Fuzzy Vector Quantization on the Modeling of Discrete Hidden Markov Model for Speech Recognition

Shing-Tai Pan

Abstract

This paper applies fuzzy vector quantization (FVQ) to the modeling of Discrete Hidden Markov Model (DHMM) and then to improve the speech recognition rate for the Mandarin speech. Vector quantization based on a codebook is a fundamental process to recognize the speech signal by DHMM. A codebook will be first trained by K-means algorithms using Mandarin training speech. Then, based on the trained codebook, the speech features are quantized by the fuzzy sets defined on each vectors of the codebook. Subsequently, the quantized speech features are statistically applied to train the model of DHMM for the speech recognition. All the speech features to be recognized should go through the FVQ based on the fuzzy codebook before being fed into the DHMM model for recognition. Experimental results in this paper shows that the speech recognition rate can be improved by using FVQ algorithm to train the model of DHMM.

Keywords: Fuzzy Vector Quantization, Speech Recognition, Discrete Hidden Markov Model.

1. Introduction

The dependence of human life on electronic products is higher and higher due to the development in IT technology. To make the electronic products popular, attractive functions and good services are very important requirement on the products. The interface between these products and user is quite important too. For example, handwritten input and touch screen monitor are favored by users. Recently, the topic on the process of audio signal attracts more attention [1-3]. There are many researches about speech recognition [4-6] because speech recognition will be a standard interface in the future.

As for the history of the speech recognition, the first recognition platform is Dynamic Time Warping (DTW) [4] which used dynamic programming [7] to calculate the difference between the target speech and testing speech to recognize the testing speech. Then, Artificial Neural Network (ANN) was proposed to replace DTW for speech recognition. Because that the structure of ANN will be fixed after it is determined, the recognition rate can't be improved by online learning with more additive speech signals. Recentlt, Hidden Markov Model (HMM) [8] was widely applied on speech recognition [9, 10]. It can solve the problem arises from variant speech speed and be constructed layer by layer to achieve automatic speech recognition (ASR). Before speech recognition, speech signal have to be pre-processed. The pre-process of speech signal includes speech sampling, point detection, pre-emphasis, Hamming window and features capture. After these processes, we can evaluate the probabilities of every HMM model corresponding to each speech and find the model which has highest probability to be the result of recognition. Consequently, in this paper, the HMM is adopted to be the speech recognition algorithm. Moreover, in order to reduce the number of data for computing, the DHMM is used here. Moreover, the feature of speech signal which was used in this paper is obtained by Mel-Frequency Cepstrum Coefficient (MFCC) [8]. However, the process of Mel-frequency Cepstrum Coefficient includes many floating-point operations which will cost much computation time and power of embedded system. Indeed, according to the experimental results in this paper, the process of MFCC will cost most computation time during the speech recognition process. This is due to the fact that the float FFT would be performed in MFCC process and hence waste much time. Consequently, in order to improve the recognition speed of speech, it is the most important work to reduce the computation for FFT. For this purpose, this study will use Integer FFT [11] to replace Float FFT [12] in the process of MFCC.

Furthermore, the codebook for the speech feature quantization plays an important role on the training of DHMM model. A well-trained codebook will enhance the total performance of the speech recognition systems. In the past, the speech features are quantized by a codebook trained by K-means algorithm. This winner-take-all algorithm can not perform well on vector quantization, since multiple level of an element in some vectors exists in many applications. Consequently, the fuzzy vector

Corresponding Author: Shing-Tai Pan is with the Department of Computer Science and Information Engineering, National University of Kaohsiung, No. 700, Kaohsiung University Rd., Nanzih Dist., Kaohsiung 811, Taiwan, R.O.C. (Phone: 886-7-5919476; Fax: 886-7-5919514)
E-mail: stpan@nuk.edu.tw

quantization is used improve the performance of the vector quantization. Many research shows that the fuzzy vector quantization outperform VQ [13]. Besides, the relative works of fuzzy vector quantization, e.g. the fuzzy clustering [14, 15] and fuzzy data retrieval systems [16] also demonstrate the benefit of fuzzy quantization method.

This paper is organized as follows. The speech pre-processes used in this paper are first introduced in Section II. In section III, the platform for the speech recognition and FVQ is investigated. The implementation strategy of the integer FFT is presented in Section IV. The experiment of the speech recognition system is then presented in Section V where the speech recognition rates by using VQ and FVQ with or without noise are compared.

