

Speaker Verification System based on the Cerebellum Architecture

Abdul Wahab, Mathias Dharmawirya and Chai Quek
Center for Computational Intelligent, Nanyang Technological University,
Blk N4 #2A-36, Nanyang Avenue, Singapore 639798

Abstract - *The Cerebellar Model Articulation Controller (CMAC) and fuzzy systems have been active areas of research since its initial introduction. Fuzzy CMAC is basically a CMAC that is coupled with a fuzzy system. This paper presents the incorporation of fuzzy systems into CMAC with Approximate Analogical Reasoning Scheme (AARS) as its inference rules and Discrete Incremental Clustering (DIC) as its technique to classify the input space, which is used for the speaker verification system. Features extracted from a known speaker using either Mel Frequency Cepstrum Coefficients (MFCC) or Delta Mel Frequency Cepstrum Coefficients (DMFCC) methods are used as the inputs for the system. The variables used for the performance analysis are False Acceptance Rate (FAR), False Rejection Rate (FRR), and Equal Error Rate (EER), training and testing time.*

Keywords: CMAC, fuzzy, FCMAC, MFCC, EER,

1. Introduction

Two major fields of research in speech technology are speech recognition and speaker recognition. Speaker recognition is the process of automatically recognizing who is speaking by distinguishing qualities of speaker's voice. For this purpose it is important to preserve the speaker specific information in the speech signal. This is in contrast to the speech recognition task, where the linguistic content of the speaker's voice signal is extracted. Speech recognition then corresponds to a classification problem.

The speaker recognition can be divided into speaker identification and speaker verification. Speaker identification is one of the major researches in speech technology. It is not the same as speaker verification. Speaker identification determines which registered speaker provides a given utterance from amongst a set of known speakers. Speaker verification accepts or rejects the identity claim of a speaker.

Speaker verification methods can be divided into text-independent and text-dependent methods. In a text-

independent system, speaker models capture characteristics of somebody's speech irrespective of what he/she is saying. In a text-dependent system, on the other hand, the verification of the speaker's is based on his or her speaking one or more specific phrases.

Speaker verification systems consist of two phases. The first one is training phase, where each registered speaker has to provide samples of their speech so that the system can build or train a reference model for that speaker. The second one is testing phase, where the input speech is matched with stored reference model(s) and recognition decision is made.

The basic structure for most speaker verification [2] systems is shown in Figure 1. As can be seen, a general speaker verification approach consists of front-end processing (feature extraction from speech signal), speaker modeling and pattern matching, and making an accept/reject decision.

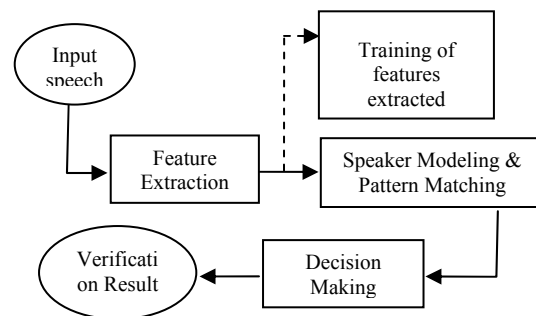


Figure 1 Basic Structure of Speaker Verification Systems

Features extracted in the front-end processing are compared to a model representing the claimed speaker obtained from previous enrollment, and to some model(s) representing the imposter speaker by pattern matching. The pattern matching will produce match score that measures the similarity of the computed input feature vectors to feature vectors pattern of the claimed speaker.

2. Feature Extraction

Feature extraction is a process to obtain some type of parametric representation from speech waveform. Speech waveform, which is high-dimensional signal, is to be converted into relatively low-dimensional features subspace while preserving the speaker's distinctive information/characteristics.

The dimensionality of extracted features also needs to be considered. It may be considered that the more relevant features, the better the recognition results. But, there is a phenomenon called '*the curse of dimensionality*' that shows that the needed data for training and testing grow exponentially with the dimensionality of the input space. Otherwise the representation will be very poor [1].

The attributes of features that are desirable for speaker verification systems are [4]:

- Easy to extract, easy to measure, occur frequently and naturally in speech
- Not affected by speaker physical state
- Not change over time, and utterance variations (fast talking vs. slow talking rates)
- Not affected by ambient noise
- Not subject to mimicry

Nevertheless, no feature has all this attributes. But, from researches in speaker verification systems, spectrum-based features have been found to be the most effective. In this paper, the Mel Frequency Cepstrum Coefficients are selected as the features. The processes to obtain the MFCC can be summarized as in Figure 2.

