot)$ is an unknown non-linear function. The fitness function used by the GA can be defined by (3.5), where

$$err = \sum_{j=1}^{n_{out}} \frac{\sum_{t=1}^u |y_j^d(t) - y_j(t)|}{u \times n_{out}}. \quad (3.13)$$

The objective is to minimize the value of (3.13) using the improved GA by setting the chromosome to be $[\bar{x}_{ig_i}, \sigma_{ig_i}, \varsigma_{gj}, w_{gj}]$ for all i, j, g_i, g_j . The value of (3.13) is the mean absolute error (MAE). The range of *fitness* in (3.5) is $[0, 1]$. A larger value of *fitness* indicates a smaller *err*. Owing to the presence of switches for the fuzzy rules, an NFN with selectable number of rules and membership functions can be achieved.

3.3.3 Graffiti Interpreter

As shown in Fig. 3-3, a multi-NFN system is used to realize the graffiti interpreter. Each set of graffiti training samples is used to train its corresponding NFN. The desired outputs of the NFN are set to be the inputs of the NFN. During the interpretation operation, the sampled points of the input graffiti will be

fed to all the m NFNs. The outputs of the m NFNs are fed to the graffiti determiner, which measures the similarity between the input graffiti and the outputs of the NFNs, in order to identify the possible graffiti input. The similarity of the input graffiti to the output of an NFN is defined as,

$$S_i = \|\bar{y}_i - \bar{z}\|, i = 1, 2, \dots, m \quad (3.14)$$

where

$$\bar{y}_i = \frac{\mathbf{y}_i}{\|\mathbf{y}_i\|} = [\bar{y}_{i1}(t) \quad \bar{y}_{i2}(t) \quad \dots \quad \bar{y}_{i n_m}(t)] , i = 1, 2, \dots, m, \quad (3.15)$$

$$\bar{z} = \frac{\mathbf{z}}{\|\mathbf{z}\|} = [\bar{z}_1(t) \quad \bar{z}_2(t) \quad \dots \quad \bar{z}_{n_m}(t)] , \quad (3.16)$$

\bar{y}_i and \bar{z} denote the normalized outputs and the normalized input of the NFNs respectively. A smaller value of S_i implies a closer match of the input graffiti to the graffiti represented by the i -th NFN. The smallest similarity value among the m NFNs is defined as,

$$S_j \equiv \min_i S_i \quad (3.17)$$

The index j of (3.17) is the output of the graffiti determiner, which indicates the j -th graffiti is the most likely input graffiti.

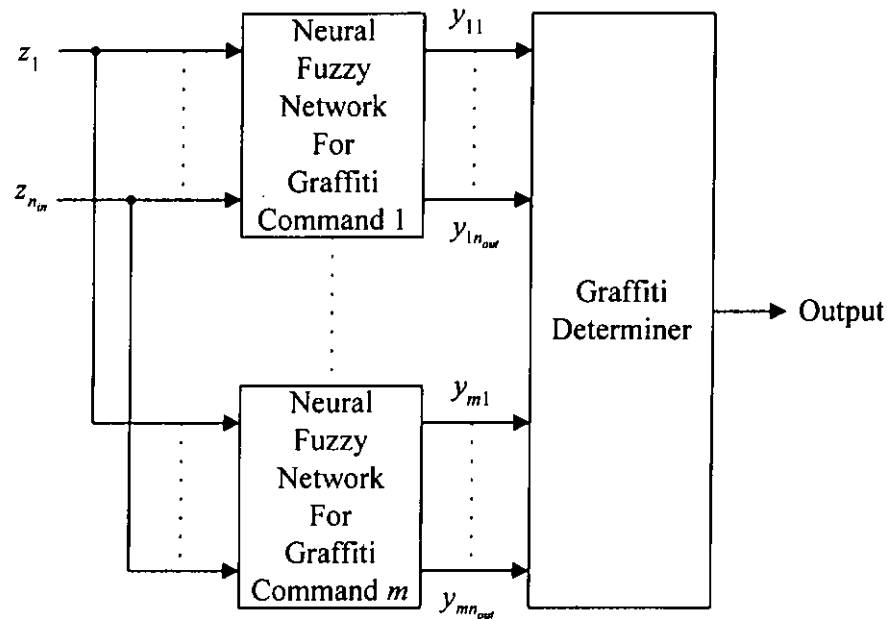


Fig. 3-3. Block diagram of the graffiti interpreter.

3.4 Comparison

The experimental results will be reported in Chapter 5. In this section, brief qualitative comparisons between the traditional NN, the traditional NFN, the self-structure NN, and the self-structured NFN will be made.

The structure of the traditional 3-layer feed-forward fully connected NN and NFN are quite similar. When an application can make use of expert knowledge and experience in terms of fuzzy rules to help making the decision, an NFN can be used to absorb the fuzzy rules into the network for training. In this way, the rules provide the network with an initial framework, and the training process is to fine-tune the weights of the network so as to optimize the system. However, the performance of the NFN depends very much on how well the rules are being set. On the other hand, the performance of an NN depends mainly on the training algorithm. A good global searching algorithm may help the NN to find the global optimum of an application. On comparing the structure of traditional networks to

that of the self-structured networks, the introduction of the link switches or rule switches can reduce the number of parameters and links, and offer a smaller structure than a conventional 3-layer fully connected NN.

Table 3-1 shows the comparison between the four kinds of networks in terms of their number of links/rules, and the factors of consideration for choosing them.

	Self-structured NN	Traditional NN	Self-structured NFN	Traditional NFN
Number of links/rules	Small	Large	Small	Large
Factors of consideration	No expert knowledge is needed		Expert knowledge can be applied	

Table 3-1. Comparisons between the four neural networks.

Comparing the two proposed self-structured NN and NFN, they have their own strength in different applications. Table 3-2 gives a summary of the comparison between them from the application of recognising 16 graffiti patterns. (The details about the application will be given in section 5.3.4.) For the self-structured NFN, increasing the number of membership functions to model the application should increase the learning ability of the network. Some of the rule(s) can possibly be removed through training, and expert knowledge can be applied by fixing the link switch of the corresponding fuzzy rules to be always closed during the training. However, the number of parameters of an NFN might inherently be large owing to the large amount of available fuzzy rules. As a result, if some applications require a high recognition rate, and expert knowledge about the applications is available, a self-structured NFN is appropriate if we can afford the potentially large number of parameters. Otherwise, if we do not have any expert knowledge about the application, and we can tolerate the potentially larger error, a self-structured NN can

be a good choice.

	Self-structured NN	Self-structured NFN
Number of parameters	Smaller	Potentially larger
Number of FLOPS	Smaller	Potentially larger
Recognition Performance	Acceptable	Potentially higher

Table 3-2. Comparisons between the proposed self-structured NN and NFN.

3.5 Summary

In this chapter, a self-structured neural network and a self-structured neural-fuzzy network have been proposed to interpret graffiti sketches in an eBook application. By introducing switches to some links of the networks, the structure of the graffiti interpreter can be tuned. Hence, the network after training might not be a fully connected network as some of the link switches are open. In this way, some parameters or fuzzy rules can be removed after the training process. Thanks to the reduction to the structural complexity, the training time of the system may be decreased. This is especially important to the self-structured NFN because reducing the number of fuzzy rules may significantly reduce the complexity of the network.

Comparisons between the traditional NN, the traditional NFN, the self-structured NN, and the self-structured NFN have been made. The number of links/rules and the factors of consideration for choosing these four networks have been reported. Considering the two proposed approaches, the self-structured NN is believed to be suitable for cases that allow slightly larger error with fewer network parameters. The self-structured NFN, on the other hand, is appropriate for the cases that have expert knowledge available and require higher recognition performance, while a larger number of parameters can be accepted.

Experimental results based on the proposed networks will be shown in Chapter 5 of this thesis to demonstrate the merits of the proposed approaches for graffiti interpretation.

CHAPTER 4

NEURAL NETWORK FOR SPEECH RECOGNITION

4.1 Introduction

This chapter presents four neural network approaches for Cantonese-digit/Cantonese-command speech recognition in an eBook environment. By using a Cantonese speech recognizer, speech commanding and speech-to-text applications can be realized.

The first approach uses a self-structured neural-fuzzy network (NFN), which has the same architecture as that mentioned in Section 3.3 of Chapter 3. By applying this network to implement the Cantonese-digit/Cantonese-command speech recognition, a better recognition rate than that of a conventional NFN can be achieved under the application data sets provided in this thesis. The second and third approaches improve the network structure by connecting two similar networks together to form a variable-parameter network. The second approach involves two NFNs and the third approach involves two neural networks (NNs). In both approaches, one network serves as an associative memory of the other network. The associative memory is termed the tuner network, which processes the input data to generate the parameters of the second network. The second network is termed the classifier network, which processes the input data with respect to the parameters from the associative memory to do the classification. By connecting the two networks in this manner, the

variable-parameter network effectively has multiple networks that can cope with a wide range of input patterns. The search space of the recognition system can be widened.

To further develop the proposed variable-parameter neural network, the fourth approach uses a recurrent neural network as the classifier network in order to realize dynamic pattern classification for a Cantonese-number speech recognition system. Owing to a recurrent loop, the information obtained from the previous state can be brought to the current state. Thus, both the temporal and dynamic properties of the speech can be modeled. In practice, the sequence of a multi-syllable speech can be tackled by this proposed dynamic variable-parameter neural network.

This chapter is organized as follows. In Section 4.2, the Cantonese-digit/Cantonese-command speech recognition system will be described. In Section 4.3, the network structure, training and classification process of the three proposed static pattern classification for the Cantonese-digit/Cantonese-command speech recognition will be presented. The proposed dynamic pattern classification for the Cantonese-number speech recognition will be discussed in Section 4.4. In Section 4.5, qualitative comparisons between the four proposed approaches in this chapter will be provided. A chapter summary will be given at the end of this chapter.

4.2 Cantonese Speech Recognition System

As shown in Fig. 4-1, the proposed Cantonese speech recognition involves two steps: 1) speech feature extraction (pre-processing) and 2) speech classification. Feature extraction is a process to extract the specific feature coefficients representing each speech sample. Since the speech signal obtained from a recorder is effectively represented as a vector of a very large size, signal processing techniques have to be applied to reduce the size. Several methods are widely used in the feature extraction process, and one of them is the filter-bank analysis. Filter-bank analysis [24] is

developed based on a model of the human ears. By using a bank of band-pass filters, the cepstral coefficients of the speech can be obtained. These coefficients will be the inputs for the classification process. Since every speech has its own set of feature coefficients, classification can be performed by a speech recognition system to determine the meaning of the speech. In this thesis, four speech classification methods are proposed to realize the Cantonese-digit/Cantonese-command speech recognition, and they will be presented in the following sections.

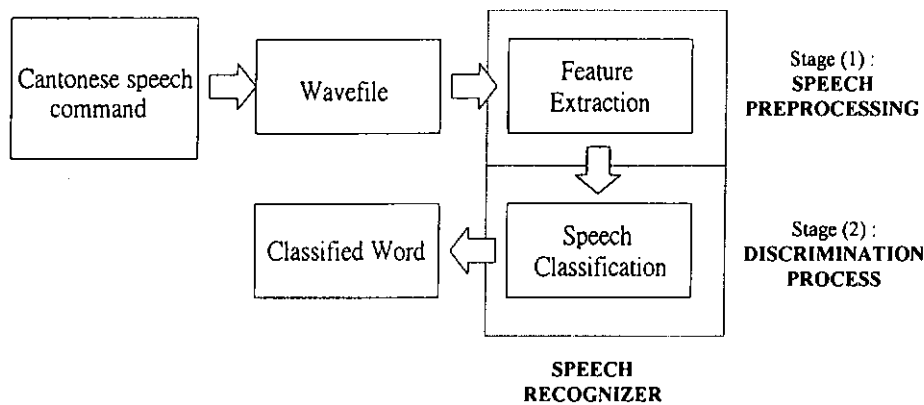


Fig. 4-1. Block diagram of a Cantonese-digit/Cantonese-command speech recognition system.

4.3 Static Pattern Classification for Cantonese Speech Recognition

The classification process in a speech recognition system can be done by a static pattern classification process or a dynamic pattern classification process. Static pattern classification for Cantonese speech recognition refers to classifying temporal speech, single-Cantonese-character speech or mono-syllabic speech. Examples of mono-syllabic Cantonese speech include the Cantonese digits “/ling4/” (0), “/jat1/” (1), “/gau2/” (9), etc. The time-domain speech waveform of a typical single-Cantonese-character speech is shown in Fig. 4-2. Static modeling techniques can be employed to recognize the Cantonese-digit zero to ten. As the capability of

neural networks (NNs) for doing discrimination is high, the classification of Cantonese-digit/Cantonese-command speech using the template matching method can be realized. Three approaches for the static pattern classification for Cantonese-digit/Cantonese-command speech recognition are proposed. Their network structures, training and classification are to be discussed.

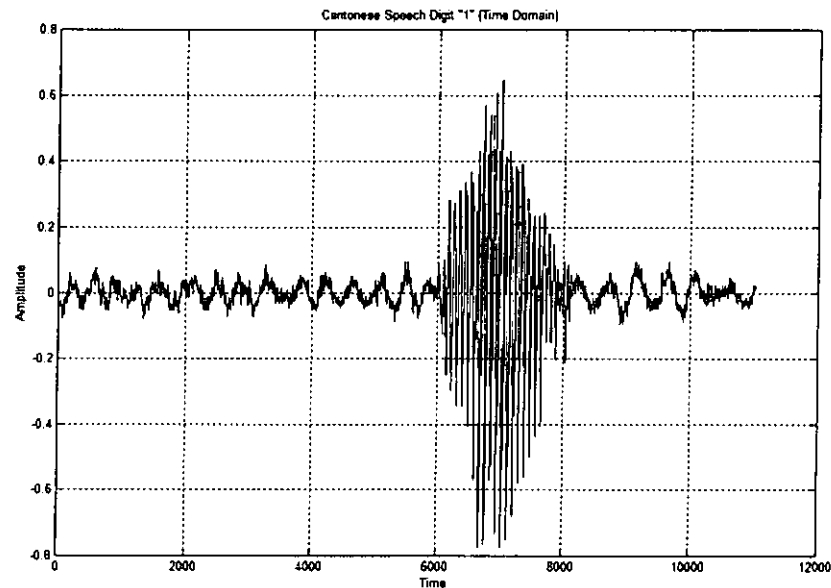


Fig. 4-2. Speech signal of the Cantonese-digit speech, digit “1” .

4.3.1 Self-structured Neural-fuzzy Network

The self-structured NFN for graffiti interpretation in an eBook environment has been discussed in Chapter 3. In order to verify that this network is a good universal approximator, it is also applied to perform the classification process of the Cantonese-digit/Cantonese-command speech recognition in an eBook reading device.

4.3.1.1 Network structure

The self-structured NFN used for Cantonese-digit/Cantonese-command speech recognition has a structure as shown in Fig. 3-2 of Chapter 3. The inputs of the

network are the feature parameters obtained from the feature extraction process.

4.3.2 Variable-parameter Neural-fuzzy Network

The definition of a variable-parameter neural network is a network in which some of the parameters of a network are varied by another network during the operation. A variable-parameter neural-fuzzy network is proposed to recognize Cantonese-digit/Cantonese-command speech. As shown in Fig. 4-3, the structure of the proposed network consists of two NFNs, namely a tuner NFN and a classifier NFN. In general, the parameters of traditional NFNs are fixed after the training. The size of the parameter search space for the training depends solely on the complexity of the network structure. In the proposed NFN, some parameters of the classifier NFN are adjusted by the tuner NFN (which have fixed parameters after training) based on the input data. Thanks to the variable-parameter structure, a relatively large search space that adapts to changing input data can be obtained by a single network.

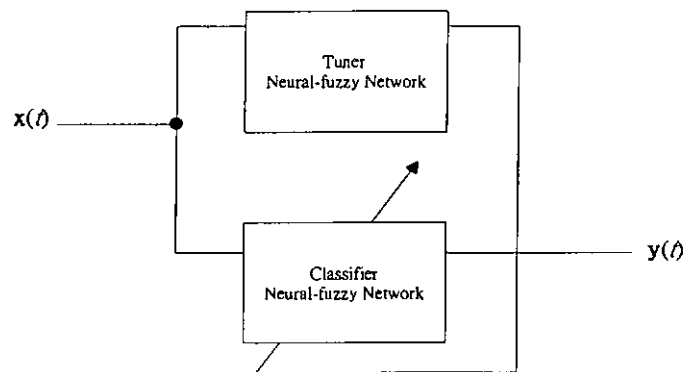


Fig. 4-3. Block diagram of the modified neural-fuzzy network.

We use a fuzzy-associative-memory (FAM) [3, 26, 34] type of rule base for both the tuner and classifier NFNs. As mentioned in Chapter 3, an FAM can be interpreted as a geometric or tabular representation of a fuzzy logic rule base. The structure of the

proposed NFN is shown in Fig. 4-4.

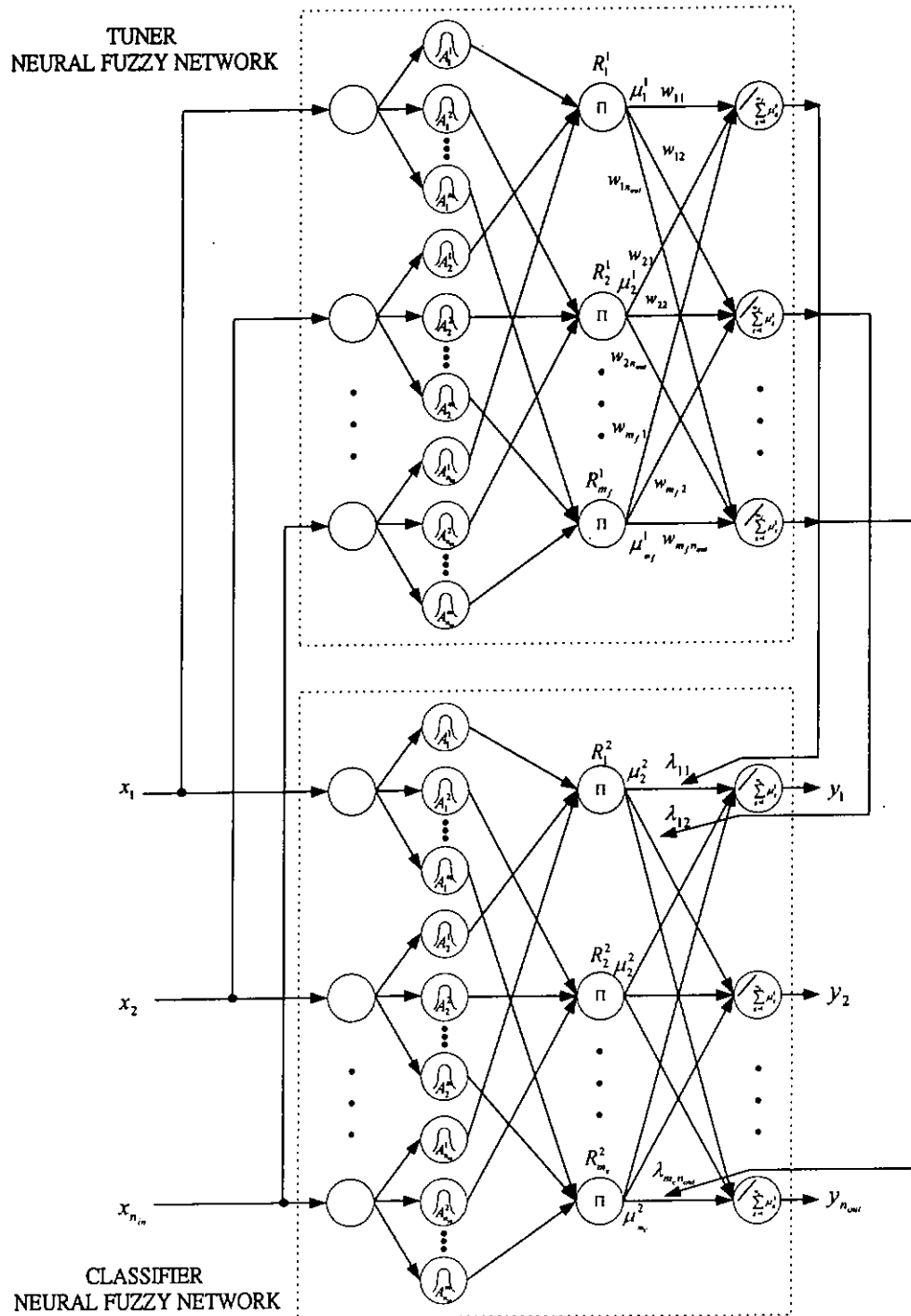


Fig. 4-4. Three-layer neural-fuzzy network.

4.3.2.1 Tuner neural-fuzzy network

The tuner neural-fuzzy network (TNFN) is a traditional partially connected neural-fuzzy network which is used as the associative memory of the system. The parameters of the TNFN are used to store the feature information of each training pattern. Thus, the TNFN parameters are fixed after the training process. The fuzzy rules and the membership function used in the TNFN are given in (3.7) – (3.10) of the previous chapter. The job of the TNFN is to bring the information corresponding to the input patterns into the associative memory to help the classification process. In practice, each input pattern will generate its own set of parameters for the operation of the classifier NFN.