2. Speech Pre-Processing

A. Speech Sampling

The continuous speech signal which was recorded by a microphone must be transformed into discrete data because computer only can process discrete data. All the values which were recorded at any specific time can describe the wave of speech. Unsuitable sampling frequency is an important reason for the loss of speech. Higher sampling frequency will loss less data but has to deal with more data while lower sampling frequency will loss more data but has less data to be processed. According to sampling theorem [17]: sampling frequency can not smaller than 2 times of the signal bandwidth, we adopt 8kHz to be sampling frequency because the bandwidth of speech signal is smaller than 4kHz.

B. Point Detection

The recorded speech sound signals will include speech segments, silence segments and background noise. The process to separate speech segments and silence segments is called End-Point Detection (EPD), see please Fig. 1 for example. If the unnecessary parts, i.e. the silence segment, are removed, the number of frames for recognition will decrease, and then the recognition speed will be enhanced.

There are many algorithms for speech sound signal EPD, which can be roughly divided into three types according to the different domain for representing the signal: (1) time-domain EPD; (2) frequency-domain EPD (3) mixed-parameter EPD. Among them, time-domain EPD is one of the simplest and the most popular ways. But it has the disadvantage of weak anti-noise capacity. As for frequency-domain EPD and mixed-parameter EPD, both have stronger anti-noise capacity and hence are more precise in recognition. But the disadvantage is that more complex calculation is needed for frequency-domain analysis.

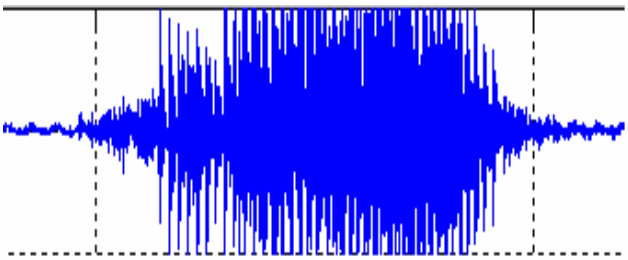

Fig. 1. End-point detection.

C. Pre-emphasis

A speech signal will attenuate in magnitude when it spreads via air. The signal with higher frequencies will attenuate more seriously. In order to compensate the attenuated magnitude of high-frequency speech signals, the speech signal will be fed into high-pass. The high-pass filter used in this paper is governed by the equation as follows.

$$S(n) = X(n) - 0.95X(n-1), \quad 1 \le n \le L;$$

In the equation (1), $S(n)$ represents the signal that has been processed with pre-emphasis, while $X(n)$ represents the original signal, and L is the length (number of sampling) of each audio frame.

D. Hamming Window

The purpose to apply Hamming window to each frame of speech signals is to avoid the discontinuity exists between every two frames and in both ends of every frames. By multiplying by Hamming window, the influence of non-continuity will decrease (to make each audio frame more centered on the frequency spectrum). Hamming window can be expressed by the following equation:

$$W(n) = \begin{cases} 0.54 - 0.46\cos(\frac{2n\pi}{N-1}), 0 \le n \le N-1; \\ 0, otherwise \end{cases}$$

$$F(n) = W(n) \times S(n);$$

in which N is the length of audio frame; $S(n)$ is a frame of speech signal; $W(n)$ is the Hamming window and $F(n)$ is the result of speech signal multiplied by Hamming window.

E. Feature Capture

In speech recognition, the methods commonly used for extracting the feature of speech signals can be divided into two main categories: one is time-domain analysis, and the other is frequency-domain analysis. The way of the time-domain analysis is more direct and time-saving, with fewer operations. On the other hand, the frequency-domain analysis has to take Fourier transform on the signal, so it needs more operations and is more complicated and hence leads to the requirement of

much more computation time compared to time-domain analysis. The most popular methods for features extraction are Linear Predict Coding (in time domain) and Cepstrum Coefficient and MFCC (in frequency domain) [8]. Because MFCC is more close to the distinction made by human ears toward speech sound, we use it to extract the feature for speech sound in this paper. The processes of MFCC are described as follows. First, each audio frame is transformed to frequency domain, says $|X(k)|$. Due to masking effect in sound, we make the energy in each frequency domain $|X(k)|$ be multiplied by a triangle filter as follows.

$$B_m(k)=\begin{cases}0, k<f_{m-1}\\ \dfrac{k-f_{m-1}}{f_m-f_{m-1}}, f_{m-1}\le k\le f_m\\ \dfrac{f_{m+1}-k}{f_{m+1}-f_m}, f_m\le k\le f_{m+1}\\ 0, f_{m+1}<k\end{cases} \tag{1}$$

where $1\le m\le M$ and M is the total number of the filters. After accumulating and applying the *log*(.) function, we can get a energy function

$$Y(m)=\log\left\{\sum_{k=f_{m-1}}^{f_{m+1}}|X(k)|B_m(k)\right\}. \tag{2}$$

Applying the Discrete Cosine Transform on $Y(m)$, we then obtain

$$c_x(n)=\frac{1}{M}\sum_{m=1}^{M}Y(m)\cos(\frac{\pi n(m-\frac{1}{2})}{M}) \tag{3}$$

in which $c_x(n)$ is the obtained MFCC.