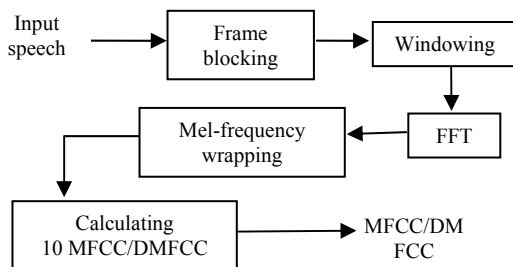


Figure 2 Method of obtaining MFCC

3. FCMAC AARS

The Cerebellar Model Articulation Controller (CMAC) is a neural network that models the structure and function of the part of the brain known as the cerebellum. It exhibits properties such as generalization, learning interference, discrimination, and forgetting that are characteristic of motor learning in biological creatures. Its primary advantages are its

ability to learn very fast and approximate a wide variety of non-linear functions [8, 12, 13].

However, there are some limitations of CMAC:

- Uniform quantization of the input space. The number of regions in each input dimension affects the size of the required memory. Hence, data may not be stored efficiently using a uniformly quantized structure.
- Output discretization i.e. discrete output space. Close inputs may be mapped to identical set of cells, hence generating identical outputs. Modification is required in order to achieve a more smoothed response.
- These limitations motivated the incorporation of fuzzy system into CMAC neural network. Some fuzzy inference schemes have been proposed, such as Compositional Rule of Inference (CRI) [9], Truth Value Restriction (TVR) [10], and Approximate Analogical Reasoning Scheme (AARS) [7]. AARS will be discussed in detail.

3.1 Fuzzy CMAC

Researches had been done to overcome the limitation of the CMAC by combining the capabilities of both fuzzy system and neural network. By doing so, CMAC is transformed into a white box whereby fuzzy rules could be generated to interpret the operation of neural network.

Kim and Lin [3] investigated an adaptive technique for input space quantization through mapping function. Learning starts with uniform quantization. But, the quantization intervals for areas with larger variation are compressed and those with small variation are extended. Hence learning accuracy is improved and storage requirement is reduced.

FCMAC using fuzzy quantization was proposed by Shu [5]. This is achieved by using the Discrete Incremental Clustering (DIC) [6], which dynamically forms the necessary fuzzy clusters. The input space is divided into regions and the quantization ratio in that region is proportional to the number of fuzzy sets that cover the region.

3.2 Discrete Incremental Clustering

Clustering techniques may be classified into hierarchical-based and partition based techniques. The main drawback of hierarchical clustering is that it is static and points committed to a given cluster in the early stages cannot be moved to a different cluster. Partition-based technique on the other hand is dynamic, and data points can be moved from one

cluster to another. However, it needs prior knowledge such as the number of classes. Such information may be unknown and is difficult to estimate in data sets. It also suffers the stability-plasticity dilemma where new information cannot be learned without the risk of eroding previously learned information. These deficiencies are the main motivation behind the development of DIC technique [6].

In a neural network, DIC will be responsible for the self-organizing part. It uses raw numerical values of a training data set with no pre-processing. The full algorithm of DIC can be seen in [6]. DIC requires five parameters: a plasticity parameter β , a tendency parameter TD, an input threshold IT, an output threshold OT and a fuzzy set support parameter SLOPE.

3.3 AARS

Analogical Reasoning Scheme AARS is a fuzzy inference rules scheme proposed by I.B. Turksen [7]. Its fuzzy rule can be expressed as:

Rule: IF x is A , THEN y is B , threshold = τ
 Fact: x is \tilde{A}

There are three key items for this scheme: a similarity measure (SM) for antecedent matching; a threshold value (τ) for decision of rule firing; and a modification function (MF) to deduce the consequence set.

Similarity measure (SM) evaluates the similarity between a rule's antecedents A and the observation \tilde{A} . SM between fuzzy sets is derived from distance measure (DM) that can be calculated as follows:

$$SM = (1 + DM)^{-1}, \quad SM \in [0, 1] \quad (1)$$

The distance measure can be defined as:

$$DM(A, \tilde{A}) = |c(A) - c(\tilde{A})| \quad (2)$$

Where $c(\bullet)$ denotes the centroid of the fuzzy set.