4.3.2.2 Classifier neural-fuzzy network

The classifier neural-fuzzy network (CNFN) analyzes the information of the input pattern with respect to the parameters obtained from the TNFN, and produces the network outputs. From Fig. 4-3, we can see that the structures of both the tuner and the classifier NFNs are the same. The outputs of the CNFN are governed by the following equation.

$$y_j(t) = \frac{\sum_{g=1}^{m_c} \mu_g^2(t) \lambda_{jg}}{\sum_{g=1}^{m_c} \mu_g^2(t)} \quad (4.1)$$

It should be noted that the outputs of the TNFN are the values of the output singletons of all the rules, $\lambda_{jg}; j = 1, 2, \dots, n_{out}; g = 1, 2, \dots, m_c$, of the CNFN. The outputs of the CNFN ($\mathbf{y}(t) = [y_1(t) \ y_2(t) \ \dots \ y_{n_{out}}(t)]$) describes the target class of the input pattern.. The desired value of $\mathbf{y}(t)$ is $\mathbf{y}_d = [a_1 \ a_2 \ \dots \ a_j \ \dots \ a_{n_{out}}]$. Only the

value of a_j for a particular class j is equal to 1, and the rest elements of y_d are all zero.

The rule base of the CNFN will change with respect to the input pattern.

4.3.3 Variable-Parameter Neural Network

Neural networks are good tools for doing classification. For a typical neural network, a fixed set of learning weights are obtained through the training processes. When the training input data set is extracted from a large domain, which is due to the characteristics of an application, a fixed set of network parameters may not be enough to learn the characteristics of the application sufficiently. As a result, the network architecture as shown in Fig. 4-5 is proposed for the Cantonese-digit/Cantonese-command speech recognition. It consists of a tuner neural network (TNN) and a classifier neural network (CNN) connected together.

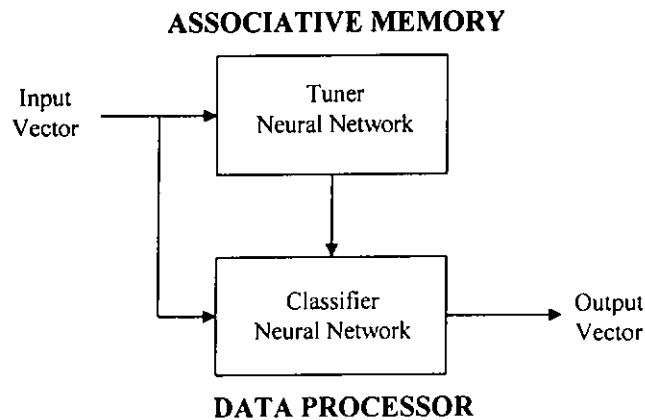


Fig. 4-5. Architecture of the proposed neural network.

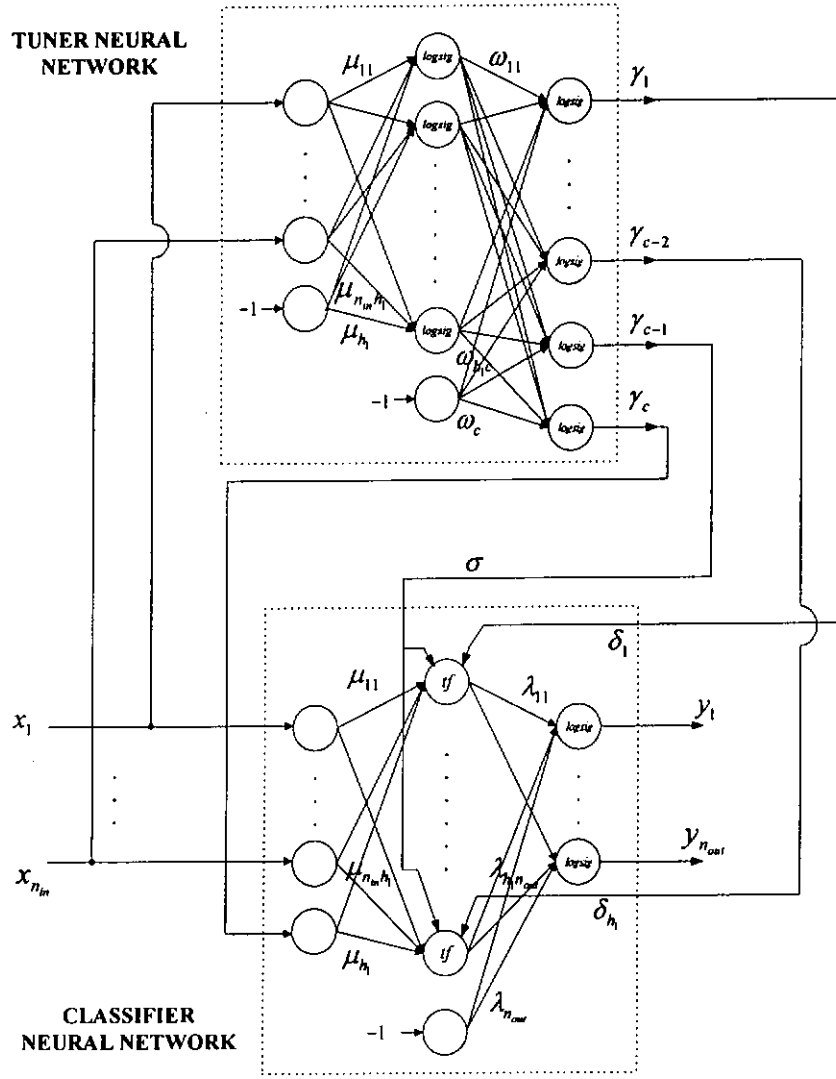


Fig. 4-6. Structure of the proposed neural network.

4.3.3.1 Tuner neural network

As shown in Fig. 4-6, the tuner neural network (TNN) is used as the associative memory, which is implemented by a traditional 3-layer fully-connected feed-forward neural network. It supplies the parameters to the data processor according to each input pattern. Therefore, each input pattern will have its dedicated parameters set in the data processor. The input-output relationship of the TNN is defined as follows,

$$\gamma_g(t) = \text{logsig} \left[\sum_{j=1}^{h_1} \omega_{jg} \text{logsig} \left(\sum_{i=1}^{n_{in}} \mu_{ij} x_i(t) - \mu_j \right) - \omega_g \right], g = 1, 2, \dots, c \quad (4.2)$$

where $\gamma_g(t)$ denotes the g -th output of the TNN; $\omega_{jg}, j = 1, 2, \dots, h_1$; denotes the weight of the link between the j -th hidden node and the g -th output; h_1 is a non-zero positive integer denoting the number of hidden nodes; $\mu_{ij}, i = 1, 2, \dots, n_{in}$ denotes the weight of the link between the i -th input and the j -th hidden node; $x_i(t)$ denotes the i -th input of the TNN; t denotes the current input pattern number; μ_j denotes the weight of the j -th bias term for the hidden layer; ω_g denotes the weight of the g -th bias term for the output layer; n_{in} and c denote the numbers of input and output respectively.

The logarithmic sigmoid function is used as the activation function in the hidden and output nodes. In general, other activation functions can also be applied to the network besides the logarithmic sigmoid function. The outputs of the TNN will be some parameters used by the CNN.

4.3.3.2 Classifier Neural Network

As shown in Fig. 4-6, the classifier neural network (CNN) is the data processor, which is implemented by a 3-layer fully-connected feed-forward neural network. It classifies the input patterns to their corresponding class. The input-output relationship of the CNN is defined as follows.

$$y_k(t) = \text{logsig} \left[\sum_{j=1}^{h_1} \lambda_{jk} \text{tf} \left(\sum_{i=1}^{n_{in}} \mu_{ij} x_i(t) + \mu_j \gamma_c(t); \delta_j, \sigma \right) - \lambda_k \right], \quad k = 1, 2, \dots, n_{out} \quad (4.3)$$

where $y_k(t)$ denotes the k -th output of the CNN; $\lambda_{jk}, j = 1, 2, \dots, h_1; k = 1, 2, \dots, n_{out}$, denotes the weight of the link between the j -th hidden node and the k -th output; $\mu_{ij}, i = 1, 2, \dots, n_{in}$ denotes the weight of the link between the i -th input and the j -th hidden node. It should be noted that the CNN shares the same input-hidden node link weights (μ_{ij}) with the TNN; $\gamma_c(t)$ denotes the last output of the TNN, $c = h_1 + 2$; λ_k denotes the weight

of the k -th bias term for the output layer; $tf(\cdot)$ denotes the hidden-node activation function, which is defined as follows,

$$tf(x_j; \delta_j, \sigma) = \frac{2}{1 + e^{\frac{-(x_j - \delta_j)}{2\sigma^2}}} - 1, j = 1, 2, \dots, h_1 \quad (4.4)$$

where x_j , δ_j and σ denote the input and the two parameters of the activation function respectively. The effects of δ_j and σ to the shape of the activation function are shown in Fig. 4-7. It should be noted that (4.4) is used in order to handle input data which take values between -1 and 1 . δ_j and σ are the outputs of the TNN. Thus, $\delta_j = \gamma_j$ and $\sigma = \gamma_{c-1}$. The parameters provided by the TNN can be regarded as the knowledge instructing the CNN how to handle the input data. The weights in the links of both the TNN and the CNN are trained by the improved GA.

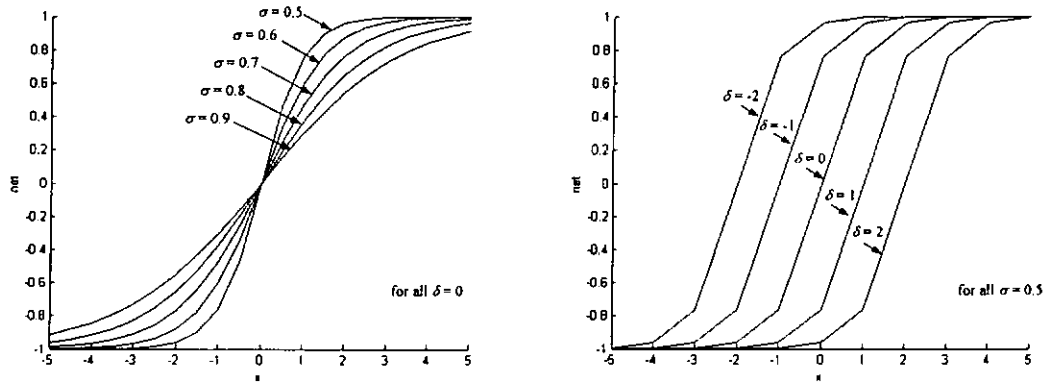


Fig. 4-7. Effects of δ_j and σ to $tf(\cdot)$.

4.3.4 Training and Classification

The training process of the Cantonese-digit/Cantonese-command speech recognition system is to minimize the error between the outputs of the network and the desired values. A fitness function as that given by (3.5) can be used to train the

networks. Based on (3.5), the value of err can be calculated as the mean square error,

$$err = \frac{\sum_{t=1}^{num_pat} \|y_d(t) - y(t)\|^2}{n_{out} \cdot num_pat} \quad (4.5)$$

where $y_d(t)$ denotes the desired output vector; $y(t)$ denotes the network output vector, num_pat denotes the number of training patterns. The desired output vector of the network is defined as follows,

$$y_d = [a_1 \quad a_2 \quad \cdots \quad a_k \quad \cdots \quad a_{n_{out}}] \quad (4.6)$$

where $a_k, k = 1, 2, \dots, n_{out}$, describes the target class of the system. Only the value of a_k for a particular class k is equal to 1, and the rest elements of y_d are all zero. The improved GA trains the proposed networks based on (3.5) and (4.5).

The classification process takes place after the parameters of the network have been trained. A single-network-multi-class scheme is used to realize the classification process, such that the index of the element in $y(t)$ that has the largest value indicates the most likely class of the input pattern.

4.4 Dynamic Pattern Classification for Cantonese Speech Recognition

It has been mentioned in Section 4.3 that the static pattern classification for Cantonese-digit/Cantonese-command speech recognition is suitable for single Cantonese-digit speeches. In order to recognize multi Cantonese-digit speeches, a dynamic pattern classification approach should be used in the classification process. Two example time-domain waveforms of multi-syllable speech are shown in Fig. 4-8.

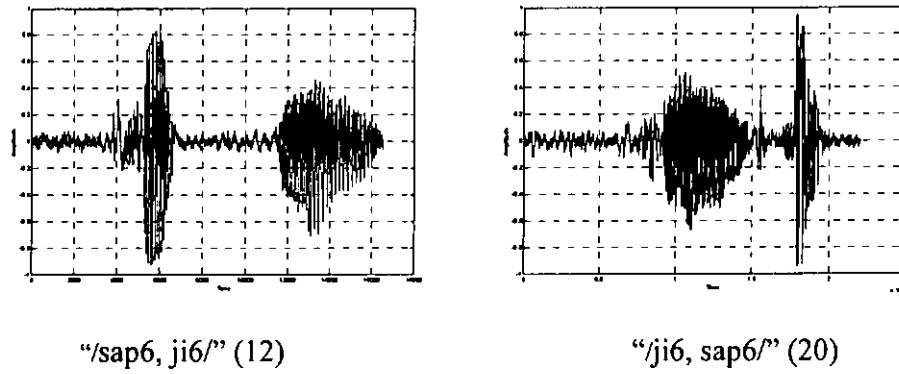


Fig. 4-8. Time-domain waveforms of the Cantonese-number “12” and “20”.

From Fig. 4-8, we see that both the Cantonese-number “12” and “20” contain two syllables, “/sap6/” and “/ji6/”, but with a different sequence. In general, if the static pattern classification approach is used to do the recognition, the first and second syllables are merged together in the recognition process, and the dynamic properties are missed. In order to discriminate the two words, both the temporal characteristics and the dynamic properties of the speech should be modelled. In short, a dynamic pattern classification should be employed to classify multi-syllable speeches.

A recurrent neural network (RNN) is a network that is commonly used to tackle complex dynamic sequential problems. However, training the recurrent network parameters to the desired values is not an easy task. It is because a fixed number of parameters might not be enough to model both the temporal and dynamic features of the input data sets. On the other hand, if this number is too big, the overly complicated network structure might increase the difficulty of the training process. In the following sub-sections, a new architecture of the recurrent neural network is proposed in order to alleviate the problem.

4.4.1 Dynamic Variable-Parameter Neural Network

In order to improve the classifying ability of a traditional RNN, a new structure for the network is proposed, which is shown in Fig. 4-9.

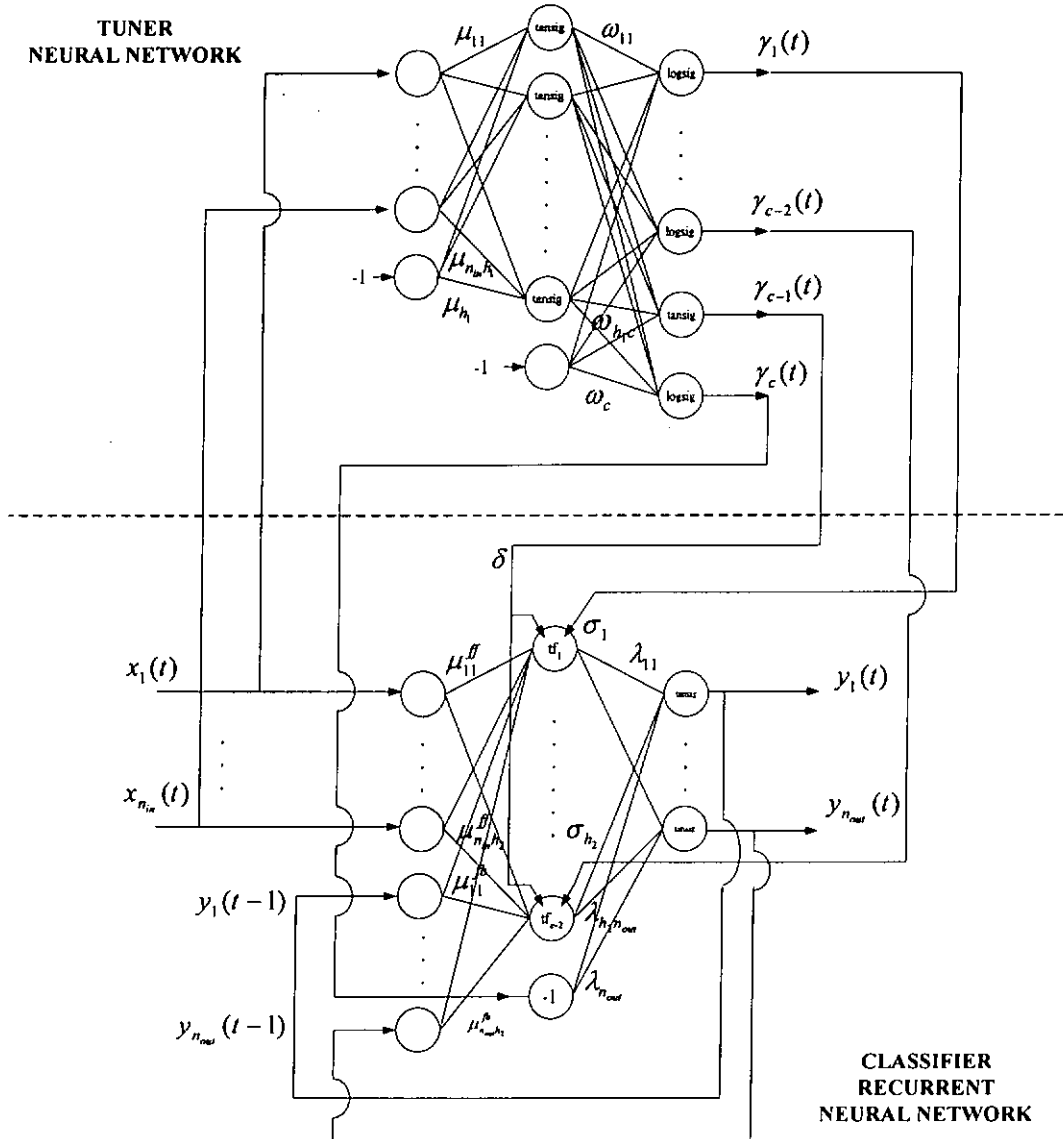


Fig. 4-9. Modified recurrent neural network structure.

As shown in Fig. 4-9, the proposed network consists of a traditional 3-layer feed-forward neural network and a recurrent neural network. The 3-layer

feed-forward neural network is a tuner neural network that serves as the associative memory of the classifier recurrent neural network. The classifier recurrent neural network models the dynamic properties of the input patterns, and processes them based on the parameters provided by the tuner neural network.

4.4.1.1 Tuner neural network

Tuner neural network (TNN) is a traditional 3-layer feed-forward neural network that provides the suitable parameters to the classifier recurrent neural network according to the input patterns. The input-output relationship of the TNN is defined as follows.

$$\gamma_g(t) = \text{logsig} \left[\sum_{j=1}^{h_1} \omega_{jg} \text{tansig} \left(\sum_{i=1}^{n_{in}} \mu_{ij} x_i(t) - \mu_j \right) - \omega_g \right], g = 1, 2, \dots, c-2 \quad (4.7)$$

$$\gamma_{c-1}(t) = \text{tansig} \left[\sum_{j=1}^{h_1} \omega_{jc-1} \text{tansig} \left(\sum_{i=1}^{n_{in}} \mu_{ij} x_i(t) - \mu_j \right) - \omega_{c-1} \right] \quad (4.8)$$

$$\gamma_c(t) = \text{logsig} \left[\sum_{j=1}^{h_1} \omega_{jc} \text{tansig} \left(\sum_{i=1}^{n_{in}} \mu_{ij} x_i(t) - \mu_j \right) - \omega_c \right] \quad (4.9)$$

where $\gamma_g(t)$, $\gamma_{c-1}(t)$, and $\gamma_c(t)$ are the outputs of the TNN for the t -th speech frame; $\text{tansig}(\alpha) = \frac{2}{1 + e^{-2\alpha}} - 1$, $\alpha \in \mathfrak{R}$, is the tangent sigmoid function; $\text{logsig}(\cdot)$ denotes the logarithmic sigmoid function; ω_{jg} , $j = 1, 2, \dots, h_1$, denotes the weight of the link between the j -th hidden node and the g -th output; h_1 is a non-zero positive integer denoting the number of hidden nodes; μ_{ij} , $i = 1, 2, \dots, n_{in}$, denotes the weight of the link between the i -th input and the j -th hidden node; $x_i(t)$ denotes the i -th input of the TNN; μ_j denotes the weight of the j -th bias term for the hidden layer; ω_g , ω_{c-1} , and ω_c are the

link weights of the bias terms for the output layer; n_{in} and c denote the numbers of input and output respectively.