3. Speech Recognition Platform

After speech pre-processing, features of speech are available. Then these features will be fed into recognition platform for recognition. The recognition platform in the paper is Discrete Hidden Markov Model (DHMM).

A. Fuzzy Vector Quantization

In order to train the DHMM model, a codebook should be set up first. In the codebook, feature vector must be classed as by vector quantization [8]. The *K*-means algorithm is adopted to train the codebook at head. Then, combing the fuzzy set, the codebook is arranged as a fuzzy codebook and all the speech features includes speech in training phase and in testing phase will be fuzzy vector quantized through the fuzzy codebook. Suppose a set of real numbers are expressed as $\{v_1, v_2, \cdots, v_n\}$ in which $v_i\in R$ and the corresponding fuzzy set is expressed as $\{u_i(x)\,|\,i=1,2,\cdots n\}$, where $\sum_1^n u_i(x)=1, \forall x\in R$. The triangular fuzzy set is depicted in Fig. 2. It is noted that since the elements of the codebook are not uniformly distributed in the domain of the speech signal after the training by *K*-means algorithm, the triangular fuzzy set is not symmetric in this application. The membership function in Fig. 2 is described as [13].

$$u_i(x)=\begin{cases}\dfrac{x-v_{i-1}}{v_i-v_{i-1}}, & \forall x\in[v_{i-1}, v_i],\\ \dfrac{x-v_{i+1}}{v_i-v_{i+1}}, & \forall x\in[v_i, v_{i+1}].\end{cases}$$

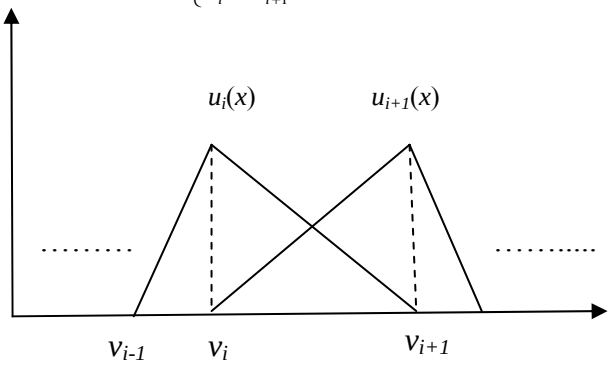

Fig. 2. Fuzzy set corresponding to each element of codebook.

B. Discrete Hidden Markov Model (DHMM)

The DHMM is a double layers random process. The transfer of hidden states will corresponds to the transfer of observations. Each model of DHMM can describe a specific speech. The features of a speech are the observations used to estimate the hidden states. The target speech can be recognized by calculating the probability of the DHMM model. The model with highest probability in all DHMM models represents the most possibility of the recognized speech corresponding to the model [8].

In this section, the DHMM model is denoted as

$$\lambda=\{A,B,\pi,S,V\} \tag{4}$$

in which

$S=\{s_1,s_2,\cdots,s_N\}$ is the hidden state;

$V=\{v_1,v_2,\cdots,v_M\}$ is the output set of the DHMM model;

$A=\{a_{ij}\}, a_{ij}=P(q_t=s_j\,|\,q_{t-1}=s_i)$ is the probability matrix of state transfer;

$B=\{b_j(k)\}, b_j(k)=P(o_t=v_k\,|\,q_t=s_j)$ is the probability matrix of transfer out from each states;

$\pi=\{\pi_i\}, \pi_i=P(q_1=s_i), 1\le i\le N$ is the initial state probability;

$O=\{o_1,o_2,\cdots,o_T\}$ is the observation sequence;

$Q=\{q_1,q_2,\cdots,q_T\}$ is the state sequence.

As for the training of the DHMM model, the matrices *A*, *B*, and π are randomly generated initially. Then, the matrices are updated by the following algorithm.

$$state_t = \arg \max_{0 \le i \le 6} \alpha_t(i) \tag{5}$$

in which $state_t$ is the best estimated hidden state at time *t*; $\alpha_t(i)$ is the probability for *i*th guessed state at time *t* according to *A*, *B*, and π. Moreover, the matrices *A* and *B* are trained as follows.