The threshold for SM is a value below which the rule is considered too irrelevant to the facts and should be filtered out for consequence deduction.

A modification function (MF) is used to deduce the consequence. It is a function of the rule's consequence B and the similarity measure SM, i.e. $\tilde{B} = MF(B, SM)$.

3.4 FCMAC AARS

Each of the memory cells can contain either fuzzy sets or its selection of fuzzy sets. The output dimension is also pre-clustered, and O_k denotes the k^{th} output clusters. Since the FCMAC AARS is implemented for a speaker verification system, there are only two outputs, O_1 and O_2 .

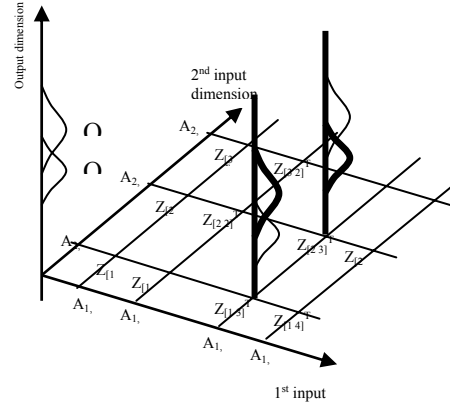


Figure 3 FCMAC-AARS Network Structure with 2 input dimensions

3.5 FCMAC AARS Inference Rules

Given an input tuple $\mathbf{x} = [x_1, x_2, \dots, x_i, \dots, x_j]^T$, the steps to derive the output are described as follows:

Step 1: Input Fuzzification

Since the inputs are crisp-valued, fuzzification has to be performed before they are fed into the fuzzy inference engine:

$$\forall i \in \{1, 2, \dots, I\} \quad X_i = F(x_i) \quad (3)$$

$F(x_i)$ represents a fuzzifier that translates a real value into a fuzzy set. The most commonly used fuzzifier is singleton fuzzifier, denoted by $F_s(\chi)$, is defined by:

$$\mu_{F_s(x_0)}(x) = \begin{cases} 1 & \text{if } x = x_0 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Other form of fuzzifier can be defined if the situation requires.

Step 2: Rule Selection

Given the fuzzified input from step 1, the neighbouring clusters can be identified as follows:

$$\forall i \in \{1, 2, \dots, I\} \quad S_i = \begin{cases} \{1\} & \text{if } c(A_{i,1}) > c(X_i) \\ \{J_i\} & \text{if } c(A_{i,J}) < c(X_i) \\ \{j, j+1\} & \text{if } \exists j : c(A_{i,j}) \leq c(X_i) \leq c(A_{i,j+1}) \end{cases} \quad (5)$$

Where $c(A)$: Centroid of fuzzy set A , S_i : Collection (i.e., a Set) of indices of selected neighboring clusters in i^{th} input dimension

Step 3: Antecedent Matching

The antecedent matching means the comparison between the fuzzified inputs derived in step 1 with the antecedents of rules selected from step 2, which is evaluated in terms of similarity measure (SM).

$$\forall i \in \{1, 2, \dots, I\}, \forall s \in S_i, SM_{i,s} = SM(A_{i,s}, X_i) \quad (6)$$

Where

- I : Total number of input dimensions
- S_i : Collection of indices of selected neighboring clusters for i^{th} input dimension, which is derived in step 2.
- $SM_{i,s}$: Similarity measure between fuzzified input X_i and cluster $A_{i,s}$
- $SM(\bullet)$: Formula for computing SM between 2 fuzzy sets.

Step 4: Rule Fulfillment

The overall similarity, or aggregate similarity measure (ASM) between the input tuple and rule's antecedent is taken as the average of SMs from all input dimensions [7]. The firing decision of a rule is also determined by comparing ASM and AARS threshold τ .

$$\forall n \in N, ASM_n = \frac{1}{I} \sum_{i=1}^I SM_{i,q_{n,i}} \quad (7)$$

$$\delta_n = \begin{cases} 1 & \text{if } ASM_n \geq \tau \\ 0 & \text{if } ASM_n < \tau \end{cases}$$

Where

- N : Number of selected rules, derived in step 2
- $q_{n,i}$: i^{th} component of index pattern of n^{th} selected rule, derived in step 2
- ASM_n : Aggregate similarity measure between input tuple and the antecedent of n^{th} selected rule
- τ : AARS threshold, tunable
- δ_n : Firing decision for n^{th} selected rule