4.4.1.2 Classifier Recurrent Neural Network

As shown in Fig. 4-9, the classifier recurrent neural network (CRNN) is a 3-layer recurrent network. It analyses the input patterns, recurrent information and the information provided by the TNN to produce the network outputs. The input patterns are tokens of short-time speech feature frames. The speech feature frames are fed to the inputs of the network one by one. The outputs of the network are fed back to the network inputs with one sampling-period delay. The input-output relationship of the CRNN is defined as follows.

$$y_k(t) = \text{tansig} \left[\sum_{j=1}^{h_2} \lambda_{jk} \text{tf} \left(\sum_{i=1}^{n_{in}} \mu_{ij}^{ff} x_i(t) + \sum_{q=1}^{n_{out}} \mu_{qj}^{fb} y_q(t-1) \right) - \lambda_k \gamma_c(t) \right],$$

$$k = 1, 2, \dots, n_{out} \quad (4.10)$$

where $y_k(t)$ denotes the k -th output of the CRNN for the t -th speech frame; h_2 is a non-zero positive integer denoting the number of hidden nodes (excluding the bias node); $\lambda_{jk}, j = 1, 2, \dots, h_2, k = 1, 2, \dots, n_{out}$, denotes the weight of the link between the j -th hidden node and the k -th output; $\mu_{ij}^{ff}, i = 1, 2, \dots, n_{in}$ denotes the weight of the link between the i -th input and the j -th hidden node; $\gamma_c(t)$ denotes the last output from the TNN; $\mu_{qj}^{fb}, q = 1, 2, \dots, n_{out}$, denotes the weight of the link between the q -th recurrent input and the j -th hidden node; λ_k denotes the link weight of the k -th bias term of the hidden layer; $\text{tf}(\cdot)$ denotes the hidden node activation function and is defined as follows,

$$tf(\chi_g; \delta, \sigma_g) = \frac{2}{1 + e^{\frac{-(\chi_g - \delta)}{2\sigma_g^2}}} - 1, g = 1, 2, \dots, h_2 \quad (4.11)$$

χ_g , δ and σ_g denote the input and the two parameters of the activation function respectively. It should be noted that δ and σ_g are the outputs of the TNN. Thus, $\sigma_g = \gamma_g$, $g = 1, 2, \dots, h_2$; $h_2 = c - 2$, and $\delta = \gamma_{c-1}$. These parameters are given by the TNN to guide the CRNN how to handle the input data. Owing to the dynamic structure of the proposed NN, both the static and dynamic properties of the speech can be modeled. Different parameter values change the shape of the non-linear activation function (4.11). The CRNN performs dynamic adaptation to the input frame variations through the recurrent links. Each frame pattern will have its own parameter set.

4.4.2 Training and Classification

An individual class-network training process is proposed. Every speech class has its own network of the same structure. The block diagram of the training and classification system is shown in Fig. 4-10.

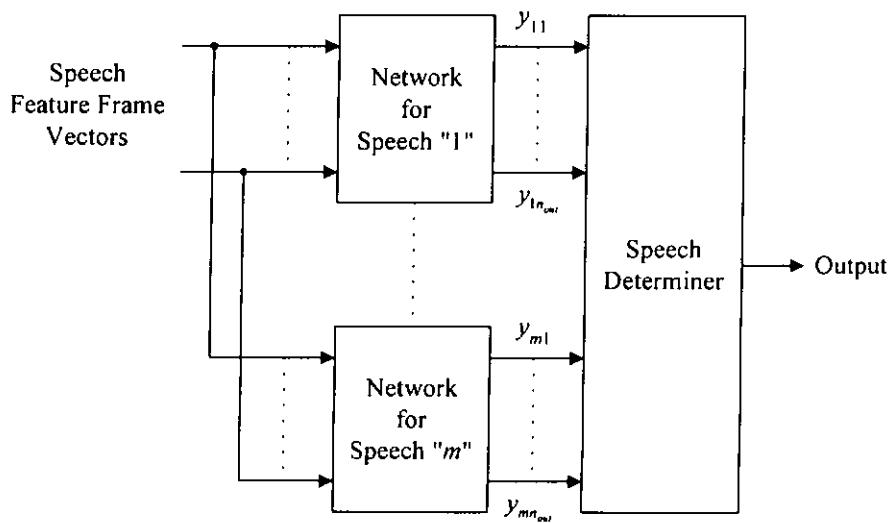


Fig. 4-10. Block diagram of the proposed training and classification process.

The objective of the training process for each network is to adjust the parameters so as to minimize the error between the network outputs and the desired values, where the desired values are the inputs of the network. The performance of the network is governed by the value of *fitness* in (3.5) of Chapter 3, where the error value *err* is given by

$$err = \mathbf{e}_w \text{sort} \left(\left[\sum_{\eta=1}^{ng} |d_{\eta}^1 - y_{\eta}^1| \quad \sum_{\eta=1}^{ng} |d_{\eta}^2 - y_{\eta}^2| \quad \cdots \quad \sum_{\eta=1}^{ng} |d_{\eta}^{n_{out}} - y_{\eta}^{n_{out}}| \right]^T \right) \quad (4.12)$$

$$\mathbf{e}_w = [e_w^1 \quad e_w^2 \quad \cdots \quad e_w^{n_{out}}] = \left[\frac{1}{n_{out}} \quad \frac{2}{n_{out}} \quad \cdots \quad 1 \right] \quad (4.13)$$

where *err* is governed by the sum absolute error between the desired output $\mathbf{d}_{\eta} = [d_{\eta}^1 \quad d_{\eta}^2 \quad \cdots \quad d_{\eta}^{n_{out}}]$ and the network output $\mathbf{y}_{\eta} = [y_{\eta}^1 \quad y_{\eta}^2 \quad \cdots \quad y_{\eta}^{n_{out}}]$; \mathbf{e}_w denotes the error-weight vector, *sort*(·) returns a vector that has the argument vector's elements sorted in descending order; *ng* denotes the number of groups that can be segmented from the speech (the method to obtain the segmentation group for a speech will be described in Chapter 5). It should be noted that the desired value is equal to the input in the training process. The fitness value will be optimized by the improved GA.

After the training process of each network has been done, *m* sets of parameters are obtained. In order to classify the target class to which an input pattern belongs, the input pattern will be tested simultaneously by the networks. Therefore, *m* fitness values will be produced by the networks according to (4.12). The class of the network with the highest fitness value is the most likely class to which the input pattern belongs.

4.5 Comparison

Comparison between the four proposed Cantonese-digit/Cantonese-command speech recognition approaches will be given in this section. First, the static pattern classification for Cantonese-digit/Cantonese-command speech recognition comparisons will be made between the self-structured NFN, variable-parameter NFN and variable-parameter NN. Then, the best performed static pattern classification approach will be compared with the dynamic pattern classification approach.

Out of the three static pattern classification approaches, two types of networks are used: neural-fuzzy network and neural network. The differences between an NFN and an NN have been discussed in Section 3.4, Chapter 3. Considering only the NFNs, the number of rules of the self-structured NFN after training will be smaller than the variable-parameter NFN. However, thanks to the specific structure of the variable-parameter NFN, the variations of the input patterns can be adaptively tackled. Thus, the variable-parameter NFN can be used to classify a large number of classes of which each class can have input patterns separated far apart from each other. To illustrate the merits of the variable-parameter NFN more clearly, consider two recognition classes. The data sets S1 & S2 belong to class 1 but separated far apart, and the data set S3 belongs to class 2 in the (x_1 , x_2) spatial domain as shown in Fig. 4-11(a). On using a single traditional NFN, as its weights are trained to minimize the error between the network output and the desired value for the two data sets separated far apart; if the number of network parameters is not enough, the network will only be trained to recognize a data set R between S1 and S2. Therefore, S3 could be misclassified as class 1 as shown in Fig. 4-11(b). The recognition accuracy will then be lowered. However, if the number of parameters is large, the number of iteration required in the training process will be increased. In order to reduce the number of parameters of the network, the variable-parameter NFN can be considered. On using

this network, when the input data belongs to S1, the tuner network will provide parameter set 1 for the classifier network to handle the data set S1. When the input data S2 is given, the parameter set corresponding to S2 will be used. Hence, the variable-parameter NFN operates like a multi-network handling multiple classes. This property is also possessed by the variable-parameter NN. Comparisons of self-structured NFNs, variable-parameter NFNs and variable-parameter NNs in terms of number of parameters and factor of consideration on testing 5 Cantonese-command speeches and 13 Cantonese-number speeches are summarized in Table 4-1.

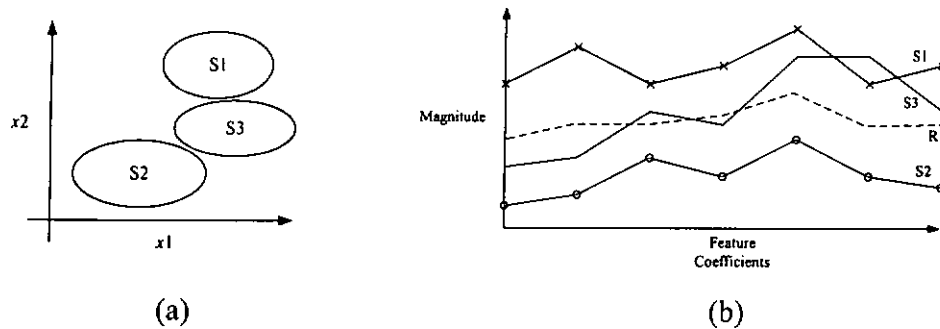


Fig. 4-11. (a) Two data sets in spatial domain. (b) Feature curves of the two sets.

	Self-structured NFN	Variable-parameter NFN	Variable-parameter NN
No. of parameters	Small	Large	Medium
Factor of Consideration	Suitable for data in a small domain	Suitable for data in a large domain	

Table 4-1. Comparisons between the self-structured NFN, the variable-parameter NFN and the variable-parameter NN.

To evaluate the performance of the static and dynamic pattern classification for Cantonese-speech recognition, the network which provide the best performance from the three static pattern classification approaches are compared with the dynamic

variable-parameter NN on testing 13 Cantonese-number speeches in terms of number of network parameters and factor of consideration. Owing to the lack of recurrent links in the static pattern classification approaches, all the speech frames have to be cascaded together so as to incorporate the dynamic feature into the speech vector. Therefore, the number of network parameter of the static pattern classification approaches is more than the dynamic pattern classification approach. Table 4-3 summarized the results of the comparison.

	Self-structured NFN/Variable-parameter NFN/NN	Dynamic variable-parameter NN
No. of parameters	Small for single-syllable speech Large for double-syllable speech	Large
Factor of Consideration	A decision stage is required to determine if it is a single- or double-syllable speech	Only single network is needed

Table 4-2. Comparisons between the static and dynamic pattern classification approaches.

In short, different approaches are suitable for different problems. Table 4-3 summarizes the uses of the proposed approaches.

	Appropriate approach(es)
Static pattern recognition	1, 2 and 3
Dynamic pattern recognition	4
Widely separated data in each target class	2, 3 and 4
Data in a small domain	1
Fast training process required	1 and 3
Long training time acceptable	2 and 4

Approach 1: Self-structured NFN

Approach 3: Variable-parameter NN

Approach 2: Variable-parameter NFN

Approach 4: Dynamic variable-parameter NN

Table 4-3. Summary on the uses of the proposed approaches.

For the static pattern recognition application, approaches 1, 2 and 3 are appropriate. This is because no recurrent path is required for this kind of application. For the dynamic pattern recognition application, approach 4 is more appropriate thanks to the presence of recurrent links. For applications that need to classify widely separated data groups, approach 2, 3 and 4 are more appropriate. It is because the associative memory of the network can supply adaptive information with respect to the input patterns for the classifier. However if an application has data located in a small domain, approach 1 is suitable since the network size can be small and only a single network can provide a good performance. Concerning the time of the training process, approaches 1 and 3 are suitable to the applications that require a fast training process, while approaches 2 and 4 are suitable if a longer training process can be accepted. It is because the networks of approaches 2 and 4 are large in size, and more network parameters (fuzzy rules parameters and recurrent path parameters, etc.) are needed.

4.6 Summary

Three approaches for the static pattern classification for Cantonese-digit/Cantonese-command speech recognition and one approach for the dynamic pattern classification for Cantonese-number speech recognition have been proposed in this chapter. A self-structured neural-fuzzy network, a variable-parameter neural-fuzzy network, and a variable-parameter neural network are developed for the static Cantonese-digit speech recognition. By employing the self-structured neural-fuzzy network, both the number of rules and the membership functions of the network can be tuned. Improvements have been made to the traditional networks for static speech recognition by introducing associative memory. By interconnecting two networks of the same structure, a better performance than that of the conventional 3-layer fully connected NN can be obtained based on the application examples in this

thesis. The two networks are named the tuner network (associative memory) and the classifier network (data processor). The parameters of the tuner network are fixed after the training process. This process emulates the memorizing process of the human brain. The classifier network associates the information supplied by the memory with the information of the input pattern, and determines the most suitable class of that input pattern. Because the associative memory depends also on the input pattern, the variable-parameter network structure can cope with a large number of different input patterns of the same class. In addition, fewer network parameters are needed to model a large number of input patterns that are separated far apart.

By applying the idea of associative memory to a recurrent neural network, the dynamic variable-parameter neural network is developed. Its tuner network is realized by a 3-layer feed-forward neural network, and its classifier network is realized by a 3-layer recurrent neural network. By using the recurrent neural network as the classifier, the dynamic properties of the input patterns can be modelled. In practice, the sequence of the syllables in a speech recognition system can be tackled by the recurrent neural network, so that multi-syllable Cantonese-digit speeches can also be recognized.

Comparisons among the traditional approach and the proposed approaches in terms of the number of parameters and factor of consideration have been made. The uses of the proposed approaches have been discussed. Experimental results of applying the four proposed networks for Cantonese speech recognition will be provided in Chapter 5.

CHAPTER 5

APPLICATIONS TO ELECTRONIC BOOK AND RESULTS

5.1 Introduction

This chapter reports the application of the proposed approaches mentioned in Chapter 3 and Chapter 4 to the electronic book (eBook). Results will be provided to illustrate the merits of the proposed approaches. Besides, the proposed networks will be compared with the traditional networks and among themselves in order to evaluate the performance of the proposed networks.

This chapter is organized as follows. The development of the electronic book reader will be reported in section 5.2. The results for the graffiti interpretation applications (Graffiti-to-command and Graffiti-to-character interpretation), and the performance comparison between the proposed approaches and the traditional approaches will be given in section 5.3. The results for the speech recognition applications (speech-to-command and speech-to-text recognition), and the performance comparison between the proposed approaches and the traditional approaches will be given in section 5.4.

5.2 Electronic Book Reader

Electronic book (eBook) reader refers to the software developed in an eBook reading device that is used to process eBook content of a given standard format

(Fig.5-1). EBooks are winning their popularity as a kind of media that can offer rich contents and features such as multimedia presentations, instant dictionaries and bookmark functions etc. within a small handheld device. The eBook reading device should have no keyboard or mouse. Input to the device can be made through a stylus on a touch-screen, or a microphone. They are two natural input methods that require neural network techniques to help their implementation. In this thesis, neural network techniques are proposed to realize four types of applications: graffiti-to-command interpretation, graffiti-to-character interpretation, Cantonese speech-to-command recognition, and Cantonese speech-to-text recognition.

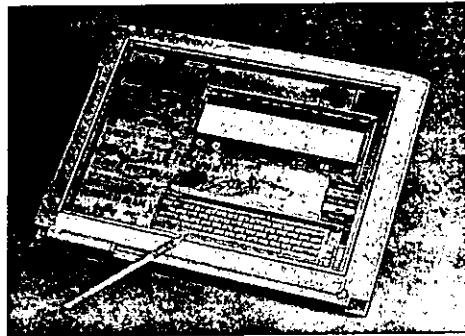


Fig. 5-1. Electronic book reading device.

5.3 Graffiti Interpretation Applications

The interpretation of graffiti commands and characters for eBooks will be presented in this section. The block diagram of the interpretation system is shown in Fig. 5-2. Thanks to the touch-screen, trajectories made by the stylus can readily be recorded as coordinate data. As the sampling rate of the eBook touch-screen is fixed, the number of coordinates data collected will vary with the speed of the sketching. The collected data could sometimes be more than enough or less than required. As a result, a preprocessing stage for the feature-point extraction is introduced.

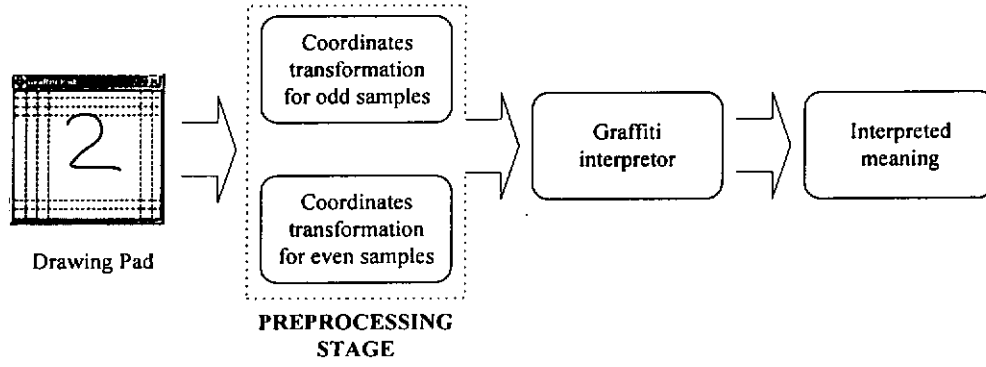


Fig. 5-2. Block diagram of the graffiti interpretation system.

5.3.1 Feature Point Extraction Methodology

A point on the eBook screen is characterized by a number based on the x - y coordinates on a writing area, which is shown in Fig. 5-3. The size of the writing area is $x_{\max} \times y_{\max}$. The upper left corner is set as $(0, 0)$. The points of the graffiti will be sampled uniformly in the following way. First, the input graffiti is divided into $n_{in}-1$ uniformly distanced segments characterized by n_{in} points, including the start point and the end point. Each point is labeled as (x_i, y_i) , $i = 1, 2, \dots, n_{in}$. The odd points (with i being an odd integer between 1 and n_{in}) taken alternatively are converted to the values ρ_{odd_i} using the following equation:

$$\rho_{odd_i} = x_i x_{\max} + y_i \quad (5.1)$$

Similarly, the even points (with i being an even integer between 1 and n_{in}) are converted to the value ρ_{even_i} using the following equation:

$$\rho_{even_i} = y_i y_{\max} + x_i \quad (5.2)$$

Then the network input

$$\mathbf{z} = [z_1 \quad z_2 \quad \cdots \quad z_{n_m}] = \frac{\begin{bmatrix} \rho_{odd_1} & \rho_{even_1} & \cdots & \rho_{even_{\frac{n_m}{2}}} \end{bmatrix}}{\begin{bmatrix} \rho_{odd_1} & \rho_{even_1} & \cdots & \rho_{even_{\frac{n_m}{2}}} \end{bmatrix}} \quad (5.3)$$

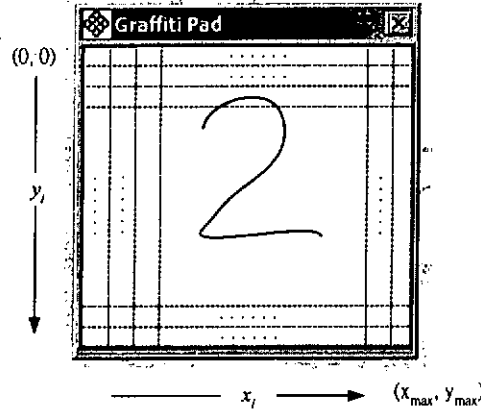


Fig. 5-3. Feature points capturing method.

These numbers, z_i , $i = 1, 2, \dots, n_m$, will be used as the inputs of the proposed networks. The advantage of using the above feature point extraction methodology is that it can better distinguish the differences between classes. The proposed networks are used as the classifier that performs the template matching operation in the interpretation process. The self-structured NN and the self-structured NFN are applied to realize the graffiti commands interpretation and the graffiti characters interpretation respectively. The results for each application are given as follows.

5.3.2 Command Interpretation by the Self-structured NN

The self-structured NN mentioned in chapter 3, section 3.2 is applied to recognize 3 graffiti commands in the eBook environment. The three graffiti patterns are shown in Fig. 5-4.

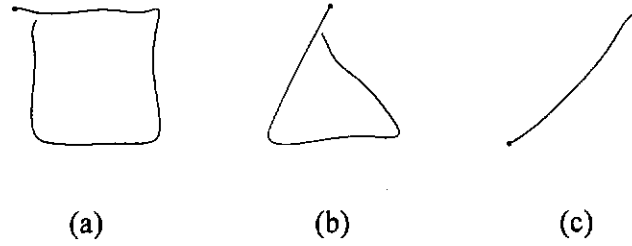


Fig. 5-4. Graffiti used in the eBook (the dot indicates the starting point): (a) square (b) triangle (c) straight line.

The interpretation process is achieved by a three-layer neural network (eight-input-single-output) with link switches. The eight inputs nodes, z_i , $i = 1, 2, \dots, 8$, represent the eight sampled points of the graffiti based on the feature point extraction methodology mentioned in the previous sub-section. The output node $y(t)$ represents the graffiti value. Its value lies between 0.1 and 0.7; $y(t) \in [0.1 \ 0.3)$ is defined for a straight line, $y(t) \in [0.3 \ 0.5)$ is defined for a triangle, $y(t) \in [0.5 \ 0.7)$ is defined for a rectangle. The neural network is trained with some known input data ($\mathbf{z}^d(t)$) and output data ($y^d(t)$). The graffiti patterns are obtained from free sketches by the same person following the patterns shown in Fig. 5-4 on the touch-screen. For each graffiti pattern, eight sets of eight sampled points are used. Hence, we have 24 sets of input-output data to train the proposed network. The training output data, $y^d(t)$, takes the middle value of each range. Thus, $y^d(t)$ is 0.2 for a straight line, 0.4 for a triangle, and 0.6 for a rectangle.