The trained parameter $\overline{a}_{ij}$ of a_{ij} and $\overline{b}_j(k)$ of $b_j(k)$ is described as

$$\overline{a}_{ij} = \frac{n(u_{ij})}{n(u_{i\bullet})} \tag{6}$$

$$\overline{b}_j(k) = \frac{n(u_{\bullet j}, o = v_k)}{n(u_{\bullet j})} \tag{7}$$

in which

u_{ij} is the event that the state s_i transfer to s_j;

$u_{i\bullet}$ is the event that the state s_i transfer to other states;

$u_{\bullet j}$ is the event that enter the state s_j;

$n(u_{ij})$ is the number of times that s_i transfer to s_j;

$n(u_{i\bullet})$ is the number of times s_i transfer to other states;

$n(u_{\bullet j})$ is the number of times that enter the state s_j;

$n(u_{\bullet j}, o = v_k)$ is the number of times that the observation v_k occur when enter the state s_j.

The number $n(u_{\bullet j}, o = v_k)$ is obtained through the fuzzy codebook as follows. Suppose that the speech features are classified into *n* classes by *K*-means algorithm, i.e., the trained codebook is expressed as $CB_{n \times m} = \{FK_1, FK_n, \cdots, FK_n\}^T$ where $FK_i \in R^m$ is the center vector of *i*th class. The fuzzy degree contribute to the number $n(u_{\bullet j}, o = v_k)$ for each speech feature $f \in R^m$ is computed as $u_i(v_i + \|FK_i - f\|)$ for $n(u_{\bullet j}, o = v_i)$ and $u_i(v_{i+1} - \|FK_{i+1} - f\|)$ for $n(u_{\bullet j}, o = v_{i+1})$. This tactic is not only used for the training phase of the DHMM model but also for the testing phase of the speech recognition. In this paper, the strategy of using DHMM to recognize the speech signal is that we train first DHMM model λ for each corresponding speech by fuzzy codebook. Then, in testing phase, the tested speech features which is viewed as a sequence of observation *O* are fed into each DHMM model. The highest probabilities are found from computing the probability for all models by using the following equation:

$$P(O \mid \lambda) = \sum_{allQ} P(O, Q \mid \lambda)$$

$$= \sum_{q1,q2,\ldots,qT} \pi_{q1} \cdot b_{q1}(o_1) \cdot a_{q1q2} \cdot b_{q2}(o_2) \cdot a_{q2q3} \cdots a_{qT-1qT} \cdot b_{qT}(o_T) \tag{8}$$

Figure 3 reveals the processes for the training of the fuzzy codebook and DHMM model. Besides, the speech recognition process also can be found in the figure.

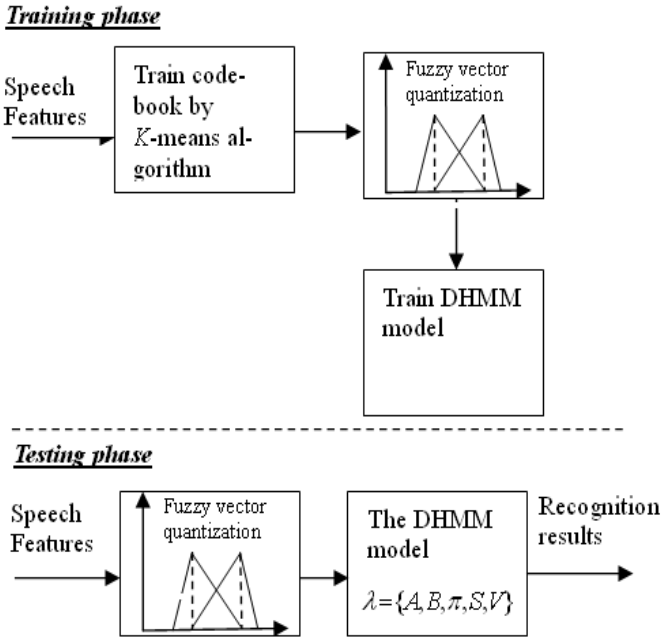

Fig. 3. Strategy of the proposed FVQ on speech recognition by DHMM.

4. Fast Fourier Transform

As for the implementation of speech recognition systems on the embedded platform, if Discrete Fourier Transform (DFT) [12] is used to transform the time-domain signal to the frequency-domain signal in the calculation of MFCC, the burdens on computation time will be too huge to have a real-time application. So, we use FFT to increase the speed. However, due to the limitation of FFT, the sampling points of each audio frame should be limited in 2^n times.

A. FFT Algorithm

To transform the discrete signal from time domain to frequency domain by DFT is described as follows [12]:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, 0 \le k \le N-1; \tag{9}$$

where $W_N = e^{\frac{-j2\pi}{N}}$; *N* is the number of sampling points in an audio frame.