Step 5: Consequence Derivation

The consequence derivation is used to infer the modified consequence based on ASM and the rule's consequence using AARS modification function.

$$\forall n \in N, \tilde{O}_n = \begin{cases} MF^x(O_{z_{d_n}}, ASM_n) & \text{if } \delta_n = 1 \\ \emptyset & \text{if } \delta_n = 0 \end{cases} \quad (8)$$

Where

- \tilde{O}_n : The modified consequence for n^{th} selected rule
- z_{d_n} : Memory cell content indexed by d_n , essentially an index of the output cluster (for fuzzy-output FCMAC AARS). d_n is derived in step 2.

$O_{z_{d_n}}$: Consequence (an output cluster) of n^{th} selected rule.

δ_n & ASM_n : Firing decision & Aggregate similarity measure for the n^{th} selected rule, both of which are derived in step 4.

Step 6: Output Deduction

The consequences from selected rules are combined to produce a single output o as the system output. The output is deduced from a number of fuzzy sets; the process is usually known as defuzzification and is described as follows:

$$o = \frac{\sum_{n=1}^N \frac{c(\tilde{O}_n)}{M(\tilde{O}_n)} \cdot \delta_n}{\sum_{n=1}^N \frac{1}{M(\tilde{O}_n)} \cdot \delta_n} \quad (9)$$

3.6 FCMAC AARS Learning Rule

The goal for learning in FCMAC AARS is to identify the output cluster for each cell. Hence the training data need to be fed into the network only once.

During the training phase, every cell in associative memory keeps a record of *accumulated similarity measure* for each output cluster. To be specific, the *accumulated SM* of the k^{th} output cluster kept by a cell indexed by d is denoted by $SMacc_{d,k}$. In the end the cluster with largest $SMacc$ wins and gets selected as the rule's consequence.

Let O^d denote the fuzzified desired output o^d , i.e., $O^d = F(o^d)$. The similarity between O^d and its neighboring clusters can be determined in the same way as for input dimensions, which is described in inference phase, step 2. Let S^O denotes the collection of indices of selected neighbouring output clusters.

The increment of $SMacc$ is determined as follows:

$$\forall s \in S^O, SMacc_{d_{\hat{n}},s}^{(T+1)} = SMacc_{d_{\hat{n}},s}^{(T)} + ASM_{\hat{n}} \cdot SM(O^d, O_s) \quad (10)$$

Where $SMacc_{d_{\hat{n}},s}^{(T)}$ is the accumulated SM after T^{th} update for s^{th} output cluster in cell located by $d_{\hat{n}}$. The equation above implies the increment of $SMacc$ is proportional to the strength of the relationship between the input pattern and output cluster, which is determined by both ASM (antecedent matching) and $SM(O^d, O_s)$ (consequence matching).

After all training samples are fed, the final selection of output cluster by cell d is

$$\mathbf{Z}_d = k_d \text{ such that } SMacc_{d,k_d} = \max_{k \in K} SMacc_{d,k} \quad (11)$$

That is, the output cluster selected by the cell is the one that has accumulated the most similarity measures during training, hence the most likely consequence of that fuzzy rule.

4. Speaker Verification System

This section implemented speaker verification systems using Mel Frequency Cepstrum Coefficients (MFCC) as the input features to FCMAC AARS. Benchmarking was also done, by replacing FCMAC AARS with other neural networks. The performance of each neural network in verifying the speakers was compared and analyzed.

The experiments were completed in Center for Computational Intelligence in Nanyang Technological University.

4.1 Training and Testing Data

The speech database used in the experiments is known as the Texas Instruments Speaker-Independent Connected-Digit Corpus (TIDIGITS). It is available from the Linguistic Data Consortium (LDC) in University of Pennsylvania. The TIDIGITS consists of 326 speakers (111 men, 114 women, 50 boys and 51 girls) each pronouncing 77 digit sequences. The corpus was collected at Texas Instrument in 1982 in a quiet acoustic enclosure using an Electronic Voice RE-16 Dynamic Cardioid microphone, digitized at 20 kHz. The waveform files are in the NIST SPHER format.