In this self-structured neural network, the number of hidden nodes is 11, which was chosen arbitrarily. The improved GA (which is detailed in Appendix) is employed to tune the parameters and structure of the neural network. The objective is to maximize the fitness function defined in (3.5) and (3.6) of Chapter 3. The best fitness value is 1 and the worst one is 0. The population size used for the improved GA is 10. The lower and the upper bounds of the link weights are defined as

$$\frac{-3}{\sqrt{n_h}} \geq v_{ij}, w_{j1}, b_j^1, b_1^2 \geq \frac{3}{\sqrt{n_h}} \text{ and, } -1 \geq s_{j1}^2, s_{ij}^1, s_j^1, s_1^2 \geq 1, i = 1, 2, \dots, 8; j = 1, 2, \dots, 11.$$

The chromosome is $[s_{j1}^2 \ w_{j1} \ s_{ij}^1 \ v_{ij} \ s_j^1 \ b_j^1 \ s_1^2 \ b_1^2]$ for all i and j . The initial values of the parameters inside the chromosome (genes) are randomly generated. For comparison, a fully connected three-layer feed-forward neural network (eight-input-one-output) trained by the improved GA is also used to interpret the graffiti commands. The working conditions are the same as those mentioned above. The number of iteration to train the two neural networks is 1000. After training, 4 samples of each kind of graffiti ($4 \times 3 = 12$ testing graffiti commands) are used to test the performance of the trained neural networks. The results are obtained from 5 runs and the best performance is tabulated in Table 5-1. The output values of the two NNs are plotted in Fig. 5-5 (training) and Fig. 5-6 (testing) respectively.

NN	Training fitness value	No. of links	Training error	Testing error	Searching Time (s)
Self-structured	0.977343	60	0.0232	0.0306	494.44
Traditional	0.971251	111	0.0296	0.0408	1857.5

Table 5-1. Results of the self-structured NN and the traditional NN for interpreting graffiti commands.

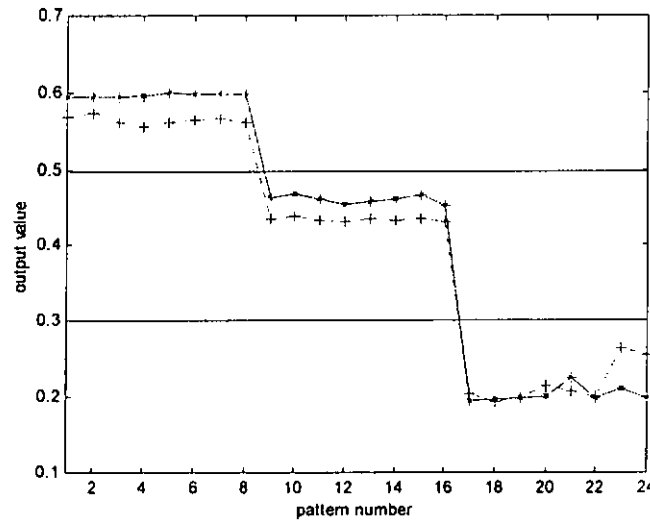


Fig. 5-5. Output values of the two neural networks for the 24 training graffiti: Solid line with (*): self-structured NN; Dash line with (+): traditional NN.

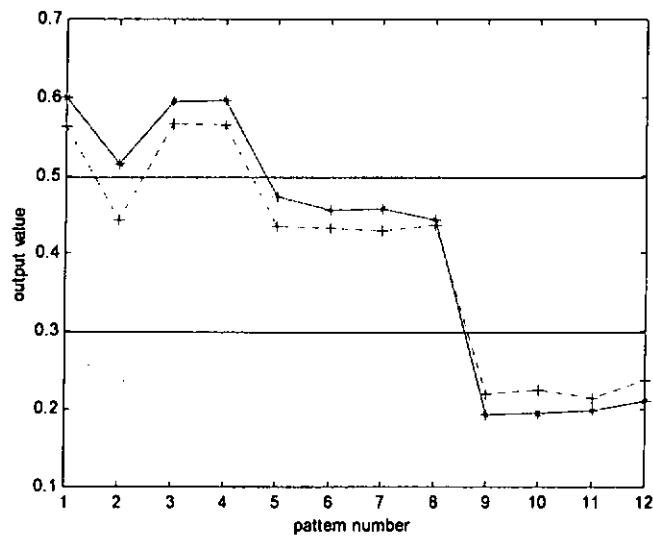


Fig. 5-6. Output values of the two neural networks for the 12 testing graffiti: Solid line with (*): self-structured NN; Dash line with (+): traditional NN.

From Table 5-1, it can be observed that the proposed self-structured NN trained with the improved GA provides similar results in terms of accuracy (fitness values) and mean absolute errors (MAE) for training and testing with the data sets provided above. However, on comparing the number of connected links, the proposed NN is 60 after learning as compared with the number of fully connected links of 111, which includes the bias links. In Fig. 5-5, the training patterns 1-8 are graffiti in rectangle, patterns 9-16 are graffiti in triangle and patterns 17-24 are graffiti in straight line. From this figure, it can be seen that all output values are within the defined region and the interpretation is successful. In Fig. 5-6, the testing patterns 1-4 are graffiti in rectangle, patterns 5-8 are graffiti in triangle and patterns 9-12 are graffiti in straight line. Again, all output values are within the defined regions when the proposed network is used. However, if the traditional neural network is used, pattern 2 is outside the correct region. It can be concluded that the proposed network performs better than the traditional network with the data sets reported in this section if the same number of hidden nodes is used.

5.3.3 Graffiti Interpretation by the Self-structured NFN

Inside the eBook reader, the graffiti of the numeric characters “0” to “9” and three control characters (*backspace*, *carriage return* and *space*) are interpreted by the self-structured NFN mentioned in Section 3.4, Chapter 3. The shapes of the graffiti patterns are shown in Fig. 5-7. Ten uniformly sampled points of the graffiti will be taken as the inputs of the interpreter. The points are obtained using the feature point extraction methodology mentioned early. These ten numbers, z_i , $i = 1, 2, \dots, 10$, are used as the inputs of the proposed NFNs as shown in Fig. 3-3 of Chapter 3. One NFN is used for one class of pattern. Each NFN has 10 inputs, 10 outputs, 15 membership functions (i.e. $m_i = 15$ for all i), and rule switches. One hundred sets of sampled points for each graffiti pattern are sketched on the touch-screen by the same person following the patterns shown in Fig. 5-7 for the training process.

















Characters	Strokes	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	

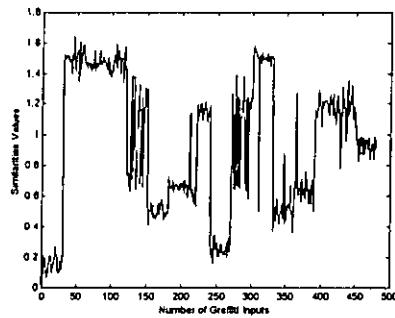
Fig. 5-7. Graffiti of the numeric and control characters (the dot indicates the starting point of the graffiti).

The improved GA is used to tune the membership functions and the number of rules of the NFNs. The objective is to maximize the fitness function given by (3.5) and (3.13) of Chapter 3. The best fitness value is 1 and the worst one is 0. The population size is 10. The lower and upper bounds of the link weights are defined as $0 \leq \bar{x}_{ig_i}, \sigma_{ig_i}, \varsigma_{gj}, w_{gj} \leq 1, i = 1, 2, \dots, 10; j = 1, 2, \dots, 10; g = 1, 2, \dots, 15; g_i = 1, 2, \dots, 15$.

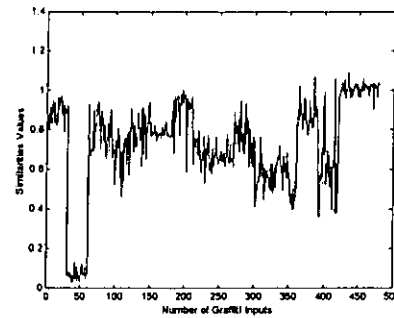
15. The chromosomes of the GA process used are $[\bar{x}_{ig_i} \quad \sigma_{ig_i} \quad \varsigma_{gj} \quad w_{gj}]$ for all i, j, g and g_i . The initial values of the parameters inside the chromosome are randomly generated. The number of iteration to train the neural networks is 2000. After training, 30 samples of each graffiti pattern, which are obtained from the same person, are used to test the performance of the trained NFNs. The testing process was done by providing the 30 test patterns of a certain graffiti pattern to all the graffiti pattern recognition networks as mentioned in Section 3.3.3. The recognition network parameters which provide the highest fitness value after 5 runs of 2000-iteration training will be chosen to perform the classification testing. The results are tabulated in Table 5-2. From this Table, it can be observed that the numbers of connected links (rules) of the NFNs are reduced after training (each NFN has 150 rules initially). Fig. 5-8 shows the similarity values (which are defined by (3.14) in Chapter 3) given by each trained NFN when 480 (30 for each type of graffiti pattern) testing graffiti patterns are input to the NFNs. It can be seen that the NFN trained by a particular class of graffiti patterns will provide a smaller similarity values when other graffiti patterns of the same class are input to that NFN. For example, in Fig. 5-8(a), the similarity values of the first 30 testing graffiti patterns (numeric character '0') are smaller, as that NFN is trained by 100 graffiti patterns of '0'.

Neural-fuzzy network	Training fitness value	No. of rule of the network	Testing fitness value	Recognition error rate (%)
0(a)	0.9986	80	0.9972	6.6667%
0(b)	0.9997	113	0.9995	0%
1	0.9978	89	0.9978	0%
2	0.9994	113	0.9990	0%
3	0.9995	109	0.9969	3.3333%
4	0.9988	92	0.9956	3.3333%
5(a)	0.9992	100	0.9974	3.3333%
5(b)	0.9985	103	0.9966	0%
6	0.9992	93	0.9948	6.6667%
7	0.9991	113	0.9989	3.3333%
8(a)	0.9995	108	0.9962	0%
8(b)	0.9997	98	0.9834	6.6667%
9	0.9990	107	0.9987	0%
Back Space	0.9996	97	0.9976	0%
Return	0.9996	113	0.9988	0%
Space	0.9995	91	0.9992	0%
Overall recognition accuracy				98%

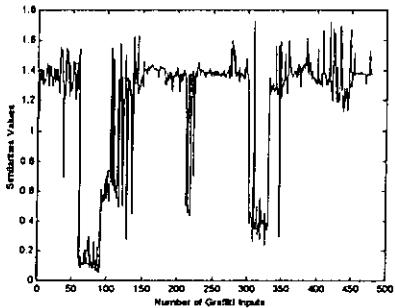
Table 5-2. Results of the proposed self-structured NFNs for interpreting graffiti.



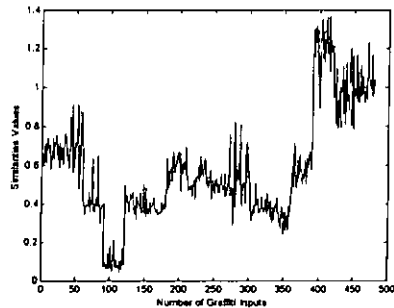
(a). '0 (a).



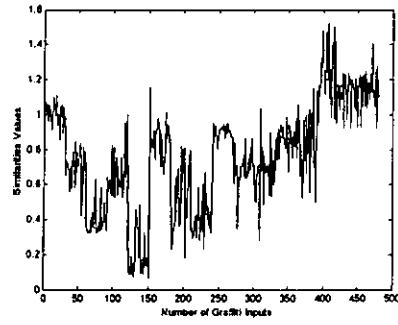
(b). '0 (b).



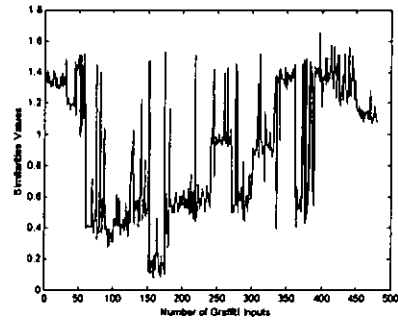
(c). '1'.



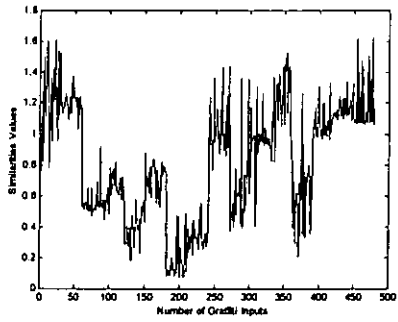
(d). '2'.



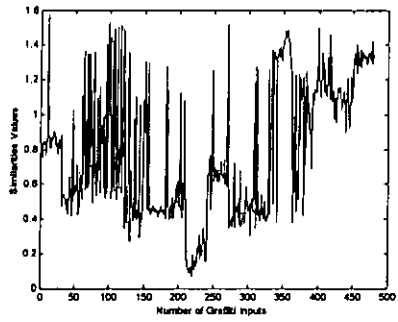
(e). '3 .



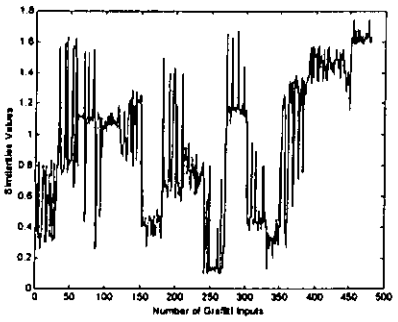
(f). '4 .



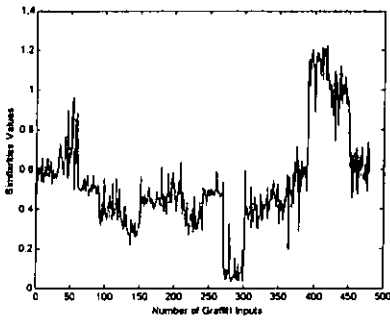
(g). '5 (a).



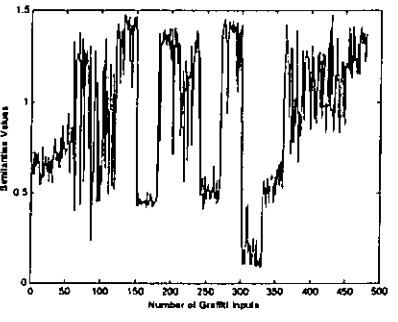
(h). '5 (b).



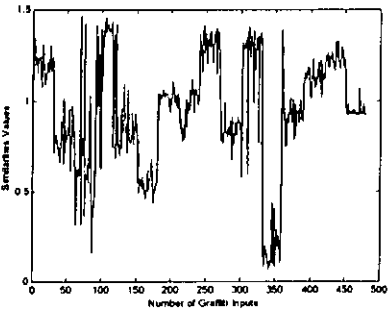
(i). '6 .



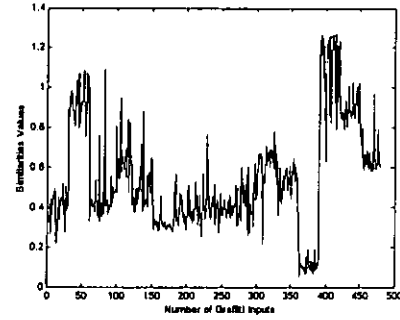
(j). '7 .



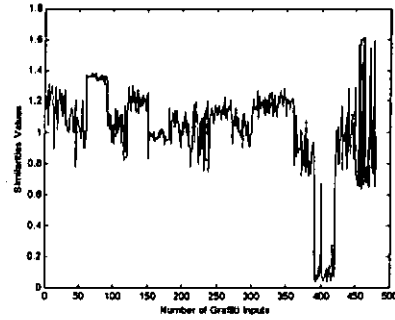
(k). '8 (a).



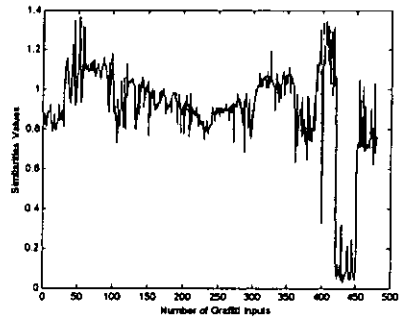
(l). '8 (b).



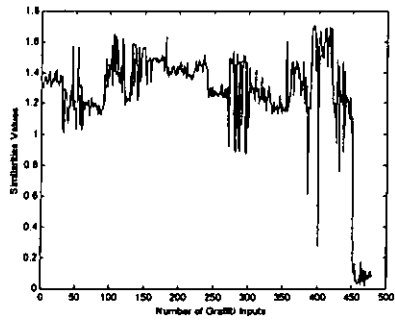
(m). '9'.



(n). 'backspace'.



(o). 'carriage return'.



(p). 'space'.

Fig. 5-8. Similarity values given by the 16 self-structured NFNs for the 480 testing graffiti patterns (30 for each type).

For comparison, a traditional NFN is also used as the graffiti interpreter. The number of membership functions used by the traditional NFN is 15, which was chosen arbitrarily. The improved GA with the same control parameters for the proposed approach is employed to train the network. Five runs were done for each training process again. The best training results and the corresponding testing results are tabulated in Table 5-3.

Neural-fuzzy network	Training fitness value	No. of rule of the network	Testing fitness value	Recognition error rate (%)
0(a)	0.9995	150	0.9974	10%
0(b)	0.9990	150	0.9976	0%
1	0.9994	150	0.9990	0%
2	0.9994	150	0.9987	0%
3	0.9995	150	0.9897	26.6667%
4	0.9997	150	0.9977	3.3333%
5(a)	0.9986	150	0.9962	6.6667
5(b)	0.9981	150	0.9740	33.3333
6	0.9993	150	0.9957	10%
7	0.9989	150	0.9890	6.6667%
8(a)	0.9996	150	0.9968	3.3333%
8(b)	0.9995	150	0.9928	13.3333%
9	0.9991	150	0.9987	6.6667%
Back Space	0.9996	150	0.9945	3.3333%
Return	0.9989	150	0.9945	3.3333%
Space	0.9996	150	0.9958	3.3333%
Overall recognition accuracy				92%

Table 5-3. Results of the traditional neural-fuzzy networks for interpreting graffiti.

It can be seen from Table 5-2 and Table 5-3 that the proposed approach provides 98% recognition accuracy while 92% recognition accuracy is obtained from the traditional approach (using a traditional NFN with 15 membership functions). In terms of the number of rules, the proposed approach uses not more than 113 while the traditional approach uses 150 for recognizing 16 graffiti patterns.

5.3.4 Comparison between the Self-structured NN and the Self-structured NFN

The previous results illustrate that both the self-structured NN and the self-structured NFN requires smaller numbers of links/rules than their corresponding traditional networks. In order to compare the self-structured NN to the self-structured NFN, the 16 graffiti patterns recognized by the self-structured NFNs

are also interpreted by the self-structured NNs. 50 training and 30 testing patterns for each class of graffiti are used for the examination. Both approaches use a 16×10 -input-10-output network structure as shown in Fig. 3-3 of Chapter 3. The number of hidden nodes of the self-structured NN is 11 and the number of membership function used by self-structured NFN is 15, which are chosen arbitrarily. The results in Table 5-4 and Table 5-5 were obtained from the best network-parameter-set after training for five times, each with 2000 times of iteration.

Neural network	Training fitness value	No. of parameter for the network	Testing fitness value	Recognition error rate (%)
0(a)	0.9976	218	0.9972	0%
0(b)	0.9976	205	0.9975	0%
1	0.9957	212	0.9959	0%
2	0.9963	216	0.9955	0%
3	0.9970	217	0.9951	3.3333%
4	0.9988	207	0.9967	3.3333%
5(a)	0.9985	204	0.9972	0%
5(b)	0.9980	147	0.9962	0%
6	0.9988	207	0.9958	10%
7	0.9963	177	0.9967	0%
8(a)	0.9985	212	0.9961	3.3333%
8(b)	0.9986	220	0.9942	33.3333%
9	0.9932	212	0.9919	6.6667%
Back Space	0.9942	212	0.9882	6.6667%
Return	0.9948	214	0.9945	0%
Space	0.9972	204	0.9969	0%
Overall recognition accuracy				96%

Table 5-4. Results given by the self-structured NN.

Neural-fuzzy network	Training fitness value	No. of parameter for the network	Testing fitness value	Recognition error rate (%)
0(a)	0.9986	240	0.9972	6.6667%
0(b)	0.9997	339	0.9995	0%
1	0.9978	267	0.9978	0%
2	0.9994	339	0.9990	0%
3	0.9995	327	0.9969	3.3333%
4	0.9988	276	0.9956	3.3333%
5(a)	0.9992	300	0.9974	3.3333%
5(b)	0.9985	309	0.9966	0%
6	0.9992	279	0.9948	6.6667%
7	0.9991	339	0.9989	3.3333%
8(a)	0.9995	324	0.9962	0%
8(b)	0.9997	294	0.9834	6.6667%
9	0.9990	321	0.9987	0%
Back Space	0.9996	291	0.9976	0%
Return	0.9996	339	0.9988	0%
Space	0.9995	273	0.9992	0%
Overall recognition accuracy				98%

Table 5-5. Results given by the self-structured NFN.

From Table 5-4 and Table 5-5, we can see that the maximum number of parameters used by the self-structured NNs is 218, while up to 339 parameters are needed by one of the self-structured NFNs. The number of FLOPs for the self-structured NN and the self-structured NFN are (187 ~ 258) and (880 ~ 923) respectively. This indicates that the network of the self-structured NFN is more complicated than the self-structure NN for recognizing the data patterns. However, it also shows that using more parameters for generating good expert knowledge (suitable choices of membership functions) might provide a better performance for this application example. From these tables, the recognition accuracy of the self-structured NN (96%) and the self-structured NFN (98%) are similar in values. Thus, both networks are good for the graffiti interpretation application in the eBook

environment. The above observation concurs with the qualitative comparisons discussed in Chapter 3.