For the calculation of DFT, we can reach a high efficiency by decomposing the DFT to many serial small

DFT and then calculating it for each small DFT. During this process, both the symmetric and periodic properties of the complex number index $W_N^{kn} = e^{-j\left(\frac{2\pi}{N}\right)kn}$ are used. The decomposition of the algorithm is based on decomposing the sequence $x[n]$ into many small sequences. Hence, it is called time-division algorithm. First, the DFT in (9) is decomposed as

$$X[k] = \sum_{n=0}^{\frac{N}{2}-1} f[n]W_{N/2}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} g[n]W_{N/2}^{nk} = F[k] + W_N^k G[k] \quad (10)$$

in which $f[n] = x[2n]$ and $g[n] = x[2n+1]$ are the even sampling and odd sampling of $x[n]$. Figure 4 shows the time division for FFT. The required multiplication complexity N^2 for original DFT can then be reduced to be $\frac{N}{2}\log_2 N$.

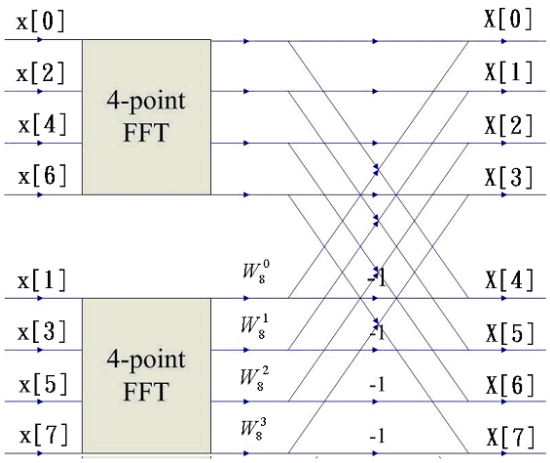

Fig. 4. Time division for 8-point FFT.

B. Integer FFT

On the embedded platform, the floating-point arithmetic operation will cost much time. Even if FFT is used, the calculation time for Mel-Frequency Cepstrum Coefficient does not conform to the real-time application. Therefore, for the multiplication and addition for floating-point numbers, integer numbers are used for substitute; however, the recognition rate decreases accordingly. In Figure 4,

$$W_N^k = e^{-j\left(\frac{2\pi}{N}\right)k} = \cos\left(-\frac{2\pi k}{N}\right) + j\sin\left(-\frac{2\pi k}{N}\right). \quad (11)$$

In Integer FFT, we shift the value of the functions *cos*(.) and *sin*(.) to the left for *n* bit and move them back to the right for n bit when the multiplication is completed. For the realization of Integer FFT, the real and imaginary part of W_N^k will be amplified by a factor of *SF* and truncated to a integer before the FFT operation. These results are then tabulated as a look-up table to accelerate the computation time. In the last stage of the operation of $W_N^k G[k]$, the factor 1/SF will be multiplied for recovering the original magnitude. We can then obtain an approximation of $W_N^k G[k]$. The detail operations are illustrated in Fig. 5.

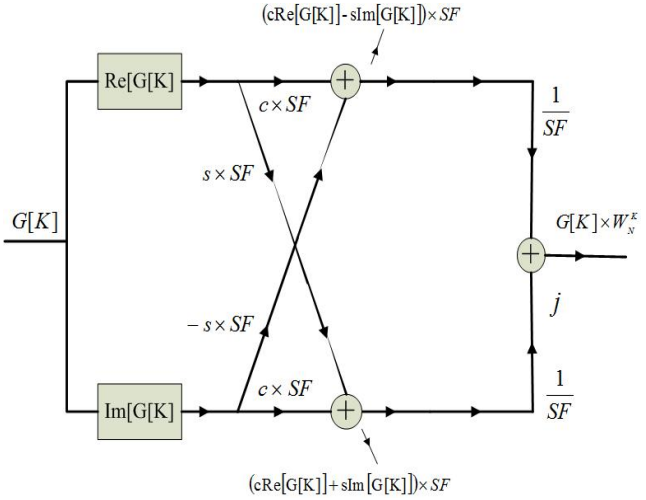

Fig. 5. Butterfly chart of complex number multiplication.

During the process of FFT, an addition may have a bit increase in the length of the temporary results of the multiplication and hence the final results will raise to N_c-1 bits, in which N_c is the bit number of the multiplier. For the variables of real numbers and imaginary numbers stored inside memory, sufficient storage space is indispensable to keep from overflow.