In this experiment 10 speakers (5 men and 5 women) are used with 70 speeches each. But, before implementing the experiment, there were some preparations to be done. First, each experiment required two sets of data as the input sources of the network. The first set contained the input features for the networks' training purposes and the other one contained the input features that were going to be tested after the networks finished their training.

The features of the speeches are extracted with 256 samples per frame, 60% overlap and using the hamming window. The features extracted are 1-40 of MFCC and DMFCC.

There were 2 classes that were considered in these experiments; class 1 was the class of the speaker being verified and class 0 represented the other 9 speakers that were not being verified at that time. The networks were expected to be able to accept class 1 data and to reject the class 0 data.

4.2 Testing Methods

The measurements used in the experiments are False Acceptance Rate (FAR), which is the frequency of false acceptance of invalid user, and False Rejection Rate (FRR), which is the frequency of false rejection of valid user.

Two types of testing methods are used in the experiments:

- Local Recall, which is the same as Memory Recall where the training data are the same as the testing data.
- Global Recall, where the testing data are totally different from the training data.

4.3 Experiment Result and Analysis

The testing results using FCMAC AARS can be summarized in the following tables and figures.

Table 1: 12 MFCC - Local Recall

local recall (MFCC 1-12)

Speaker	FR	Total	FRR (%)	FA	Total	FAR (%)	train time (s)	test time(s)
m1	2	70	2.857143	15	630	2.380952	49.06	39.375
m2	3	70	4.285714	10	630	1.587302	44.455	38.81
m3	1	70	1.428571	17	630	2.698413	50.235	39.675
m4	0	70	0	19	630	3.015873	45.614	38.156
m5	2	70	2.857143	14	630	2.222222	48.152	39.424
f1	1	70	1.428571	18	630	2.857143	49.17	40.45
f2	0	70	0	20	630	3.174603	46.263	40.226
f3	2	70	2.857143	16	630	2.539683	50.12	39.18
f4	3	70	4.285714	11	630	1.746032	47.851	39.15
f5	0	70	0	19	630	3.015873	48.12	40.67
Mean				2		2.52381	47.904	39.5116

Mean of test time = 0.056 s

Table 2: 12 DMFCC – Local Recall

local recall (DMFCC 1-12)

Speaker	FR	Total	FRR (%)	FA	Total	FAR (%)	train time (s)	test time(s)
m1	2	70	2.857143	8	630	1.269841	47.942	39.216
m2	1	70	1.428571	6	630	0.952381	46.498	39.523
m3	1	70	1.428571	7	630	1.111111	48.747	38.784
m4	2	70	2.857143	5	630	0.793651	49.996	39.046
m5	0	70	0	6	630	0.952381	49.437	40.308
f1	1	70	1.428571	4	630	0.634921	48.133	38.95
f2	1	70	1.428571	7	630	1.111111	47.845	39.831
f3	0	70	0	6	630	0.952381	47.965	38.354
f4	1	70	1.428571	7	630	1.111111	46.28	39.395
f5	2	70	2.857143	8	630	1.269841	47.569	39.578
Mean			1.571429			1.015873	48.0412	39.2985

Mean of test time = 0.056 s

It can be seen from Table 1 and Table 2 that local recall gives very good result with range of error is only from 0% to 2.5%. But, local recall test result may be misleading hence global recall testing needs to be

done to really measure the performance of the system. Training time and testing time for each case is taking not more than 1 second which could be considered satisfactory since every case is matched with the 70 trained data.

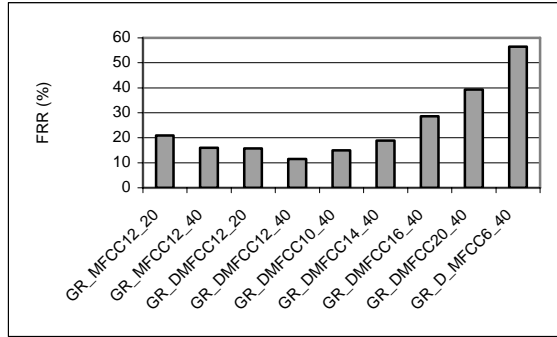


Figure 4: FRR of various testing methods

Legend (from left to right) :
 GR_MFCC12_20: Global Recall testing method with 20 training data – 12 MFCC
 GR_MFCC12_40: Global Recall testing method with 40 training data – 12 MFCC
 GR_DMFC12_20: Global Recall testing method with 20 training data – 12 DMFCC
 GR_DMFC12_40: Global Recall testing method with 40 training data – 12 DMFCC
 GR_DMFC10_40: Global Recall testing method with 40 training data – 10 DMFCC
 GR_DMFC14_40: Global Recall testing method with 40 training data – 14 DMFCC
 GR_DMFC16_40: Global Recall testing method with 40 training data – 16 DMFCC
 GR_DMFC20_40: Global Recall testing method with 40 training data – 20 DMFCC
 GR_D_MFCC6_40: Global Recall testing method with 40 training data – 6 MFCC and 6 DMFCC.