5.3.5 Comparison between the two proposed approaches and the commercial software package

In order to evaluate the performance of the handwriting recognition developed for the eBook reader, the two proposed approaches are also compared with a commercial software tool used in PDAs. The patterns “0” to “9” are used for the testing. During the tests, each handwritten pattern is sketched for 30 times by the same person. The results obtained from the two proposed approaches and the PDA are listed in Table 5-6.

Handwritten Patterns	eBook		PDA
	Self-structured NN	Self-structured NFN	
0	100%	93%	94%
1	100%	100%	100%
2	100%	100%	93%
3	97%	97%	83%
4	97%	97%	90%
5	97%	97%	97%
6	93%	93%	97%
7	100%	93%	97%
8	67%	97%	93%
9	87%	90%	97%
Overall accuracy	94%	96%	94%

Table 5-6. Comparison of the performance of the proposed approaches and the PDA software tool.

It can be seen that the overall recognition rate of the two proposed handwriting recognition approaches are similar to that of the PDA software tool. All approaches can offer a success rate of higher than 90%.

5.4 Cantonese Speech Recognition

5.4.1 Static Pattern Classification for Cantonese Speech Recognition System

The three proposed static pattern classification approaches in Chapter 4 are applied to realize Cantonese-digit/Cantonese-command speech recognition systems. The self-structured NFN is used to recognize 3 Cantonese-command speeches, the variable-parameter NFN is applied to recognize 10 Cantonese-speech of decimal digits, and the variable-parameter NN is used to recognize 5 Cantonese-command speeches. The merits of each proposed approach will be demonstrated by comparing it with the traditional approach. The procedure taken by the speech recognition system is shown in Fig. 5-9.

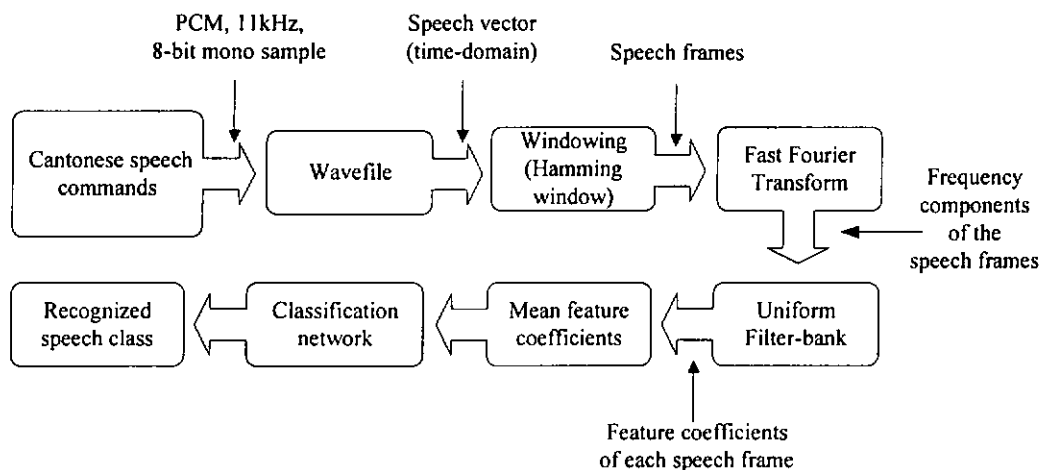


Fig. 5-9. Block diagram of the static pattern classification for Cantonese speech recognition.

5.4.1.1 Feature Extraction Methodology

Speech signals are firstly recorded by the eBook reader through the microphone. The record duration of each speech is manually controlled by the speaker. Thus, the start-record button was pressed by the speaker when he starts to pronounce the

Cantonese character and the stop-record button was pressed when the pronunciation of the character finishes. The speech signals are recorded in mono, 8-bit PCM format sampled at 11 kHz. Then the speech signal in time-domain $s(\tau)$ is windowed by 128-sample Hamming windows $wh(\cdot)$ with 50% overlap to form speech frames; where

$$wh(\tau) = \begin{cases} 0.54 - 0.46 \cos\left(\frac{2\pi\tau}{127}\right), & 0 \leq \tau \leq 127 \\ 0 & \text{otherwise} \end{cases} \quad (5.4)$$

The windowed speech signal will be transformed into the frequency domain using Fast Fourier Transform (FFT). The real part of the frequency components at the n -th speech frame is defined below.

$$\mathbf{Sf}_n = [Sf_n(0) \quad Sf_n(1) \quad \dots \quad Sf_n(127)] = \text{Re}\{\text{FFT}([wh(\tau)s_n(\tau)]\}, \quad 0 \leq \tau \leq 127 \quad (5.5)$$

where $\text{Re}\{\cdot\}$ denotes the real part of the argument vector, and $s_n(\tau)$ is the τ -th element of the n -th speech frame.

A uniform filter-bank is then applied to process the speech frames in order to retrieve the feature coefficients (one filter gives one coefficient.) This process done by the uniform filter-bank can be described by the following equations:

$$c_n^\beta = \frac{20 \log_{10} \sum_{l=1}^{\alpha} Sf_n(\rho_\beta + l - 1)}{\alpha}, \quad \beta = 1, 2, \dots, \text{no_filter} \quad (5.6)$$

$$\alpha = \text{floor}\left(\frac{128}{no_filter}\right) \quad (5.7)$$

$$\rho_\beta = \alpha(\beta - 1), \quad \beta = 1, 2, \dots, no_filter \quad (5.8)$$

$$\mathbf{D}_n = [c_n^2 - c_n^1 \quad c_n^3 - c_n^2 \quad \dots \quad c_n^{no_filter} - c_n^{no_filter-1}] \quad (5.9)$$

where c_n^β denotes the mean power of a speech frame generated by the β -th band-pass filter for the n -th frame, no_filter denotes the number of band-pass filter, α denotes the number of the frequency components entering the band-pass filter, $\text{floor}(\cdot)$ denotes the floor function which is used to round-up a floating point number; \mathbf{D}_n is a vector formed by the magnitude differences between two consecutive band-pass filter outputs of the n -th speech frame. The mean feature coefficients of all the speech frames are calculated and normalized. They are the inputs to the neural network that performs the template matching classification process.

5.4.1.2 Self-structured NFN

Three Cantonese characters, “/bat1/” (筆), “/daai6/” (大) and “/zyu3/” (註) (3 classes), are used to represent the eBook commands of “Pen”, “Zoom-in” and “Annotation”. They are to be recognized by the self-structure NFN as discussed in Chapter 3. 100 training patterns and 20 testing patterns for each command word are recorded from a male speaker in a silent office environment. Using the method mentioned in the previous sub-section, 20 feature coefficients are obtained from each command word. A 20-input-3-output self-structure NFN is then used to classify the three command words. The improved GA is employed to train the network using the training patterns. The fitness function of (3.5) in Chapter 3 is used, where.

$$err = \frac{\sum_{t=1}^{300} \frac{\|y_d(t) - y(t)\|^2}{3}}{300} \quad (5.9)$$

The first 100 training patterns correspond to the Cantonese word “筆”, and the desired output y_d is [1 0 0]. The next 100 training patterns correspond to Cantonese word “大”, and the desired output y_d is [0 1 0]. The last 100 training patterns correspond to the Cantonese word “註”, and the desired output y_d is [0 0 1]. The number of membership functions for the network is changed from 4 to 10. The number of iteration for training is 3000. After the training, the 60 testing patterns (3 Cantonese words \times 20) are used to test the performance of the self-structured NFN. The fitness values and the errors are the indices of the network performance. The best training fitness value and the number of parameters are obtained from 5 training processes, and are tabulated in Table 5-7. The testing fitness values and testing errors for each spoken word using the best parameter-set are tabulated in Table 5-8.

No. of membership functions	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 5-7. Fitness values and number of parameters of the self-structured NFNs.

No. of membership functions		4	5	6	7	8	9	10
fitness		0.8865	0.8765	0.9050	0.9008	0.9686	0.9611	0.8929
筆	/bat1/	0	0	0	0	0	0	0
大	/daai6/	15	15	14	15	3	4	16
註	/zyu3/	0	0	0	0	0	0	0

Table 5-8. Number of recognition errors and fitness values for the 3 Cantonese command words (20 testing patterns for each command).

For comparison, a traditional NFN trained by the improved GA is also used to do the same job. The number of iteration for training is 3000. The results are tabulated in Table 5-9 and Table 5-10.

<i>No. of membership functions</i>	4	5	6	7	8	9	10
<i>fitness</i>	0.9682	0.9504	0.9937	0.9903	0.9821	0.9922	0.9806
No. of parameters	172	215	258	301	344	387	430

Table 5-9. Fitness values and number of parameters of the traditional NFNs.

<i>No. of membership functions</i>		4	5	6	7	8	9	10
<i>Fitness</i>		0.8588	0.8844	0.8546	0.9144	0.9044	0.9333	0.9208
筆	/bat1/	0	0	0	0	1	1	0
大	/daai6/	20	14	20	10	9	8	9
註	/zyu3/	0	2	0	0	0	0	2

Table 5-10. Number of recognition errors and fitness values for the 3 Cantonese command characters using the traditional NFNs (20 testing patterns for each command).

From the results, it can be seen that the best recognition accuracy is obtained from the proposed approach when number of membership functions is 8. The corresponding fitness value is 0.9978 for training and the recognition error is 3 out of 60 testing patterns. By using the same number of membership function of the proposed approach to the traditional approach for comparison, the fitness value and the number of error for the traditional approach is 0.9821 and 10 respectively. The number of parameters of the self-structured NFN is 326, while 344 parameters are needed by the traditional NFN. This indicates that the introduction of link switches can reduce the size of the network structure but yet give a smaller recognition error with the data sets provided above.

5.4.1.3 Variable-parameter NFN

The Cantonese speech of the decimal digits '0' to '9' (10 classes) are to be recognized by the variable-parameter NFN discussed in Section 4.3.2, Chapter 4. 50 training patterns and 20 testing patterns for each digit are collected by a male speaker in a silent office environment. The 20 feature coefficients for each of the speech templates are obtained by the feature extraction methodology discussed early. The network has 20 inputs and 10 outputs. The improved GA is employed to train the variable-parameter NFN in order to maximize the fitness function of (3.5) in Chapter 3, where

$$err = \frac{\sum_{t=1}^{500} \frac{\|y_d(t) - y(t)\|^2}{10}}{500} \quad (5.10)$$

The total number of training patterns is 500. The first 50 training patterns correspond to the Cantonese digit '1', and the desired network output y_d is $[1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]$. Similarly, the next 50 training patterns correspond to the Cantonese digit '2', and the desired network output y_d is $[0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]$, and so on. Different numbers of membership functions are tried for the tuner and classifier NFNs. They are (3,5), (4,5), (5,5), (5,4) and (5,3), where the first number in the brackets is the number of membership functions used in the tuner NFN, and the second number is the number of membership functions used in the classifier NFN. The number of iteration for training is 50000. After the training, 200 testing patterns (10 Cantonese digits \times 20) are used to test the performance of the variable-parameter NFN. The fitness values and the errors are the indices of the network performance. The training and testing fitness values obtained from the best parameter-set after 5 runs of training are tabulated in Table 5-11. The numbers of errors for testing the digit speech are tabulated in Table 5-12.

No. of membership functions	(3,5)	(4,5)	(5,5)	(5,4)	(5,3)
Number of parameters	470	560	650	560	470
(50 training patterns for each digit)					
Fitness value	0.9968	0.9963	0.9995	0.9969	0.9940
(20 testing patterns for each digit)					
Fitness value	0.9818	0.9855	0.9952	0.9854	0.9675

Table 5-11. Fitness values under different combinations of numbers of membership functions in the proposed NFN.

		Membership functions combination				
Digit	Syllable	(3,5)	(4,5)	(5,5)	(5,4)	(5,3)
0 (零)	/ling4/	5	5	0	13	4
1 (一)	/jat1/	1	1	1	1	3
2 (二)	/ji6/	5	3	0	1	1
3 (三)	/saam1/	2	1	1	0	3
4 (四)	/sei3/	0	0	0	0	0
5 (五)	/ng5/	2	1	0	0	1
6 (六)	/luk6/	0	2	0	3	15
7 (七)	/cat1/	5	3	0	0	13
8 (八)	/baat3/	1	1	1	0	1
9 (九)	/gau2/	1	0	0	1	1

Table 5-12. Number of recognition errors for the Cantonese digits '0 - 9' given by the variable-parameter NFN (20 testing patterns for each digit).

For comparison, traditional NFNs with different number of membership functions trained by the improved GA are also used to perform the recognition. The number of iteration is 50000. The results are summarized in Tables 5-13 and Table 5-14.

<i>No. of membership functions</i>	9	10	11	12	13
<i>No. of parameters</i>	450	500	550	600	650
(50 training patterns for each digit)					
<i>Fitness value</i>	0.9891	0.9930	0.9655	0.9957	0.9962
(20 testing patterns for each digit)					
<i>Fitness value</i>	0.9626	0.9764	0.9895	0.9890	0.9932

Table 5-13. Fitness values given by the traditional neural-fuzzy network.

		<i>Number of membership functions</i>				
Digit	Syllable	9	10	11	12	13
0 (零)	/ling4/	9	5	10	1	1
1 (一)	/jat1/	2	2	2	1	1
2 (二)	/ji6/	3	2	2	1	1
3 (三)	/saam1/	1	3	1	1	0
4 (四)	/sei3/	0	0	2	1	0
5 (五)	/ng5/	2	1	0	0	2
6 (六)	/luk6/	13	4	2	4	0
7 (七)	/cat1/	14	4	0	0	0
8 (八)	/baat3/	1	0	1	1	1
9 (九)	/gau2/	2	1	2	0	0

Table 5-14. Number of recognition errors for the Cantonese digits '0 - 9' given by the traditional neural-fuzzy network (20 testing patterns for each digit).

From Table 5-12 and Table 5-14, it can be seen that only 3 misclassified patterns are detected out of 200 testing patterns from the best network (5 tuner NFNs and 5 classifier NFNs), while 6 errors occur on using the best traditional NFN (13 membership functions). The number of parameters of both networks is 650.

In order to show the merit of the improved GA in training the proposed neural-fuzzy network, the conventional back-propagation training algorithm has been used to train the same network. The numbers of membership function are (5, 5), which produce the best recognition performance as reported before. The number of iteration and the fitness value for the back-propagation training algorithm are 50000

and 0.8 respectively. The convergence curves obtained from the best parameter-set after 5 runs of the training process is shown in Fig. 5-10.

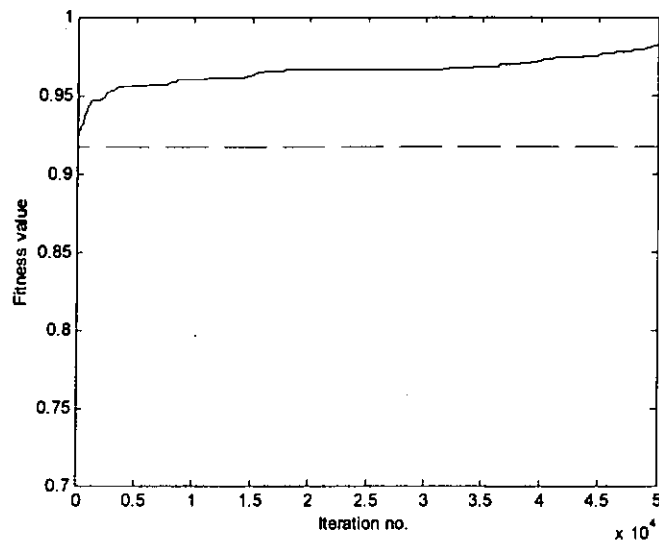


Fig. 5-10. Comparison of convergence between the improved GA (solid-line) and the back-propagation (dotted-line).

As shown in Fig. 5-10, the network trained by the back-propagation algorithm saturated rapidly with a lower fitness value than the improved GA. This indicates that the initial parameter of the back-propagation training scheme is not good, which traps the network in a local optimum point during the training process. The convergence curves demonstrate that the improved GA is better than the back-propagation in learning the training data sets provided.

5.4.1.4 Variable-parameter NN

Five classes of Cantonese characters, /bat1/ (筆), /daai6/ (大), /zyu3/ (註), /soeng5/ (上) and /haa5/ (下), are used to execute the “Pen”, “Zoom”, “Annotation”, “Page Up” and “Page Down” functions in the eBook reader. The approach discussed in Section 4.3.3, Chapter 4, is employed to recognize these five Cantonese characters. 500 speech patterns (100 patterns for each word) are used as the training

templates, and 150 speech patterns (30 patterns for each word) are used as the testing patterns. The speech patterns are obtained by a male speaker in a silent office environment. The speech signals are pre-processed to produce the feature coefficients using the feature extraction methodology discussed early. They will be used as the inputs of the proposed network.

The proposed variable-parameter NN has 5 inputs and 5 outputs. The training process is performed by networks with two to five hidden nodes, i.e. $h_1 = 2, 3, 4$, or 5. The improved GA is used as the training algorithm. The trained parameters $[\mu_{ij} \ \mu_j \ \omega_{jg} \ \omega_g \ \lambda_{jk} \ \lambda_k]$ for all i, j, g, k form the chromosomes of the GA process. The number of iteration for training is 2000 and 5 runs have been done for each training process. The lower and upper bounds of the parameter values are chosen to be -10 and 10 respectively. The initial values of the parameters are generated randomly before the training process. The training and the testing results obtained from the best parameter-set out of the 5 runs are tabulated in Table 5-15.

h_1	2	3	4	5
No. of network parameters	39	58	79	102
Training (500 patterns)				
Fitness value	0.9934	0.9974	0.9981	0.9985
Testing (150 patterns)				
Fitness value	0.9829	0.9940	0.9925	0.9943
Error rate provided by the network (%)	3.3333	1.3333	1.3333	0.6667

Table 5-15. Results given by the variable-parameter NNs.

For comparison purpose, a traditional 3-layer fully-connected feed-forward neural network trained by the improved GA is also used to do the same job. The numbers of hidden nodes of the traditional neural network are chosen to be 3, 5, 7 and 9 such that the numbers of parameters are more or less the same as those of the

proposed variable-parameter NNs. The training and testing results are tabulated in Table 5-16.

No. of hidden nodes	3	5	7	9
No. of network parameters	38	60	82	104
Training (500 patterns)				
Fitness value	0.9778	0.9841	0.9841	0.9849
Testing (150 patterns)				
Fitness value	0.9632	0.9463	0.9493	0.9685
Error rate provided by the network (%)	6	18.6667	15.3333	4.6667

Table 5-16. Results given by the traditional neural networks.

From Table 5-15 and Table 5-16, we can see that the best result provided by the variable-parameter NN is obtained when $h_1 = 5$. It produces a recognition rate of 99.3%, and the testing fitness value is 0.9943. On the other hand, the highest successful recognition rate offered by the traditional neural network is only 95.3% and the corresponding number of hidden node is 9. Besides, the proposed NN requires fewer parameters. As shown in Table 5-14, if we want a recognition error rate of 3.333%, the number of parameters of the proposed NN is 39. It is found that for the traditional NN, 104 parameters are needed to produce a near value of recognition error rate for the data sets provided above.

In order to show the merit of the improved GA, the back-propagation training algorithm will be employed to train the proposed NN for comparison. The convergence curves of the improved GA and the back-propagation under the proposed NN with 5 hidden nodes (which give the best result) are shown in Fig. 5-11. The training patterns for both algorithms are the same. The number of iteration and the fitness value for the back-propagation algorithm are 2000 and 0.8 respectively.

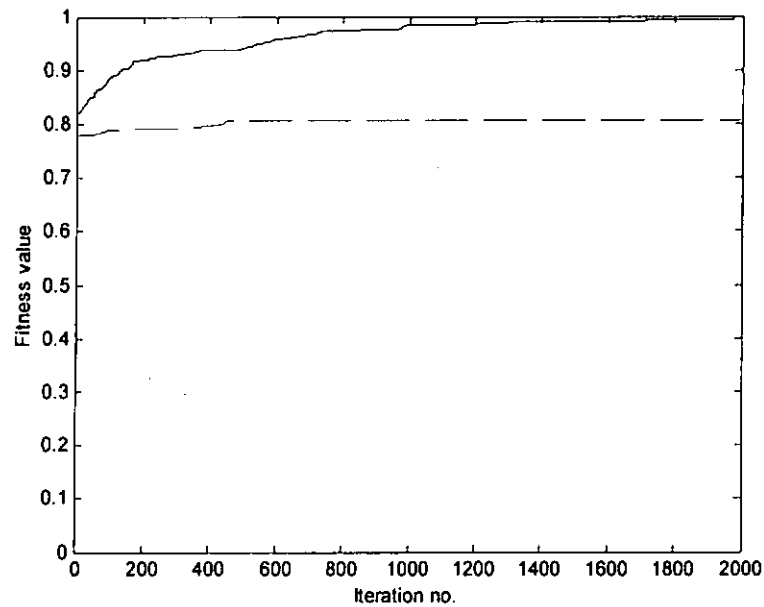


Fig. 5-11. Comparison of convergence between the improved GA (solid-line) and the back-propagation (dotted-line).