5. Experiment of DHMM Speech Recognition Systems

In speech pre-processing stage, the recording format is 8kHz, single channel, and 16bits length. The length of frame is 256 sampling points. The overlapping rate of frame is 50%. We adopted for time domain end point detection and calculated the threshold by the following formula:

$$Threshold = 7.5\% \times \max\left[E(n)\right] + \frac{1}{K}\sum_{i=1}^{k} E(i); 1 \le n \le N \quad (12)$$

in which $E(i)$ represents the energy of ith frame and N is the number of frames.

In the Integer FFT stage, shift right and multiplication will make overflow. This problem will decrease the recognition rate in the experiment. More storage space will be retained to solve the overflow problem. If the length of variable is too long, the computing time will be huge. Contrarily, if the length of variable is too short, the recognition rate will drop. To make a trade-off, we use 32 bits length memory for storing the variables.

As for the implementation of DHMM on speech recognition platform, the number of hidden states can be arbitrarily set and the number of observation is set to be the number of the cluster in the codebook which is used to quantize the speech signal to be identified. In this application, we use 7 hidden states, 64 observations (the number of cluster in codebook) for DHMM. Consequently, the dimension of the codebook in this experiment is 64×10. Moreover, the dimension of the matrices A, B, and π, in the DHMM model described by (4) are 7×7, 7×64, and 7×1, respectively. Every model starts at state 0 and all the states only can jump to next or next 2 states, see Fig. 6 for detail. Fig. 7 presents the frames corresponds to the state-observation diagram for DHMM.

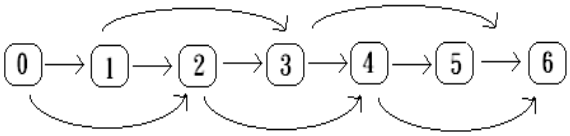

Fig. 6. DHMM states transfer structure.

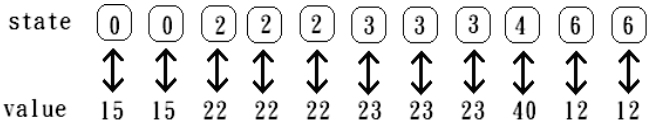

Fig. 7. DHMM corresponding state-observation diagram.

A codebook is trained by *K*-means algorithm with training speech signals. The DHMM then be modeled for each speech based on the trained codebook. The speeches which will be recognized are 0-9. And hence there are 10 DHMM models after the training phase. Each number was recorded and recognized by DHMM 100 times.

The strategy in Fig. 3 is then used to implement the speech recognition systems. The testing speech signal includes clean signal, 20db, 15db, 10db noised signal. The experiment is performed by two different ways: one is that only *K*-means algorithm is used for training the codebook and DHMM model and another one is that the fuzzy vector quantization is used to train the codebook and DHMM model. The results for clean speech signal are listed in Table 1 and Table 2, in which four sets of speech signal are recognized for comparison. Moreover, results for 20db, 15db and 10db noised speech signal are listed in Table 3, Table 4 and Table 5, respectively, in which two sets of speech signal are recognized for comparison. It is obvious that the speech recognition rate by using FVQ is better than that without FVQ no matter what the noised signals are presented.

6. Conclusions

The fuzzy vector quantization (FVQ) had been used on the modeling of Discrete Hidden Markov Model (DHMM) to improves the speech recognition rate for the Mandarin. First, a codebook was trained by *K*-means algorithms. Based on the trained codebook, the speech features are quantized by the fuzzy sets defined on the trained codebook. Subsequently, the quantized speech features are used to the modeling of DHMM for the speech recognition. All the speech features should go through the FVQ based on the fuzzy code book before being fed into the DHMM model for recognition. Experimental results reveal that the speech recognition rate can be improved by using FVQ algorithm to train the model of DHMM.

Acknowledgment

This research work was supported by the National Science Council of the Republic of China under contract NSC 99-2221-E-390-027. The authors would like to thank Xu-Yu Li for his help in implementing the proposed algorithm.

References

[1] X. Huang, Y. Abe, and I. Echizen, "Capacity Adaptive Synchronized Acoustic Steganography Scheme," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 2, pp. 72-90, Apr. 2010.

[2] J. McAuley, J. Ming, D. Stewart, and P. Hanna, "Subband Correlation and Robust Speech Recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 13, pp.956-964, 2005.

[3] K. Yamamoto and M. Iwakiri, "Real-Time Audio Watermarking Based on Characteristics of PCM in Digital Instrument," *Journal of Information Hiding and Multimedia Signal Processing,* vol. 1, no. 2, pp. 59-71, Apr. 2010.