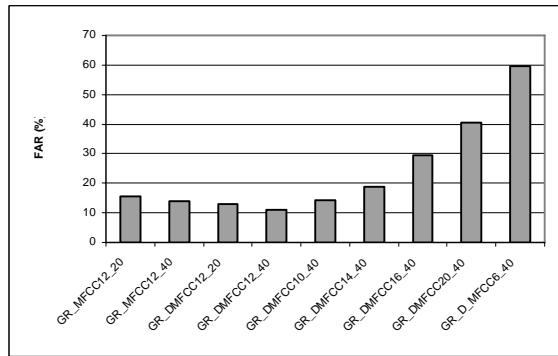


Figure 5: FAR of various testing methods

The global recall with 40 training data gives a better result than the one with 20 training data since it has more data that it can learn from to have a more precise model of the speaker. The global recall with 40 training data gives an average of 11.45 % FRR and 11.02 % FAR while the one with 20 training data gives 16 % and 14 % respectively. As can be seen in Figure 4 and Figure 5 that for the same number of training data, using DMFCC as input for the system gave better result compared to when using MFCC as the input. Characteristics of each speaker can be extracted more uniquely when using DMFCC.

Other than using 12 MFCC and 12 DMFCC, different amount of coefficients are also used. 10, 14, 16, 18, and 20 MFCC and DMFCC were used to compare

which one would give the best result. The global recall with 40 training data was used as the comparison standard of the FAR and FRR. The testing results of global recall with 40 training data with DMFCC as the inputs were used to derive the ROC curve of the family of FCMAC AARS, showing their Equal Error Rate, as shown in Figure 7.

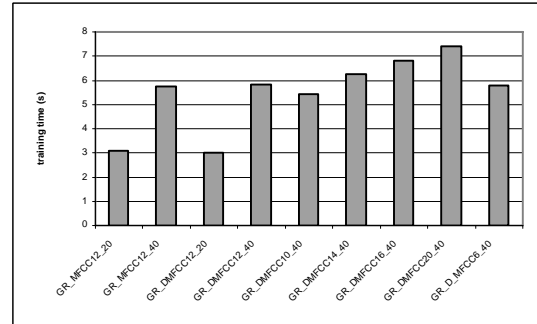


Figure 6: Training time of various testing methods

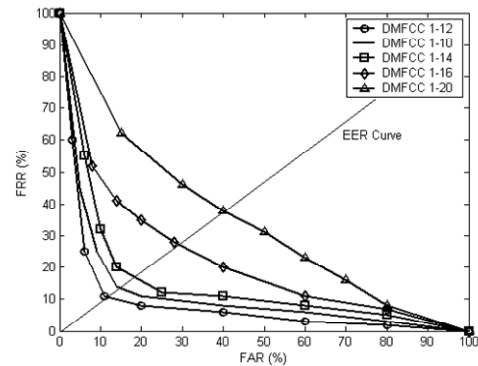


Figure 7: ROC Curve for FCMAC AARS

The amount of coefficients that gives optimum result is 12. The result was not getting better when more coefficients are used. This may be caused by the ‘curse of dimensionality’ where the needed data for training and testing grow exponentially with the dimensionality of the input space.

4.4 Testing with other Neural Networks

GenSoFNN and MLP were also used for testing to have a comparison with the result of FCMAC AARS. The comparison was made with the global recall with 40 training data. The summary of the result is shown in Table 3.

Table 3: Benchmark tests with other NNs

Neural Network	FAR	FRR	Train Time (s)	Test Time (s)
FCMAC AARS	11.45	11.02	5.8064	0.044
GenSoFNN	12.8	6.231	1485.7	0.878
MLP	9.2	3.987	987.5	0.1497

FCMAC AARS did not give the best FAR and FRR, but the training and testing time was much shorter compared to GenSoFNN and MLP. This is because GenSoFNN and MLP have higher complexity calculation, although FCMAC AARS may require bigger storage space.