As can be seen from Fig. 5-11, the improved GA provides a higher fitness value than the back-propagation algorithm. The reason for the observation has been discussed on the previous sub-section. The improved GA is thus more suitable for the training patterns provided.

5.4.1.5 Comparison between the three proposed approaches

From the previous sub-sections, the three proposed networks are found to be able to give higher performance than the traditional neural networks. In order to find the best network for the application of static Cantonese speech recognition, the three proposed approaches are also evaluated through applying them to two applications: command speech application and digit speech application.

A. Cantonese-command speech recognition

To compare the three proposed approaches, they are tested based on the same application with the same conditions. The problem of recognizing five commands speeches of “/bat1/” (不), “/daai6 (大), “/zyu3/” (註), “/soeng5/” (上) and “/haa5/” (下) is used, which has been discussed in Sub-section 5.4.1.4. 100 training patterns and 30 testing patterns of each Cantonese character are used. The networks used have 5 inputs and 5 outputs. The numbers of iteration for training the self-structured NFN, the variable-parameter NFN, and the variable-parameter NN are 10000, 20000 and 2000 respectively. The results obtained from the best network parameters after 5 runs of the training process are summarized in Table 5-17 – 5-19.

No. of membership functions	4	5	6	7
No. of parameters after optimization	43	69	80	94
Training (500 patterns)				
Fitness value	0.9349	0.9895	0.9902	0.9956
Testing (150 patterns)				
Fitness value	0.9027	0.9612	0.9746	0.9754
Recognition rate (%)	78.6667	86.6667	90	92

Table 5-17. Results given by the self-structured NFNs.

h_1, h_2	(3,3)	(3,5)	(5,3)	(5,5)
No. of network parameters	105	155	155	225
Training (500 patterns)				
Fitness value	0.9778	0.9813	0.99	0.9901
Testing (150 patterns)				
Fitness value	0.9430	0.9456	0.9583	0.9607
Recognition rate (%)	80.6667	83.3333	84.67	90

Table 5-18. Results given by the variable-parameter NFNs.

h_1	2	3	4	5
No. of network parameters	39	58	79	102
Training (500 patterns)				
Fitness value	0.9934	0.9974	0.9981	0.9985
Testing (150 patterns)				
Fitness value	0.9829	0.9940	0.9925	0.9943
Recognition rate (%)	96.6667	98.6667	98.6667	99.3333

Table 5-19. Results given by the variable-parameter NNs.

Comparing the recognition accuracy, the best results given by the self-structured NFN, the variable-parameter NFN, and the variable-parameter NN are 92%, 90% and 99.3% respectively. Their corresponding numbers of parameters of the networks are 94, 225 and 102 respectively. The results demonstrate that all the 3 proposed approaches can be successfully applied to recognize the 5 Cantonese-speech commands with a recognition accuracy of over 90%. In addition, the variable-parameter NN requires the smallest number of iteration and yet gives the highest recognition accuracy for these application data sets. The specific structure of the variable-parameter NN has improved the learning ability in this application example.

B. Cantonese-digit speech recognition

Thirteen Cantonese speeches of the numbers “0 to “10 , “12 and “20 are used to test the performance of the three proposed approaches. The Cantonese-number “0 to “10 are single-syllable Cantonese speeches while, the Cantonese-number “12 and “20 are the multi-syllable Cantonese speeches, so the feature extraction stage for these two digits are different from the single-syllable numbers. It will be explained in Section 5.4.2. 70 patterns of each Cantonese-number (50 for training and 20 for testing) are obtained to do the evaluation. The speech patterns are obtained from a

male speaker in a silent office environment. The networks have 20-inputs-10-outputs for the eleven single-syllable numbers and 60-inputs-3-outputs for the three multi-syllable numbers. In order to discriminate the single-syllable and double-syllable speeches during testing, a decision stage has been employed. The discrimination is done by checking the size of the feature vector. If the size of the feature vector of the testing speech is equal to 20, the speech is classified as a single-syllable speech; otherwise, it is classified as a double-syllable speech.

The numbers of iteration for training the self-structured NFN, the variable-parameter NFN, and the variable-parameter NN are 13000, 50000 and 10000 respectively. The results given by the three proposed networks with the best network parameters after 5 runs of the training process are tabulated in Table 5-20 – Table 5-25.

No. of membership functions		8	9	10	11	12
Optimized no. of parameters for "0~10 / 12&20"		388/969	426/1036	486/1209	528/1273	573/1395
Number	Syllable	No. of recognition errors				
0 (零)	/ling4/	20	7	2	1	1
1 (一)	/jat1/	1	0	0	1	0
2 (二)	/ji6/	0	0	1	1	0
3 (三)	/saam1/	20	0	0	2	2
4 (四)	/sei3/	0	0	0	0	0
5 (五)	/ng5/	20	19	0	0	1
6 (六)	/luk6/	3	4	4	4	3
7 (七)	/cat1/	1	20	1	3	6
8 (八)	/baat3/	0	19	0	0	0
9 (九)	/gau2/	0	0	2	1	0
10 (十)	/sap6/	0	0	0	0	0
12 (十二)	/sap6/ /ji6/	0	0	0	0	0
20 (二十)	/ji6/ /sap6/	0	0	0	0	0

Table 5-20. Number of recognition errors given by the self-structured NFNs.

N	8	9	10	11	12
Training Fitness	0.9850	0.9850	0.9993	0.9996	0.9984
Testing Fitness	0.9830	0.9796	0.9966	0.9947	0.9939
Recognition Performance (%)	75	73.5	96.2	95	95

Table 5-21. Performance of the self-structured NFNs.

h_1, h_2		(3,5)	(4,5)	(5,5)	(5,4)	(5,3)
No. of parameters for “0~10 / 12&20		470/1005	560/1140	650/1275	560/1140	470/1005
Number	Syllable	No. of recognition error				
0 (零)	/ling4/	5	5	0	13	4
1 (一)	/jat1/	1	1	1	1	3
2 (二)	/ji6/	5	3	0	1	1
3 (三)	/saam1/	2	1	1	0	3
4 (四)	/sei3/	0	0	0	0	0
5 (五)	/ng5/	2	1	0	0	1
6 (六)	/luk6/	0	2	0	3	15
7 (七)	/cat1/	5	3	0	0	13
8 (八)	/baat3/	1	1	1	0	1
9 (九)	/gau2/	1	0	0	1	1
10 (十)	/sap6/	0	0	0	0	0
12 (十二)	/sap6/ /ji6/	0	0	0	0	0
20 (二十)	/ji6/ /sap6/	0	0	0	0	0

Table 5-22. Number of recognition errors given by the variable-parameter NFNs.

h_1, h_2	(3,5)	(4,5)	(5,5)	(5,4)	(5,3)
Training Fitness	0.9978	0.9977	0.9992	0.9980	0.9965
Testing Fitness	0.9897	0.9924	0.9965	0.9916	0.9824
Recognition Performance (%)	91.5	93.5	98.8	92.7	83.8

Table 5-23. Performance of the variable-parameter NFNs.

h_1		6	7	8	9	10
No. of parameters for "0~10 / 12&20"		252/443	299/523	348/605	399/689	452/775
Number	Syllable	No. of recognition error				
0 (零)	/ling4/	4	0	3	0	2
1 (一)	/jat1/	0	0	0	0	0
2 (二)	/ji6/	0	1	0	0	0
3 (三)	/saam1/	1	0	1	1	0
4 (四)	/sei3/	0	0	0	0	0
5 (五)	/ng5/	1	0	2	0	1
6 (六)	/luk6/	0	6	0	1	1
7 (七)	/cat1/	1	1	0	1	1
8 (八)	/baat3/	0	0	0	0	0
9 (九)	/gau2/	2	1	1	1	1
10 (十)	/sap6/	0	0	0	0	0
12 (十二)	/sap6/ /ji6/	0	0	0	0	0
20 (二十)	/ji6/ /sap6/	0	0	0	0	0

Table 5-24. Number of recognition errors given by the variable-parameter NNs.

h_1	6	7	8	9	10
Training Fitness	0.9978	0.9984	0.9985	0.9995	0.9995
Testing Fitness	0.9956	0.9951	0.9965	0.9979	0.9966
Recognition performance (%)	96.5	96.5	97.3	98.5	97.7

Table 5-25. Performance of the variable-parameter NNs.

From Table 5-20 to Table 5-25, we can see that the highest recognition rate given by the self-structured NFN, the variable-parameter NFN, and the variable-parameter NN are 96.2%, 98.8% and 98.5% respectively. The corresponding numbers of parameters are 486/1209, 650/1275 and 399/689, where the first number is the number of parameters used for recognizing the Cantonese-number "0" to "10" and the second number is the number of parameters used for recognizing the Cantonese-number "12" and "20". The results indicate that the two variable-parameter networks can well recognize the 13 Cantonese-digit speeches with

over 96% accuracy under the data sets provided. One interesting observation from the comparison results is that the recognition accuracy of the Cantonese-number “4” and “5” is higher than that of the other nine single-syllable numbers. This is because there is no similar vowel of these two digits in these 11 numbers. Misclassifications occur usually on “7” and “8” because of their common vowel “/at/” that is also found in “1”. Besides that, the two double-syllable Cantonese numbers produced high recognition accuracy because a decision stage that discriminates the double-syllable Cantonese numbers from the single-syllable Cantonese numbers has been added. The two double-syllable Cantonese numbers can be classified accurately because the information of “10” and “20” are located differently in the feature vectors of “12” and “20”.

The associative memory inside these networks can exhibit higher learning ability than the self-structured NFN. Considering the number of iteration for training, the variable-parameter NN shows the lowest value. Hence, it can be seen that the variable-parameter NN performs the best among the 3 proposed networks on recognizing mono-syllabic Cantonese speech patterns mentioned in this thesis. The above observation concurs with the qualitative comparisons indicated by Table 4-1 and Table 4-2 of Chapter 4.

5.4.2 Dynamic Pattern Classification for Cantonese Speech Recognition

The results on using the dynamic pattern classification for Cantonese-digit speech recognition will be reported in this sub-section. The dynamic variable-parameter NN described in Section 4.4.1, Chapter 4, is used to recognize 11 single-syllable and 2 double-syllable Cantonese-digit speeches. Owing to the need of considering the speech dynamic properties, the feature extraction and classification processes are different from those of the static pattern classification approach

mentioned in Section 5.4.1. The operation of the dynamic pattern classification for Cantonese-digit speech recognition is shown in Fig. 5-12.

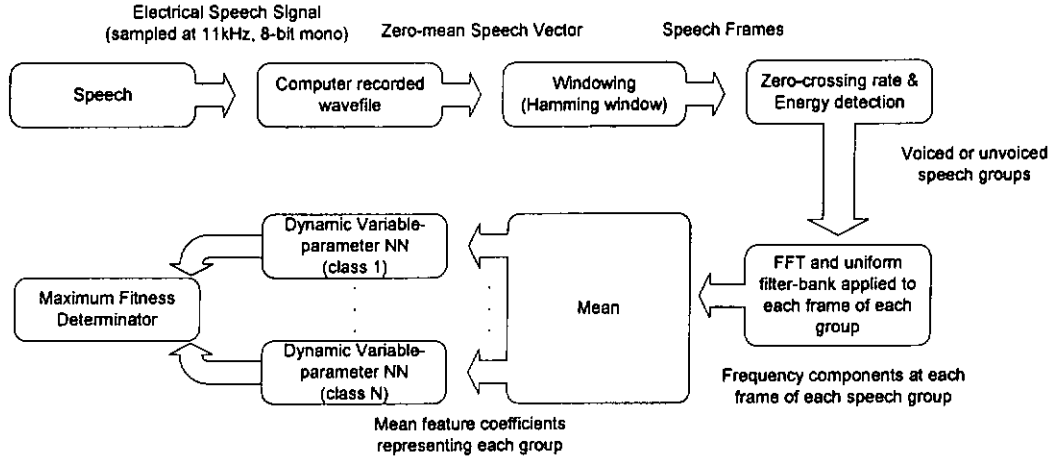


Fig. 5-12. Block diagram of the dynamic pattern classification for the Cantonese-digit speech recognition system.

5.4.2.1 Speech feature extraction methodology

The Cantonese speeches are recorded in the same way as mentioned in Section 5.4.1.1. The speeches are formatted in 8-bit PCM format sampled at 11 kHz. The obtained time-domain speech vector $s = [s(1) \ s(2) \ \dots \ s(no_sample)]$ is then windowed by the 128-sample sliding Hamming windows with 50% overlap to form speech frames. The zero-crossing rate and speech energy have to be found in order to determine the frames are voiced (or sonorant) or unvoiced (or non-sonorant). A speech frame with a high zero-crossing rate but low speech energy level is defined as the non-sonorant speech frame. Conversely, a sonorant speech frame will have a high speech energy level but low zero-crossing rate. The process is defined as follows.

$$E_n = \sum_{\tau=1}^{128} s(m_n + \tau)^2, \quad n = 1, 2, \dots, no_frame. \quad (5.11)$$

$$Z_n = \frac{1}{2 \times 128} \sum_{\tau=1}^{127} [s_s(m_n + \tau) - s_s(m_n + \tau + 1)], \quad n = 1, 2, \dots, no_frame. \quad (5.12)$$

$$m_n = [64(n-1)], \quad n = 1, 2, \dots, no_frame. \quad (5.13)$$

$$s_s(\tau) = \begin{cases} 1, & \text{when } s(\tau) \geq 0 \\ -1, & \text{when } s(\tau) < 0 \end{cases}, \quad \tau = 1, 2, \dots, no_sample. \quad (5.14)$$

where E_n denotes the speech energy of the n -th frame; m_n denotes the starting index of the n -th frame; no_frame denotes the number of frames for a speech signal; Z_n denotes the zero-crossing rate of the n -th frame; no_sample denotes the number of time samples of the speech (i.e. the number elements of the speech vector). As shown in Fig. 5-13(i), G_1 , G_2 and G_3 indicate the non-sonorant, sonorant and non-sonorant region of a single-syllable Cantonese-digit speech. The starting speech frame of a new group begins when the speech energy curve and the zero-crossing curve intersect, provided that the group has more than six speech frames (i.e. a duration longer than 40ms); otherwise, no group change will take place. Therefore, the speech sample shown in Fig. 5-13(i) gets three groups. Five groups are obtained from a double-syllable Cantonese-number speech as shown in Fig. 5-13(ii). G_1 , G_3 and G_5 are the non-sonorant regions with high zero-crossing rate and low speech energy level. G_2 and G_4 are the sonorant regions with high speech energy level and low zero-crossing rate. Since a pause has been added after the first syllable, G_3 is obtained.

Fast Fourier Transform (FFT) and uniform filter-bank filtering are then performed to each speech frame. The feature coefficients of each speech frame are defined as follows.

$$\mathbf{Sf}_n = [Sf_n(1) \quad Sf_n(2) \quad \dots \quad Sf_n(128)] = \text{Re}\{FFT([wh(\tau)s_n(\tau)])\},$$

$$1 \leq \tau \leq 128 \quad (5.15)$$

$$c_n^\beta = \frac{20 \log_{10} \sum_{l=1}^{\alpha} Sf_n(\rho_\beta + l) wt(\rho_\beta + l)}{\alpha}, \quad \beta = 1, 2, \dots, no_filter \quad (5.16)$$

$$wt(l) = \begin{cases} \frac{2l-1}{\alpha}, & 1 \leq l \leq \frac{\alpha}{2} \\ \frac{2(\alpha-l+1)}{\alpha}, & \frac{\alpha}{2} + 1 \leq l \leq \alpha \end{cases} \quad (5.17)$$

$$\alpha = \text{floor}\left(\frac{128}{no_filter}\right) \times 2 \quad (5.18)$$

$$\rho_\beta = 0.5\alpha(\beta-1), \quad \beta = 1, 2, \dots, no_filter \quad (5.19)$$

$$\mathbf{D}_n = [c_n^2 - c_n^1 \quad c_n^3 - c_n^2 \quad \dots \quad c_n^{no_filter} - c_n^{no_filter-1}] \quad (5.20)$$

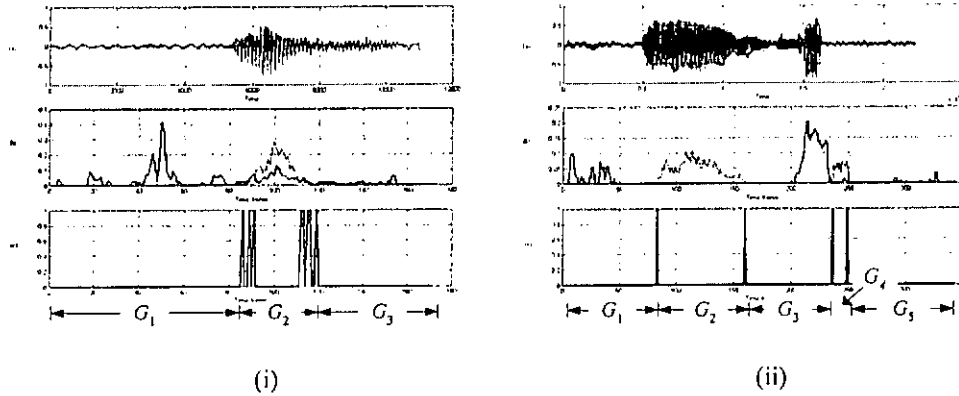
where \mathbf{Sf}_n denotes the frequency spectrum of the n -th speech frame; $\text{Re}\{\cdot\}$ denotes the real part of the argument vector; $FFT(\cdot)$ denotes the fast Fourier transform function; $s_n = [s_n(1) \quad s_n(2) \quad \dots \quad s_n(128)]$ denotes the n -th speech frame in time-domain; $wt(\cdot)$ denotes the triangular window; α denotes the number of frequency components tackled by each band-pass filter; $\text{floor}(\cdot)$ denotes the floor function which is used to round-up a floating point number; no_filter denotes the number of band-pass filters; ρ_β denotes the starting index of the β -th band-pass filter; c_n^β denotes the mean power output from the β -th band-pass filter for the n -th frame; \mathbf{D}_n is a vector

formed by the magnitude differences between two consecutive band-pass filter outputs.

The neighbouring speech frames of the same nature, i.e. voiced/sonorant or unvoiced/non-sonorant frames, are grouped together as shown in Fig. 5-13. The mean feature coefficients of all the speech frames in the same group are calculated as follows.

$$\text{scoeff}_\eta = \frac{\sum_{n=1}^{G_\eta} \mathbf{D}_n}{G_\eta}, \eta=1,2, \dots, \text{no_group} \quad (5.21)$$

where scoeff_η denotes η -th speech feature group; G_η denotes the number of speech frames in the η -th group; no_group denotes the number of groups that can be segmented from the speech. Thus, the acoustic feature sequence of the speech can be written as $\mathbf{Sp} = [\text{scoeff}_1 \text{scoeff}_2 \dots \text{scoeff}_{\text{no_group}}]$, and each element vector is then normalized as input to the dynamic variable-parameter neural network in sequence to do the speech recognition.



- (a) Time-domain speech waveform.
- (b) Zero-crossing rate (solid-line) and speech energy (dotted-line) plots.
- (c) Voiced/sonorant and unvoiced/non-sonorant speech groups.

Fig. 5-13. Speech acoustic feature groups of (i) a single-syllable Cantonese speech, (ii) a double-syllable Cantonese speech.

5.4.2.2 Dynamic variable-parameter NN

Eleven single-syllable Cantonese-number speeches (“0 to “10) and two double-syllable Cantonese-number speeches (“12 and “20) are used to test the performance of the proposed network. The double-syllable Cantonese-numbers are mainly used to test the sequential classification ability of the recurrent neural network. The two double-syllable speeches (“12 and “20) are formed by two single-syllable speeches (“2 and “10) in different sequence. Each speech is recorded by a single male speaker in a silent office. 20 training patterns and 50 testing patterns of each speech are used by the network.