[4] C. Wan and L. Liu, "Research and Improvement on Embedded System Application of DTW-based Speech Recognition," *International Conference on Anti-counterfeiting, Security and Identification*, pp. 401-404, 2008.

[5] T. Kinjo and K. Funaki, "On HMM Speech Recognition Based on Complex Speech Analysis," *Conference on IEEE Industrial Electronics*, pp. 3477-3480, 2006.

[6] H. Sayoud and S. Ouamour, "Proposal of a New Confidence Parameter Estimating the Number of Speakers -An experimental investigation," *Journal*

of Information Hiding and Multimedia Signal Processing, vol. 1, no. 2, pp. 101-109, Apr. 2010.
[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms 2nd Editon.* McGraw-Hill, 2002.
[8] X. Huang, A. Acero, and H. Wuenon, *Spoken Language Processing A Guide to Theory, Algorithm and System Developmen.* Pearson, 2005.
[9] J. Tao, L. Xin, and P. Yin, "Realistic visual speech synthesis based on hybrid concatenation method," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 17, no. 3, pp. 469-477, 2009.
[10] S. Kwong and C. W. Chau, "Analysis of parallel genetic algorithms on HMM based speech recognition system," *IEEE Trans. on Consumer Electronics*, vol. 43, no. 4, pp. 1229-1233, 1997.
[11] S. Oraintara, Y. J. Chen, and T. Q. Nguyen, "Integer Fast Fourier Transform," *IEEE Transation on SIGNAL PROCESSING*, vol. 50, pp. 607-618, 2002.
[12] A. V. Oppenheim, R. W.Schafer, and J. R Buck, *Discrete-Time Signal Processing 2nd Edition.* Pearson, 2005.
[13] W. Pedrycz and K. Hirota, "Fuzzy Vector Quantization with The Particle Swarm Optimization: A Study in Fuzzy Granulation-Degranulation Information Processing," *Signal Processing*, vol. 87, pp. 2061-2074, 2007.
[14] V. Kapoor, S. S. Tak, and V. Sharma, "Location Selection – A Fuzzy Clustering Approach," *International Journal of Fuzzy Systems*, vol. 10, no. 2, pp. 123-128, June 2008.
[15] C. H. Li, W. C. Huang, B. C. Kuo, and C. C. Hung, "A Novel Fuzzy Weighted C-Means Method for Image Classification," *International Journal of Fuzzy Systems*, vol. 10, no. 3, pp. 168-173, June 2008.
[16] A. Lakdashti, M. S. Moin, and K. Badie, "Reducing the Semantic Gap of the MRI Image Retrieval Systems Using a Fuzzy Rule Based Technique," *International Journal of Fuzzy Systems,* vol. 11, no. 4, pp. 232-249, Dec. 2009.
[17] S. Haykin and B. V. Veen, *Signals and Systems 2nd Edition.* Wiley, 2003.

Shing-Tai Pan was born in Pingtung, Taiwan, on November 4, 1966. He received the M.S. degree in electrical engineering from National Sun Yat-Sen University, Kaohsiung, Taiwan, in 1992 and the Ph. D. degree from National Chiao Tung University, Hsinchu, Taiwan, in 1996. In 2006, he joined the Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung, Taiwan, as an Associate Professor.

He is a member of Taiwanese Association for Artificial Intelligence (TAAI) and Chinese Automatic Control Society (CACS). He is also a member of The Association for Computational Linguistics and Chinese Language Processing (ACLCLP). His current research interests are in the area of digital signal process, biomedical signal processing, speech recognition, evolutionary computations, applications of artificial intelligent and intelligent control systems design.

Table 1. The recognition rate for clean speech without FVQ.

clean	Recognition for Speech data #1			Recognition for Speech data #2			Recognition for Speech data #3			Recognition for Speech data #4		
Speech	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate
0	47	3	0.94	50	0	1	50	0	1	46	4	0.92
1	49	1	0.98	50	0	1	50	0	1	49	1	0.98
2	50	0	1	50	0	1	38	12	0.76	50	0	1
3	43	7	0.86	45	5	0.9	34	16	0.68	43	7	0.86
4	19	31	0.38	35	15	0.7	50	0	1	24	26	0.48
5	50	0	1	50	0	1	50	0	1	50	0	1
6	48	2	0.96	38	12	0.76	46	4	0.92	48	2	0.96
7	50	0	1	50	0	1	50	0	1	50	0	1
8	36	14	0.72	50	0	1	49	1	0.98	30	20	0.6
9	46	4	0.92	27	13	0.54	39	11	0.78	44	6	0.88
Avg.			**0.876**			**0.89**			**0.912**			**0.868**