5. Conclusions

Many applications have been made based on the combination of Fuzzy System and Neural Network. In this paper FCMAC AARS, short for Approximate Analogical Reasoning Scheme based Fuzzy CMAC, has been investigated and implemented for a speaker verification system. The FCMAC AARS use Discrete Incremental Clustering as its clustering technique. Spectrum-based features have been found to be the most effective features for speaker verification system. In this paper, Mel Frequency Cepstrum Coefficients (MFCC) has been chosen to the features for the system.

In order to study the behavior and performance of the speaker verification system, some other neural networks, MLP and GenSoFNN, were used as the speaker modeling/pattern matching process. The performance was analyzed based on the False Rejection Rate, False Acceptance Rate, Equal Error Rate, training and testing time. The result shows that FCMAC AARS needs much less time for training time. The EER are around 11% which is not very satisfying, but has some potential to be developed further to reduce the EER. The number of DMFCC used that gave the optimum result (i.e. minimum EER) was 12.

This paper confirms the advantage of CMAC in its ability to learn very fast. Hence, FCMAC AARS is applicable for speaker verification systems. Its result can actually get better because of CMAC's localized generalization property, which means that the rules generated during training are only affected by data in/near their sphere of influence. MLP and GenSoFNN, in their learning and tuning tend to affect more rules since the rules shared common labels/nodes.

The following things are recommended for future work that can be conducted:

- Use of other method of clustering method such as invariant clustering method or fuzzy c-means.
- Use of other feature extraction method that will produce other coefficients such as LPCC.
- Use of other pattern matching technique such as HMM after the input data is clustered using the Discrete Incremental Clustering technique, which may model the speaker better.

6. References

- [1] Bishop, C.M., "Neural Networks for Pattern Recognition", Oxford University Press, Oxford, UK, 1995.
- [2] Campbell, J.P., "Speaker Recognition: A tutorial", Proceedings of the IEEE, vol. 85, pp. 1437-1462, 1997.
- [3] Kim, H., and Lin, C.S., "Use of adaptive resolution for better CMAC learning", International Joint Conference on Neural Networks, IJCNN, vol. 1, 517-522, 1992.
- [4] Reynolds, D., and Heck, L.P., "Automatic Speaker Recognition", AAAS 2000 Meeting, Humans, Computers and Speech Symposium, 2000.
- [5] Shu, Z.G., "Fuzzy associative memory for the automatic control of a continuous variable transmission control in an automobile", Report on Honours Year Project, NTU, Singapore, 2002.
- [6] Tung, W. L., and Quek, C., "DIC: a novel discrete incremental clustering technique for the derivation fuzzy membership functions", Proceedings of 7th Pacific Rim International Conference on Artificial Intelligence, Tokyo, Japan, 2002.
- [7] Turksen, I.B., and Zhong, Z., "An approximate analogical reasoning schema based on similarity measures and interval-valued fuzzy sets," Fuzzy Sets Syst., vol. 34, pp. 323-346, 1990.
- [8] Wahab, A., Tan, E.C., and Abut, H., "HCMAC Amplitude Spectral Subtraction for Noise Cancellation," Proceedings of 8th International Conference on Neural Information Processing, Shanghai, China, 2001.
- [9] Zadeh, L.A., "Outline of a new approach to the analysis complex systems and decision processes", IEEE Trans. Syst., Man, Cybern., vol. SMC-3, no. 1, pp.28-44, 1973.
- [10] Zadeh, L.A., "The concept of a linguistic variable and its applications to approximate reasoning I, II, III," Inf. Sci., vol. 8, pp. 199-249; pp.301-357, 1975; and vol. 9, pp. 43-80, 1975.
- [11] Zaiyi, G., "FCMAC-AARS: Approximate Analogical Reasoning based Fuzzy CMAC", Final Year Project, School of Computer Engineering, NTU, Singapore, 2003.
- [12] Wahab, A; Ng, GS; Dickiyanto, R, "Speaker authentication system using soft computing approaches", NEUROCOMPUTING, 68: 13-37 OCT 2005.
- [13] Ang, KK; Quek, C; Wahab, A, "MCMAC-CVT: a novel on-line associative memory based CVT transmission control system", NEURAL NETWORKS, 15 (2): 219-236 MAR 2002