One network is used to recognize one class of the Cantonese speech. The configuration of the recognizer is similar to that shown in Fig. 3-3 of Chapter 3. Each network has 24 inputs and 12 outputs. The network inputs are the 12 elements of the current speech feature group and the 12 feedback signals of the previous-state network outputs. The number of recurrent loops is determined by the number of acoustic feature groups of the speech. The initial feedback values of the recurrent neural network are all zero. The improved GA is used to train the network based on the chromosome $[\mu_{ij} \ \mu_j \ \omega_{jg} \ \omega_g \ \mu_{ij}^{ff} \ \mu_{ij}^{fb} \ \lambda_{jk} \ \lambda_k]$ for all i, j, g, q, k . The objective of the training process is to maximize the fitness function value as given by (3.5), (4.12) and (4.13) of the previous chapters. The upper and lower bound of all the parameters inside the chromosome are 2.0 and -2.0 respectively. The number of iteration used for training is 2000. The initial values of the parameters inside the chromosomes are generated randomly. Different combinations of the number of hidden nodes for the tuner and classifier networks are tried in order to find the best performance. These combinations are (3, 5), (5, 5), (5, 10), (10, 10) (10, 12) and (12, 12), where the first number in the brackets is the number of hidden nodes of the TNN

(h_1), and the second number is the number of hidden nodes of the CRNN (h_2). The results for the 11 single-syllable Cantonese-number speeches and 2 double-syllable Cantonese-number speeches tabulated in Table 5-26 are obtained from the best network parameters after 5 runs of training process. The overall performance given by the proposed method is summarized in Table 5-27.

h_1, h_2		(3,5)	(5,5)	(5,10)	(10,10)	(10,12)	(12,12)
No. of parameters		259	299	509	634	728	782
Number	Syllable	Recognition rate (%)					
0 (零)	/ling4/	84	98	92	88	96	84
1 (一)	/jat1/	94	96	100	100	98	98
2 (二)	/ji6/	98	100	94	100	100	100
3 (三)	/saam1/	92	88	88	90	86	86
4 (四)	/sei3/	100	100	100	100	100	100
5 (五)	/ng5/	98	100	98	100	100	100
6 (六)	/luk6/	92	92	94	100	98	88
7 (七)	/cat1/	90	86	90	90	86	86
8 (八)	/baat3/	86	80	72	94	90	68
9 (九)	/gau2/	100	92	96	92	100	92
10 (十)	/sap6/	64	66	86	96	78	72
12 (十二)	/sap6/ /ji6/	98	90	92	98	86	78
20 (二十)	/ji6/ /sap6/	88	90	98	94	98	96

Table 5-26. Recognition rate for 11 Cantonese-number speeches and 2 double-syllable Cantonese-number speeches (test data).

h_1, h_2	(3,5)	(5,5)	(5,10)	(10,10)	(10,12)	(12,12)
Recognition rate for testing (%)	91.1	90.6	92.3	95.5	93.5	88.3
Recognition rate for training (%)	98.5	99.6	98.8	99.2	99.2	98.8

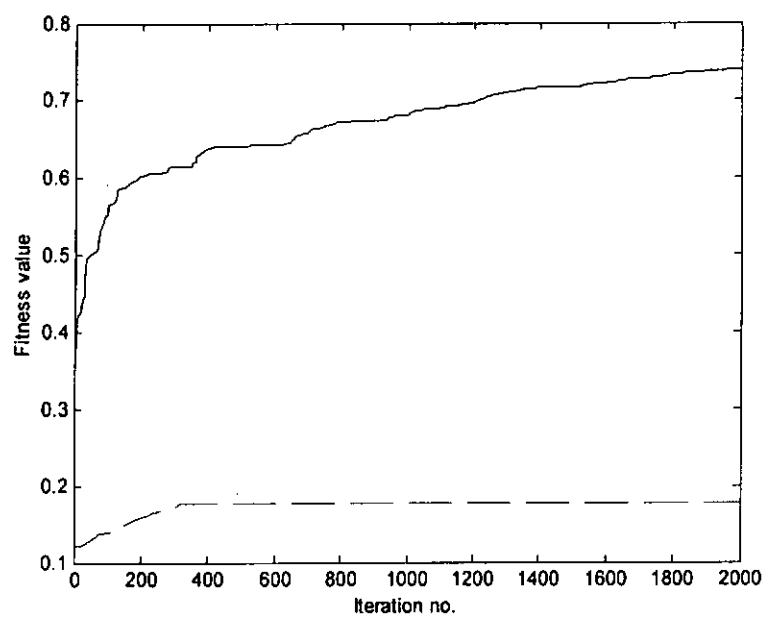
Table 5-27. Overall performance on testing and training for the 13 Cantonese-number speeches.

From Table 5-26 and Table 5-27, we can see that the best results are obtained when (h_1, h_2) = (10, 10). The recognition rate is 95.5% and the number of

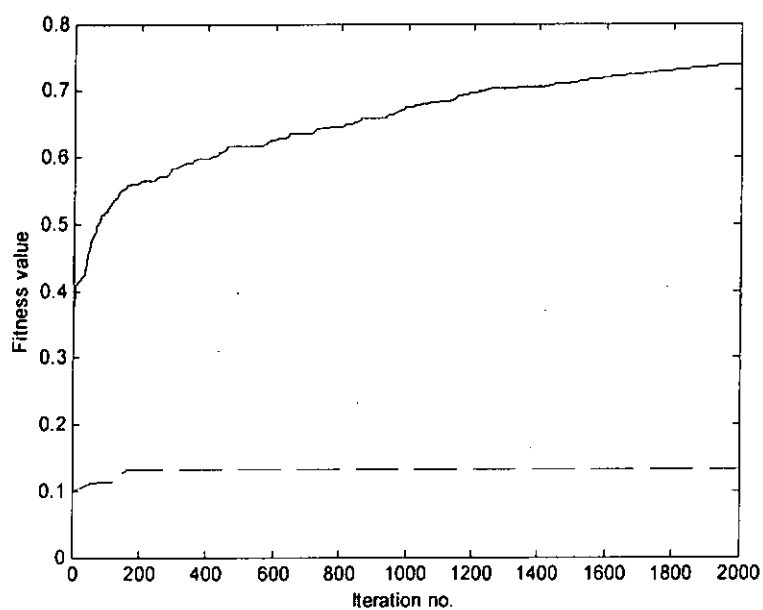
parameters is only 634. Even when the number of parameters is reduced to 259, i.e. 41% of that gives the best results, the recognition rate can still be over 91%. It should be noted that some Cantonese numbers have the same vowel. For example, the digits “1”, “7”, and “8” have the same vowel of /a/, and “6” and “9” have the same vowel of /u/. By using the proposed network, the speech patterns mentioned above can be discriminated from one another with a success rate of over 90%. Apart from that, the sequence of the speech syllables can also be discriminated by the proposed network. The recognition rates of four Cantonese -numbers “2”, “10”, “12”, and “20” are 100%, 96%, 98% and 94% respectively, despite the fact that their syllables are similar. In short, both the static and dynamic features of the Cantonese speech can be extracted and classified effectively by the proposed method.

It is interesting to see that when the numbers of hidden nodes are set to (10, 12) and (12, 12), the recognition rates drop to 93.5% and 88.3% respectively. The numbers of parameters (728 and 782 respectively) could be too large that some parameters might have introduced unnecessary information in that application example.

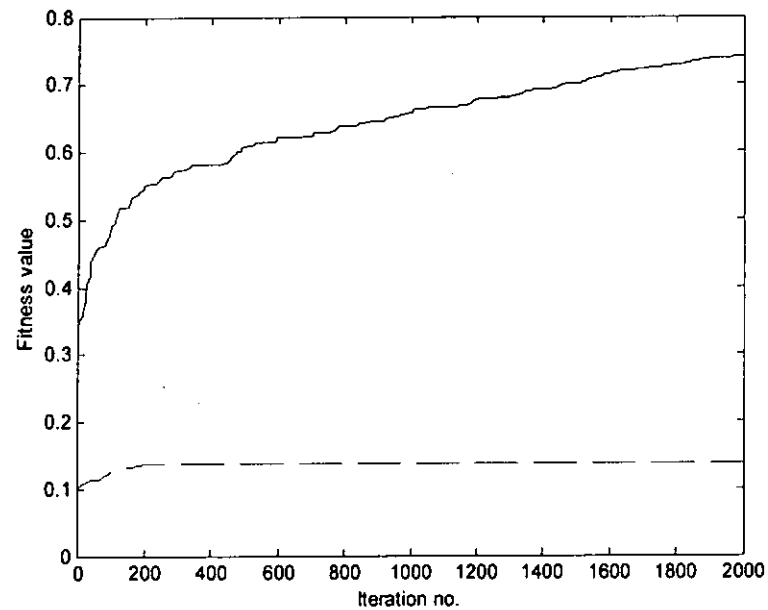
In order to elaborate the merit of the improved GA training algorithm to the dynamic variable-parameter NN, the back-propagation algorithm is used to train the same network as a comparison. The comparisons are made to three of the dynamic variable-parameter NNs, those for the digits 1, 7 and 8. These three digits have the same vowel and are easy to be misclassified. A good training fitness value for each digit can reduce the chance of misclassification. The convergence curves of the improved GA and the back-propagation are shown in Fig. 5.14. The number of iteration for the training is 2000. The number of hidden node combination for the dynamic variable-parameter NN is (10, 10).



Cantonese-speech digit 1.



Cantonese-speech digit 7.



Cantonese-speech digit 8.

Fig. 5-14. Comparison of convergence between the improved GA (solid-line) and the back-propagation (dotted-line).

The convergence curves in Fig. 5.14 demonstrate that the improved GA learns better than the back-propagation algorithm in the proposed network with the data sets given. By using the back-propagation algorithm, the fitness value saturates rapidly while the fitness values produced by the improved GA are increasing. Therefore, the improved GA is more suitable than the back-propagation in training the dynamic variable-parameter NN with the data sets provided in this thesis.

5.4.3 Comparison for Static and Dynamic Pattern Classification for Cantonese Speech Recognition

The variable-parameter NN that gives the best performance for the static pattern classification for Cantonese-digit speech recognition will be compared with the dynamic variable-parameter NN proposed in the previous sub-section. Comparison will be made in terms of the recognition ability for 13 Cantonese numbers and the

number of parameters used by the classification network.

Based on the characteristic of the static pattern classification for Cantonese-digit speech recognition mentioned in this thesis, one can say that it is not suitable for the multi-syllable speech recognition problem. The reason is because the feature extraction methods for the static and dynamic pattern classification approaches are different. However if we do want to use the static pattern classification approach to recognise multi-syllable speech, the feature extraction method for the dynamic pattern classification could be employed. Thus, the input vector of the Cantonese-speech for the static pattern classifier is formed by cascading the feature groups. Again, 13 variable-parameter NNs are used. The number of iteration for the training is 2000 and 5 runs are done for each training process. The results for the 13 Cantonese-numbers on using the best network parameters are listed in Tables 5-28 and 5-29.

h_1, h_2		5, 5	6, 6	7, 7	8, 8	9, 9	10, 10
No. of parameters for 0~9/12 & 20		443/707	530/842	619/979	710/1118	803/1259	898/1402
Number	Syllable	Recognition rate (%)					
0 (零)	/ling4/	72	72	80	78	80	68
1 (一)	/jat1/	94	90	92	94	92	92
2 (二)	/ji6/	98	100	100	100	98	98
3 (三)	/saam1/	70	76	72	74	62	82
4 (四)	/sei3/	100	100	100	96	100	100
5 (五)	/ng5/	100	98	98	98	100	100
6 (六)	/luk6/	96	94	94	96	96	92
7 (七)	/cat1/	96	92	92	94	92	92
8 (八)	/baat3/	88	78	88	90	90	88
9 (九)	/gau2/	64	84	78	78	88	90
10 (十)	/sap6/	96	92	84	96	100	100
12 (十二)	/sap6/ /ji6/	82	80	84	82	84	82
20 (二十)	/ji6/ /sap6/	100	100	100	100	100	100

Table 5-28. Recognition rate for the 13 Cantonese speeches (test data).

h_1, h_2	5, 5	6, 6	7, 7	8, 8	9, 9	10, 10
Recognition rate for testing (%)	88.9	88.9	89.4	90.5	90.9	91.1
Recognition rate for training (%)	94.6	95.8	96.2	96.9	97.3	98.1

Table 5-29. Overall performance on testing and training for the 13 Cantonese speeches.

From the results shown in Table 5-28 and Table 5-29, the best performance of the variable-parameter NN is produced when $(h_1, h_2) = (10, 10)$. The recognition accuracy of the variable-parameter NN at this setting is 91.1% and its training accuracy is 98.1%. The number of parameters used for the network is 898 for each single-syllable Cantonese-number speech network and 1402 for each double-syllable Cantonese-number speech network.

By comparing the recognition accuracy of the static and dynamic pattern classification approaches, the dynamic pattern classification approach produces higher accuracy and needs fewer network parameters. This is due to the fact that the size of the input vector for the static pattern classifier is large. Also, when comparing the network structure, the size of the feed-forward NN would be changed according to the size of the input vector. Therefore, two values of the number of network parameter are used. Besides that, since the feed-forward NN operates in parallel processing, all the elements in the speech feature vector are processed at the same instant. As a result, a larger network is needed. The above comparison shows that both the static and dynamic pattern classification can recognize the 13 Cantonese-number patterns used in this thesis with over 90% of accuracy.

5.4.4 Comparison for Static Pattern Classification, Dynamic Pattern Classification and Commercial Classification for Cantonese Speech Recognition

The performance of the static and dynamic pattern classification to obtain the best result reported before will be compared with that of the commercial classification product PenPower. All the testing patterns are the same speech files used the previous tests. The training procedure of the PenPower is realized by reading three articles provided by the software. After the three articles have been read, the software learns the speaking properties of the speaker. Then, the testing procedure is done by playing back the recorded sound files for the software to recognize. 50 testing patterns are used for the test and the results of the 3 Cantonese speech recognition approaches are listed in Table 5-30.

Cantonese speech	Syllable	Variable-parameter NN	Dynamic variable-parameter NN	Commercial
0 (零)	/ling4/	68	88	88
1 (一)	/jat1/	92	100	90
2 (二)	/ji6/	98	100	100
3 (三)	/saam1/	82	90	88
4 (四)	/sei3/	100	100	100
5 (五)	/ng5/	100	100	100
6 (六)	/luk6/	92	100	100
7 (七)	/cat1/	92	90	90
8 (八)	/baat3/	88	94	96
9 (九)	/gau2/	90	92	90
10 (十)	/sap6/	100	96	96
12 (十二)	/sap6/ /ji6/	82	98	100
20 (二十)	/ji6/ /sap6/	100	94	98
Overall Performance (%)		91.1	95.5	95.2

Table 5-30. Performance comparison between the 2 proposed approaches and the commercial approach.

The recognition accuracy and the training scheme are taken into consideration. The overall recognition accuracy for the variable-parameter NN, the dynamic variable-parameter NN and the commercial recognition software are 91.1%, 95.5% and 95.2% respectively. This indicated that the three Cantonese-number speech recognition approaches can successfully recognize the testing speeches. Comparing the training schemes of the three recognition approaches, the commercial recognition software provides high flexibility. This is because users are only required to read 3 articles to enable the software recognizing all the Cantonese characters/words. The two proposed approaches, on the other hand, require a vocabulary database (reference templates) for each testing Cantonese speech.

CHAPTER 6

CONCLUSION

6.1 Achievements

An electronic book (eBook) reader program has been developed to provide a multimedia platform for eBook materials. Two main features in an eBook environment are considered in this thesis: graffiti interpretation and Cantonese speech recognition. Neural network techniques are applied in these two applications.

For the graffiti interpretation applications, a graffiti-to-command interpreter and a graffiti-to-character interpreter have been developed. The graffiti command interpreter employs a proposed self-structured neural network. Five eBook commands, namely “Pen”, “Zoom”, “Annotation”, “Page Up”, and “Page Down”, have been used to test the self-structured neural network. With the self-structured neural network interpretation scheme, the structure of the neural network can be tuned during the training process. Thus, the network structure can be reduced when the network operates. Besides that, the number of network parameters of the proposed neural network can be smaller than that of the traditional neural network. The recognition performance of the self-structured neural network is found to have a good result in our application examples.

A self-structured neural-fuzzy network has been developed to interpret graffiti characters (digit “0” to digit “9”, “backspace”, “carriage return”, and “space”) for the eBook applications. By employing the idea of the self-structuring again to a

neural-fuzzy network, the number of rules and the membership functions can be reduced. A traditional neural-fuzzy network has also been applied for comparison. The results show that the self-structured neural-fuzzy network can give good results in the application examples of this thesis.

The above two proposed approaches have been evaluated by using the same application (recognition of characters). The results show that both approaches can successfully recognize all the testing patterns with high accuracy.

For the Cantonese speech recognition applications, the Cantonese-command speech recognizer and the Cantonese-number speech recognizer have been considered. Totally four approaches of Cantonese speech recognition have been investigated. The first approach for realizing the Cantonese-command speech recognition uses the self-structured neural-fuzzy network. By implementing the self-structured neural-fuzzy network again to recognize three Cantonese-command speeches (“/bat1 -筆, “/daai6 -大 and “/zyu3 -註), a good performance can be obtained. This indicates that the proposed approach can be used in different pattern-matching applications mentioned in this thesis.

The second approach for the Cantonese-command speech recognition employs associative memory inside a proposed variable-parameter neural network. Five Cantonese speech commands: “/bat1/” (不), “/daai6 (大), “/zyu3/” (註), “/soeng5/” (上), and “/haa5/” (下), have been used. With the specific structure of the proposed network, the associative memory serves as a tuner neural network for another classifier neural network. Because both the tuner neural network and the classifier neural network process the same input, the parameters of the classifier neural network can be adjusted by the tuner neural network according to the input patterns during the operation. The search domain of the network can be enlarged. The effects to the recognition performance brought about by the associative memory

inside the variable-parameter neural network have been reported. The results are compared with those from a traditional neural network.

For the Cantonese-number speech recognition, a variable-parameter neural-fuzzy network has been employed. Ten single-syllable Cantonese digits (digit “0 to “9) have been used. These ten single-syllable Cantonese digits are then used again, together with the single-syllable Cantonese-number “10 and the double-syllable Cantonese-number “12 and 20 , in a dynamic Cantonese-speech recognition system. This system uses a dynamic variable-parameter neural network. Both approaches utilized the idea of variable parameter again. The static approach uses a feed-forward neural-fuzzy network as the classifier. The dynamic approach uses a recurrent neural network as the classifier. Both approaches can achieve a better performance than the traditional approaches. Results have verified that the proposed approaches can have a good learning and recognition abilities of the Cantonese speech patterns provided in this thesis.

The first three approaches have been evaluated with two application examples: recognizing five Cantonese-command speech and thirteen single-syllable Cantonese-number speech. The results have shown that the variable-parameter neural network performs very well in both applications.

On comparing the variable-parameter neural network to the dynamic variable-parameter neural network in terms of recognition ability to the thirteen Cantonese-number speeches, results show that the dynamic network exhibits a superior performance.

6.2 Further Research

Large vocabulary and continuous-speech recognition approaches can be further investigated. By using of the dynamic variable-parameter neural network reported in this thesis, continuous speeches could be recognized. On further elaborating the idea of associative memory, the number of recognized speeches could be increased. In order to reduce the recognition load and complexity of the classification network, the segmentation algorithm for the continuous speeches have to be explored. Besides, introducing a discrimination network as a decision maker might further improve the recognition performance.

APPENDIX

IMPROVED GENETIC ALGORITHM

A.1 Introduction

Genetic algorithm (GA) is a directed random search technique [4, 31] that is widely applied in optimisation problems [4, 31]. This 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 to find out the optimal solution globally over a domain. It has been applied in different areas such as fuzzy control [15, 40, 41], path planning, greenhouse climate control [34], modelling and classification [28], etc. A lot of research efforts have been spent to improve the performance of GA.

The standard GA process [42] is shown in Fig. A-1. 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. Fourth, genetic operations of crossover and mutation are performed. The produced offspring replaces their parents in the initial population. This GA process repeats until a user-defined criterion is reached. In this section, the standard GA is modified and new genetic operators are introduced to improve its performance. The improved GA process is shown in Fig. A-2. Its details will be given as follows.

A. Initial Population

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

$$P = \{p_1, p_2, \dots, p_{pop_size}\}, \quad (A.1)$$

$$p_i = [p_{i_1} \quad p_{i_2} \quad \dots \quad p_{i_j} \quad \dots \quad p_{i_{no_vars}}],$$

$$i = 1, 2, \dots, pop_size; j = 1, 2, \dots, no_vars, \quad (A.2)$$

$$para_{min}^j \leq p_{i_j} \leq para_{max}^j \quad (A.3)$$

where pop_size denotes the population size; no_vars denotes the number of variables to be tuned; p_{i_j} , $i = 1, 2, \dots, pop_size$; $j = 1, 2, \dots, no_vars$, are the parameters to be tuned; $para_{min}^j$ and $para_{max}^j$ are the minimum and maximum values of the parameter p_{i_j} respectively for all i . It can be seen from (A.1) to (A.3) that the potential solution set P contains some candidate solutions p_i (chromosomes). The chromosome 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,

$$fitness = f(p_i) \quad (A.4)$$

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 [42]. 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 :

$$q_i = \frac{f(\mathbf{p}_i)}{\sum_{k=1}^{pop_size} f(\mathbf{p}_k)}, i = 1, 2, \dots, pop_size \quad (A.5)$$

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

$$\hat{q}_i = \sum_{k=1}^i q_k, i = 1, 2, \dots, pop_size \quad (A.6)$$

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 $\hat{q}_{i-1} < d \leq \hat{q}_i$ ($\hat{q}_0 = 0$). 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 [22] 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 parent 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,

$$\mathbf{o}_{s_c}^1 = \begin{bmatrix} o_{s_1}^1 & o_{s_2}^1 & \cdots & o_{s_{no_vars}}^1 \end{bmatrix} = \frac{\mathbf{p}_1 + \mathbf{p}_2}{2} \quad (\text{A.7})$$

$$\mathbf{o}_{s_c}^2 = \begin{bmatrix} o_{s_1}^2 & o_{s_2}^2 & \cdots & o_{s_{no_vars}}^2 \end{bmatrix} = \mathbf{p}_{\max}(1-w) + \max(\mathbf{p}_1, \mathbf{p}_2)w \quad (\text{A.8})$$

$$\mathbf{o}_{s_c}^3 = \begin{bmatrix} o_{s_1}^3 & o_{s_2}^3 & \cdots & o_{s_{no_vars}}^3 \end{bmatrix} = \mathbf{p}_{\min}(1-w) + \min(\mathbf{p}_1, \mathbf{p}_2)w \quad (\text{A.9})$$

$$\mathbf{o}_{s_c}^4 = \begin{bmatrix} o_{s_1}^4 & o_{s_2}^4 & \cdots & o_{s_{no_vars}}^4 \end{bmatrix} = \frac{(\mathbf{p}_{\max} + \mathbf{p}_{\min})(1-w) + (\mathbf{p}_1 + \mathbf{p}_2)w}{2} \quad (\text{A.10})$$

$$\mathbf{p}_{\max} = \begin{bmatrix} para_{\max}^1 & para_{\max}^2 & \cdots & para_{\max}^{no_vars} \end{bmatrix} \quad (\text{A.11})$$

$$\mathbf{p}_{\min} = \begin{bmatrix} para_{\min}^1 & para_{\min}^2 & \cdots & para_{\min}^{no_vars} \end{bmatrix} \quad (\text{A.12})$$

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 $\mathbf{o}_{s_c}^1$ to $\mathbf{o}_{s_c}^4$, the one with the largest fitness value is used as the offspring of the crossover operation. The offspring is defined as,

$$\mathbf{o}_i \equiv \begin{bmatrix} o_{s_1} & o_{s_2} & \cdots & o_{s_{no_vars}} \end{bmatrix} = \mathbf{o}_{s_c}^{w_i} \quad (\text{A.13})$$

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

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 [42]. The offspring generated by these methods, however, may not be better than that from our approach. As seen from (A.7) to (A.10), the potential offspring after the crossover operation spreads over the domain. While (A.7) and (A.10) result in searching around the centre region of the domain (a value of w near to 1 in (A.10) can move $\mathbf{o}_{s_c}^4$ to be near $\frac{\mathbf{p}_1 + \mathbf{p}_2}{2}$), (A.8) and (A.9) move the potential offspring to be near the domain boundary (a small value of w in (A.8) and (A.9) can move $\mathbf{o}_{s_c}^2$ and $\mathbf{o}_{s_c}^3$ to be near \mathbf{p}_{\max} and \mathbf{p}_{\min} respectively).