Table 2. The recognition rate for clean speech with FVQ.

clean	Recognition for Speech data #1			Recognition for Speech data #2			Recognition for Speech data #3			Recognition for Speech data #4		
Speech	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate
0	44	6	0.88	50	0	1	50	0	1	44	6	0.88
1	49	1	0.98	50	0	1	50	0	1	50	0	1
2	50	0	1	50	0	1	50	0	1	50	0	1
3	40	10	0.8	41	9	0.82	41	9	0.82	45	5	0.9
4	28	22	0.56	33	17	0.66	36	14	0.72	23	27	0.46
5	50	0	1	48	2	0.96	50	0	1	50	0	1
6	48	2	0.96	44	6	0.88	47	3	0.94	47	3	0.94
7	50	0	1	50	0	1	50	0	1	50	0	1
8	48	2	0.96	50	0	1	50	0	1	50	0	1
9	38	12	0.76	48	2	0.96	50	0	0.68	33	17	0.66
Avg.			**0.89**			**0.928**			**0.916**			**0.884**

Table 3. The comparison of recognition rate for 20db noised speech with and without FVQ.

20db	Recognition for Speech data #1 (without FVQ)			Recognition for Speech data #2 (without FVQ)			Recognition for Speech data #1 (FVQ)			Recognition for Speech data #2 (FVQ)		
Speech	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate
0	42	8	0.84	50	0	1	47	3	0.94	50	0	1
1	44	6	0.88	38	12	0.76	50	0	1	50	0	1
2	38	12	0.76	50	0	1	50	0	1	50	0	1
3	39	11	0.78	4	46	0.08	34	16	0.68	7	43	0.14
4	34	16	0.68	28	22	0.56	36	14	0.72	27	23	0.54
5	21	29	0.42	34	16	0.68	46	4	0.92	8	42	0.16
6	41	9	0.82	8	42	0.16	50	0	1	13	37	0.26
7	50	0	1	50	0	1	50	0	1	50	0	1
8	47	3	0.94	48	2	0.96	37	13	0.74	50	0	1
9	48	2	0.96	24	26	0.48	20	30	0.4	43	7	0.86
Avg.			**0.808**			**0.668**			**0.84**			**0.696**

Table 4. The comparison of recognition rate for 15db noised speech with and without FVQ.

15db	Recognition for Speech data #1 (without FVQ)			Recognition for Speech data #2 (without FVQ)			Recognition for Speech data #1 (FVQ)			Recognition for Speech data #2 (FVQ)		
Speech	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate
0	30	20	0.6	24	26	0.48	37	13	0.74	46	4	0.92
1	43	7	0.86	37	13	0.74	49	1	0.98	26	24	0.52
2	27	23	0.54	28	22	0.56	50	0	1	44	6	0.88
3	36	14	0.72	0	50	0	36	14	0.72	5	45	0.1
4	32	18	0.64	16	34	0.32	33	17	0.66	28	22	0.56
5	1	49	0.02	14	36	0.28	1	49	0.02	0	50	0
6	39	11	0.78	37	13	0.74	45	5	0.9	14	36	0.28
7	50	0	1	50	0	1	50	0	1	50	0	1
8	31	19	0.62	22	28	0.44	17	33	0.34	43	7	0.86
9	44	6	0.88	13	37	0.26	37	13	0.74	3	47	0.06
Avg.			**0.666**			**0.482**			**0.71**			**0.518**

Table 5. The comparison of recognition rate for 10db noised speech with and without FVQ.

10db	Recognition for Speech data #1 (without FVQ)			Recognition for Speech data #2 (without FVQ)			Recognition for Speech data #1 (FVQ)			Recognition for Speech data #2 (FVQ)		
Speech	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate	Correct (times)	Wrong (times)	Recogn. rate
0	10	40	0.2	5	45	0.1	35	15	0.7	45	5	0.9
1	48	2	0.96	50	0	1	42	8	0.84	50	0	1
2	45	5	0.9	46	4	0.92	50	0	1	39	11	0.78
3	31	19	0.62	5	45	0.1	3	47	0.06	3	47	0.06
4	12	38	0.24	14	36	0.28	24	26	0.48	24	26	0.48
5	0	50	0	0	50	0	0	50	0	0	50	0
6	11	39	0.22	38	12	0.76	6	44	0.12	0	50	0
7	50	0	1	50	0	1	50	0	1	50	0	1
8	3	47	0.06	10	40	0.2	24	26	0.48	14	36	0.28
9	3	47	0.06	0	50	0	7	43	0.14	0	50	0
Avg.			**0.426**			**0.436**			**0.482**			**0.45**