2. Mutation

The offspring (A.13) 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. In general, various methods like boundary mutation, uniform mutation or non-uniform mutation [42] 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, a different process of mutation is proposed. The details are as follows. Every gene of the offspring \mathbf{o}_s of (A.13) will have a chance to mutate governed by a probability of mutation, $p_m \in [0 \ 1]$, 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 and updated instantly. The gene of the offspring of (A.13) is then mutated by:

$$\hat{o}_{s_k} = \begin{cases} o_{s_k} + \Delta o_{s_k}^U & \text{if } f(\mathbf{o}_s + \Delta \mathbf{o}_{s_k}^U) \geq f(\mathbf{o}_s - \Delta \mathbf{o}_{s_k}^L) \\ o_{s_k} - \Delta o_{s_k}^L & \text{if } f(\mathbf{o}_s + \Delta \mathbf{o}_{s_k}^U) < f(\mathbf{o}_s - \Delta \mathbf{o}_{s_k}^L) \end{cases}, k = 1, 2, \dots, no_vars \quad (\text{A.14})$$

where

$$\Delta o_{s_k}^U = w_{m_k} r (para_{\max}^k - o_{s_k}) \quad (\text{A.15})$$

$$\Delta o_{s_k}^L = w_{m_k} r (o_{s_k} - para_{\min}^k) \quad (\text{A.16})$$

$$\Delta \mathbf{o}_{s_k}^U = [0 \ 0 \ \dots \ \Delta o_{s_k}^U \ \dots \ 0] \quad (\text{A.17})$$

$$\Delta \mathbf{o}_{s_k}^L = [0 \ 0 \ \dots \ \Delta o_{s_k}^L \ \dots \ 0] \quad (\text{A.18})$$

$r \in [0 \ 1]$ is a randomly generated number; $w_{m_k} \in (0 \ 1]$ 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} is varied by the value of $\frac{\tau}{T}$, which is used for the fine-tuning purpose. T is the total number of iteration. In order to perform a local search, the value of weight w_{m_k} should become small as $\frac{\tau}{T}$ increases in order to reduce the significance of the mutation. Under this requirement, a monotonic decreasing function governing the w_{m_k} is proposed to be,

$$w_{m_i} = w_f \left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_r}} \geq 0 \quad (\text{A.19})$$

where $w_f \in [0 \ 1]$ and $w_r > 0$ are both variables to be chosen to determine the initial value and the decay rate respectively. For a large value of w_f , it can be seen from (A.15) and (A.16) that $\Delta o_{s_i}^U \approx w_f r (para_{\max}^k - o_{s_i})$ and $\Delta o_{s_i}^L \approx w_f r (o_{s_i} - para_{\min}^k)$ initially as $\left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_r}} \approx 1$ which ensure a large search space. When the value of $\left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_r}} \approx 0$, it can be seen that the values of $\Delta o_{s_i}^U$ and $\Delta o_{s_i}^L$ are small to ensure a small search space for fine-tuning.

E. Reproduction

The reproduction process will take place after the genetic operation. The new offspring will be evaluated using the fitness function of (A.4). 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 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. Hence, the mechanism of reaching the global optimum is kept.

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. the change of the fitness values between the current and the previous iteration is less than a given value, or a defined number of iteration has been reached.

A.2 Benchmark Tests

A suite of eight benchmark test functions [40] is used to test the performance of the improved GA. Many kinds of optimisation problems are covered by these benchmark test functions. They are divided into three main categories: unimodal functions, multimodal functions with only a few local minima, and multimodal functions with many local minima. Functions f_1 to f_4 are unimodal functions. Function f_1 is a sphere model which is probably the most widely used test function. It is smooth and symmetric. The performance on this function is a measure of the general efficiency of an algorithm. Function f_2 is a step function that is a representative of flat surfaces. Flat surfaces are obstacles for optimisation algorithms because they do not give any information about the search direction. Unless the algorithm has a variable step size, it can get stuck in one of the flat surfaces. Function f_3 is a quartic function, which is a simple unimodal function padded with noise. The Gaussian noise causes the algorithm never getting the same value at the same point. Many algorithms that do not do well in this function are due to the noisy data. Function f_4 is a Schwefel's problem. Each parameter affects the overall performance deeply. Functions f_5 to f_6 are multimodal functions with only a few local minima, and the dimension of each function is small. Function f_5 is a Shekel's foxholes function. Function f_6 is a Goldstein-price's function. Functions f_7 to f_8 are multimodal functions with many local minima, and the dimension of each function is comparatively large. Function f_7 is a generalized Rastrigin's function.

Function f_8 is a generalized Griewank function. The eight test functions are defined as follows,

$$f_1(\mathbf{x}) = \sum_{i=1}^n x_i^2, -5.12 \leq x_i \leq 5.12 \quad (\text{A.20})$$

where $n = 30$ and $\min(f_1) = f_1(\mathbf{0}) = 0$. The fitness function for f_1 is defined as,

$$fitness = \frac{1}{1 + f_1(\mathbf{x})}.$$

$$f_2(\mathbf{x}) = \sum_{i=1}^n \text{floor}(x_i + 0.5)^2, -5.12 \leq x_i \leq 5.12 \quad (\text{A.21})$$

where $\text{floor}(\cdot)$ is defined to round down the argument to an integer, $n = 30$ and $\min(f_2) = f_2(\mathbf{0}) = 0$. The fitness function for f_2 is defined as,

$$fitness = \frac{1}{1 + f_2(\mathbf{x})}.$$

$$f_3(\mathbf{x}) = \sum_{i=1}^n ix_i^4 + \text{random}[0, 1), -1.28 \leq x_i \leq 1.28 \quad (\text{A.22})$$

where $\text{random}(\cdot)$ is defined to generate a floating-point number between 0 and 1, $n = 30$ and $\min(f_3) = f_3(\mathbf{0}) = 0$. The fitness function for f_3 is defined as,

$$fitness = \frac{1}{1 + f_3(\mathbf{x})}.$$

$$f_4(\mathbf{x}) = \max_i \{x_i | 1 \leq i \leq n\}, -100 \leq x_i \leq 100 \quad (\text{A.23})$$

$n = 30$ and $\min(f_4) = f_4(\mathbf{0}) = 0$. The fitness function for f_4 is defined as,

$$fitness = \frac{1}{1 + f_4(\mathbf{x})}.$$

$$f_5(\mathbf{x}) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (x_i - a_{ij})^6} \right]^{-1}, \quad -65.536 \leq x_i \leq 65.536 \quad (\text{A.24})$$

$$\min(f_5) = f_5([-32, 32]) \approx 1,$$

$$\text{where } a_{ij} = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \cdots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \cdots & 32 & 32 & 32 \end{pmatrix}.$$

The fitness function for f_5 is defined as,

$$\text{fitness} = \frac{1}{f_5(\mathbf{x})}.$$

$$f_6(\mathbf{x}) = \left[1 + (x_1 + x_2 + 1)^2 \cdot (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \right] \cdot \left[30 + (2x_1 - 3x_2)^2 \cdot (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) \right], \quad -2 \leq x_1, x_2 \leq 2 \quad (\text{A.25})$$

$\min(f_6) = f_6([0, -1]) = 3$. The fitness function for f_6 is defined as,

$$\text{fitness} = \frac{1}{-2 + f_6(\mathbf{x})}.$$

$$f_7(\mathbf{x}) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10], \quad -5.12 \leq x_i \leq 5.12 \quad (\text{A.26})$$

$n = 30$ and $\min(f_7) = f_7(\mathbf{0}) = 0$. The fitness function for f_7 is defined as,

$$\text{fitness} = \frac{1}{1 + f_7(\mathbf{x})}.$$

$$f_8(\mathbf{x}) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1, \quad -600 \leq x_i \leq 600 \quad (\text{A.27})$$

$n = 30$ and $\min(f_8) = f_8(\mathbf{0}) = 0$. The fitness function for f_8 is defined as,

$$\text{fitness} = \frac{1}{1 + f_8(\mathbf{x})}.$$

The improved GA goes through the above eight test functions. The results are compared with those obtained by the published GAs with arithmetic, heuristic or one-point crossover and non-uniform mutation, depending on which one gives the best result in each time of iteration. For each test function, the population size is 10. All the simulation results are averaged ones out of 50 runs. All fitness functions are normalized to lie between 0 and 1 so that the best fitness value is equal to one ($fitness = 1$). The probability of acceptance (p_a) is set at 0.1 for all test functions. The simulation results (averaged, maximum and minimum fitness values, and the standard deviation) of f_1 to f_8 obtained by the improved GA and the published GA, and the number of iteration are tabulated in Table A-1. The parameter settings for them are tabulated in Table A-2. All settings are chosen by trial and error through experiments for good performance. The results of f_1 to f_8 obtained by the improved GA and the published GA are shown in Fig. A-2. It can be seen that the improved GA performs more efficiently, and provides a faster convergence than the published GA.

```
Procedure of the standard GA
begin
     $\tau \rightarrow 0$  //  $\tau$ : iteration number
    initialize  $P(\tau)$  //  $P(\tau)$ : population for iteration  $\tau$ 
    evaluate  $f(P(\tau))$  //  $f(P(\tau))$ : fitness function
    while (not termination condition) do
        begin
             $\tau \rightarrow \tau + 1$ 
            select 2 parents  $p_1$  and  $p_2$  from  $P(\tau-1)$ 
            perform genetic operations (crossover and mutation)
            reproduce a new  $P(\tau)$ 
            evaluate  $f(P(\tau))$ 
        end
    end
end
```

Fig. A-1. Procedure of the standard GA.

```

Procedure of the improved GA
begin
     $\tau \rightarrow 0$            //  $\tau$ : iteration number
    initialize  $P(\tau)$    //  $P(\tau)$ : population for iteration  $\tau$ 
    evaluate  $f(P(\tau))$  //  $f(P(\tau))$ : fitness function
    while (not termination condition) do
        begin
             $\tau \rightarrow \tau + 1$ 
            select 2 parents  $p_1$  and  $p_2$  from  $P(\tau-1)$ 
            perform crossover operation according to equations (A.7) - (A.10)
            perform mutation operation according to equation (A.14) to generate the offspring  $os$ 
            // reproduce a new  $P(\tau)$ 
            if random number  $< p_a$  //  $p_a$ : probability of acceptance
                 $os$  replaces the chromosome with the smallest fitness value in the
                population
            else if  $f(os) > \text{smallest fitness value in } P(\tau-1)$ 
                 $os$  replaces the chromosome with the smallest fitness value
            end
            evaluate  $f(P(\tau))$ 
        end
    end
end

```

Fig. A-2. Procedure of the improved GA.

		Improved GA				Published GA			
	No. of iters.	Ave. Fitness	Max. Fitness	Min. Fitness	Standard Deviation	Ave. Fitness	Max. Fitness	Min. Fitness	Standard Deviation
f_1	500	1.0000	1.0000	1.0000	0.0000	0.9998	1.0000	0.9984	0.0003
f_2	500	1.0000	1.0000	1.0000	0.0000	0.8467	1.0000	0.3333	0.2377
f_3	500	0.9998	1.0000	0.9984	0.0004	0.9214	0.9649	0.8662	0.0239
f_4	5000	1.0000	1.0000	1.0000	0.0000	0.6169	0.6979	0.5200	0.0450
f_5	500	1.0000	1.0000	1.0000	0.0000	1.0000	1.0000	1.0000	0.0000
f_6	1500	1.0000	1.0000	1.0000	0.0000	0.8039	1.0000	0.0122	0.3963
f_7	4000	1.0000	1.0000	1.0000	0.0000	0.7193	0.9993	0.3319	0.2818
f_8	1000	1.0000	1.0000	1.0000	0.0000	0.8274	1.0000	0.2758	0.2196

Table A-1. Average fitness values obtained from the improved GA and the standard GA for the benchmark test functions.

	Improved GA				Published GA		
	w	p_m	w_f	w_r	b	p_c	p_m
f_1	0.1	0.3	0.01	0.01	5	0.7	0.9
f_2	0.1	0.2	1	10	0.1	0.7	0.2
f_3	0.5	0.2	0.01	10	1	0.8	0.5
f_4	0.1	0.3	0.1	0.1	1	0.8	0.4
f_5	0.5	0.8	0.1	0.5	5	0.8	0.35
f_6	0.1	0.9	1	0.05	1	0.8	0.9
f_7	0.1	0.1	0.1	0.1	1	0.8	0.1
f_8	0.1	0.1	0.1	0.1	5	0.8	0.1

Table A-2. Control parameters of GAs for the benchmark test functions.

REFERENCES

- [1] A. P. Engelbrecht, *Computational Intelligence An Introduction*. England: John Wiley & Sons, 2002.
- [2] B.D. Liu, C.Y. Chen, and J.Y. Tsao, "Design of adaptive fuzzy logic controller based on linguistic-hedge concepts and genetic algorithms," *IEEE Trans. Syst., Man, Cybern. B*, vol. 31, no. 1, pp. 32-53, Feb. 2001.
- [3] B. Kosko, *Neural Networks and Fuzzy System: A Dynamical Systems Approach to Machine Intelligence*. Prentice Hall, 1991.
- [4] B.K. Dolenko and H.C. Card, "Handwritten digit feature extraction and classification using neural networks," *Canadian Conf.* vol. 1, pp. 88-91, 14-17 Sept. 1993.
- [5] C.C. Lee, "Fuzzy logic in control systems: fuzzy logic controller – part I & II," *IEEE Trans., Syst., Man, Cybern.*, vol. 20, No. 2, pp. 404-435, 1990.
- [6] C.F. Juang, J.Y. Lin, and C.T. Lin, "Genetic reinforcement learning through symbiotic evolution for fuzzy controller design," *IEEE Trans. Syst., Man, Cybern. B*, vol. 30, no. 2, pp. 290–302, April 2000.
- [7] C.M. Travieso, C.R. Morales, I.G. Alonso, and M.A. Ferrer, "Handwritten digits parameterisation for HMM based recognition," *IEE Conf. Image Processing and its Application*, no. 465, 1999
- [8] D.E. Rumelhart, GE. Hinton, and R.F. Williams, "Learning internal representations by error propagation," *Parallel Distributed Processing*, vol. 1, pp. 318-362, 1986.
- [9] D. Nauck, *Neural-Fuzzy Systems*. Chichester, England: Wiley, 1997.

- [10] D.T. Pham and D. Karaboga, *Intelligent Optimization Techniques, Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks*. Springer, 2000.
- [11] G. Castellano and A. M. Fanelli, "An iterative pruning algorithm for feedforward neural networks," *IEEE Trans. on Neural Network*, vol. 8, no. 3, May 1997.
- [12] G.P. Zhang, "Neural networks for classification: a survey," *IEE Trans. Syst., Man, Cybern. C*, vol. 30, No. 4, Nov. 2000.
- [13] H. Costin, A. Ciobanu, and A. Todirascu, "Handwritten script recognition system for languages with diacritic signs," *IEEE Int. Conf. on Computational Intelligence*, pp. 1188-1193, 4-9 May 1998.
- [14] H. Juidette and H. Youlal, "Fuzzy dynamic path planning using genetic algorithms," *Electronics Letters*, vol. 36, no. 4, pp. 374-376, Feb. 2000.
- [15] H.K. Lam, F.H.F. Leung and P.K.S. Tam, "Design and stability analysis of fuzzy model based nonlinear controller for nonlinear systems using genetic algorithm," *IEEE Trans. Syst., Man and Cybern, Part B: Cybernetics*, vol. 33, no. 2, pp. 250-257, Apr. 2003.
- [16] J.A. Goguen, "L-fuzzy sets," *Journal of Mathematics, Analysis and Applications*, vol. 18, pp. 145-174, 1967.
- [17] J. A. Markowitz, *Using Speech Recognition*. New Jersey, Prentice Hall, 1996.
- [18] J.H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: University of Michigan Press, 1975.
- [19] J.J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," in *Proc. Nat. Acad. Sci. US*, 1984, vol. 81, pp. 3088-3092.

- [20] J. Wu and C. Chan, "Isolated word recognition by neural network models with cross-correlation coefficients for speech dynamic," *IEEE Trans. Pattern Analysis and Machines Intelligence*, vol. 15, no. 11, Nov. 1993.
- [21] K. Belarbi and F. Titel, "Genetic algorithm for the design of a class of fuzzy controllers: an alternative approach," *IEEE Trans. Fuzzy Systems*, vol. 8, no. 4, pp. 398-405, Aug. 2000.
- [22] K.F. Leung, F.H.F. Leung, H.K. Lam, and P.K.S. Tam, "Recognition of speech commands using a modified neural network and an improved GA," in *Proc. 12th IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE 2003)*, St. Louis, MO, 25-28 May 2003, pp. 190-195.
- [23] L.A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338-353, 1965.
- [24] L. Rabiner and B.H. Juang, *Fundamentals of Speech Recognition*. London: Prentice Hall, 1993.
- [25] L.X. Wong, *A Course in Fuzzy System and Control*. NJ: Prentice Hall, 1997.
- [26] M. Brown and C. Harris, *Neuralfuzzy Adaptive Modeling and Control*. Prentice Hall, 1994.
- [27] 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.
- [28] M.R. Kaimal, *Neural-Fuzzy Control System*. New Delhi: Narosa Publishing House, 1997.
- [29] M. Setnes and H. Roubos, "GA-fuzzy modeling and classification: complexity and performance," *IEEE Trans. Fuzzy Systems*, vol. 8, no. 5, pp. 509-522, 2000.
- [30] O. Matanetal, "Reading handwritten digits: A ZIP code recognition system," *IEEE Comput. Mag.*, vol. 25, no.7, pp. 59-63, July 1992.

- [31] R. Caponetto, L. Fortuna, G. Nunnari, L. Occhipinti, and M. G. Xibilia, "Soft computing for greenhouse climate control," *IEEE Trans. Fuzzy Systems*, vol. 8, no. 6, pp. 753-760, Dec. 2000.
- [32] R.J.S. Jang, *Neural-Fuzzy and Soft Computing*. NJ: Prentice Hall, 1997.
- [33] R. Reed, "Pruning algorithm – A survey," *IEEE Trans. on Neural Network*, vol. 4, no. 5, Sept. 1993.
- [34] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd Ed. Prentice Hall, 1999.
- [35] S.W. Lee and S.Y. Kim, "Integrated segmentation and recognition of handwritten numerals with cascade neural network," *IEEE Trans. on Sys. Man, and Syb.*, Part C, vol. 29, no. 2, Feb. 1999.
- [36] T. Lee, P.C. Ching, and L.W. Chan, "Isolated word recognition using modular recurrent neural networks," *Pattern Recognition*, vol. 31, no. 6, pp. 751-760, 1998.
- [37] 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.
- [38] T.J. Cholewo and J.M. Zurada, "Sequential network construction for time series prediction," in *Proc. Int. Conf. Neural Networks*, 1997, vol. 4, pp. 2034-2038.
- [39] T. Kohonen, *Self-Organization and Associative Memory*. Berlin: Springer-Verlag, pp. 132-1988.
- [40] 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.

- [41] Y.S. Zhou and L.Y. Lai, "Optimal design for fuzzy controllers by genetic algorithms," *IEEE Trans. Industry Applications*, vol. 36, no. 1, pp. 93-97, Jan-Feb. 2000.
- [42] Z. Michalewicz, *Genetic Algorithm + Data Structures = Evolution Programs*, 2nd extend ed. Springer-Verlag, 1994.
- [43] A Chinese-Cantonese Syllabary,
<[http://humanum.arts.cuhk.edu.hk/Lexis/Cant Canton/](http://humanum.arts.cuhk.edu.hk/Lexis/Cant%20Canton/)>
- [44] Acrobat Reader, <<http://www.adobe.com/acrobat/>>
- [45] Microsoft Reader, <<http://www.microsoft.com/reader/>>