Hunny Pahuja, Jitender Chhabra, Ajay Khokhar / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 3, May-Jun 2012, pp.1466-1476

Ant colony optimization-based selected features for Text-independent speaker verification

Hunny Pahuja*, Jitender Chhabra, Ajay Khokhar*****

*(Department of Electronics and Communication, Lovely Professional University, Punjab,)
** (Department of Electronics and Communication, Sat Priya Group of Institutions, Rohtak,)
*** (Department of Electronics and Communication, Sat Priya Group of Institutions, Rohtak,)

ABSTRACT

With the growing trend toward remote security verification procedures for telephone banking, biometric security measures and similar applications, automatic speaker verification (ASV) has received a lot of attention in recent years. The complexity of ASV system and its verification time depends on the number of feature vectors, their dimensionality, the complexity of the speaker models and the number of speakers. At present there are several methods for feature selection in ASV systems. To improve performance of ASV system we present another method that is based on ant colony optimization (ACO) algorithm. After feature reduction phase, feature vectors are applied to a Gaussian mixture model universal back-ground model (GMM-UBM) which is a text-independent speaker verification model. The results of experiments indicate that with the optimized feature set, the performance of the ASV system is improved. Moreover, the speed of verification is significantly increased since by use of ACO, number of features is reduced over 80% which consequently decrease the complexity of our ASV system.

Keywords – **Ant colony optimization (ACO), Gaussian mixture model universal background model (GMM-UBM), Genetic algorithm (GA), Feature selection, Speaker Verification.**

1. INTRODUCTION

Automatic speaker recognition (ASR) system are generally divided into two categories, namely: automatic speaker identification (ASI) system which are designed to answer the question "who is the speaker?" or automatic speaker verification (ASV) systems that aim to answer the question "is the speaker who they claim to be?"

ASV refers to the task of verifying speaker's identity using speaker-specific information contained in speech signal. Speaker verification methods are totally divided into text-dependent and text-independent applications. When the same text is used for both training and testing, the system is called to be text-dependent while the text-independent operation, the text is used to train and test of the ASV system is completely unconstrained. Text-independent speaker verification requires no restrictions on the type of input speech. In contrast, text-independent speaker verification usually gives less performance then text-dependent verification, which requires text input to be the same sentence as training data. (Xiang & Berger, 2003).

Application of speaker verification can be found in biometric person authentication such as an additional identity check during credit card payments over the internet while, the potential application of speaker identification can be found in multi-user systems.

The objective of FS is to simplify a dataset by reducing its dimensionality and identifying relevant underlying features without sacrificing predictive accuracy.

In real world problems, FS is a must sue to the abundance of noisy, irrelevant or misleading features. Selected feature should have high inter-class variance and low inter class variability. Ideally they should also be as independent of each other as possible in order to minimize the redundancy.

Among too many methods that are proposed for FS, population based optimization algorithm such as genetics algorithm (GA)-based method and ant colony optimization (ACO)-based method have attracted a lot of attention. These methods attempt to achieve better solutions by application of knowledge from previous iterations.

Genetic algorithm is optimization techniques based on the mechanism of natural selection. They used operation found in natural genetics to guide itself through the paths in the search space (Siedlecki & Skansky, 1989). Because of their advantages, recently, Gas have been widely used as tool for feature selection in data mining (Srinivas & Patnik, 1994).

In our previous work, we have proposed an ACO algorithm for feature selection in GMM-based ASV system. In this paper we propose some modification to the algorithm and apply it to larger feature vectors containing mel-frequency cepstrail coefficient (MFSSs) and their delta coefficients, two energies, further below

The rest of this paper is organized as follows. Section 2 presents a brief overview of ASV systems. Feature selection methods are described in section 3. Ant colony optimization is described in section 4. Section 5 explains the proposed feature selection algorithm. Genetic algorithm is described in section 6. Section 7 reports computational experiments. It also includes a brief

discussion of the results obtained and finally the conclusion and future works are offered in the last section.

2. An over view of ASV system

The typical process is most proposed ASV system involves some forms of processing of the data (silence removal) and feature extractions, followed by some form of speaker modelling to estimate class dependent feature distributions (see fig. 1). A comprehensive over view can be found in Camp-bell (1997). Adopting this strategy the ASV problem can be further divided into the two problem domains of:

1) Pre-processing, feature generation and selection
2) Speaker modelling and matching.

2.1 Feature extraction

The original signal, the speech waveform, contains all information about the speaker, and each step i the extraction process can only reduce the mutual information or leave it unchanged. The objective of the feature extraction is to reduce the dimension of the extracted vectors and thereby reduce the complexity of the system. The main speaker-discriminating information as possible into as few features as possible.

The choice of features in any proposed ASV system is of primary concern. Because if the feature set does not yield sufficient information then trying to estimate class dependent feature distributions is futile (Basiri, Ghasem-Aghaee, & Aghdam, 2008). Most feature extraction techniques in speaker verification were original used in speech recognition. However, the focus in using these techniques was shifted to extract features with high variability among people.

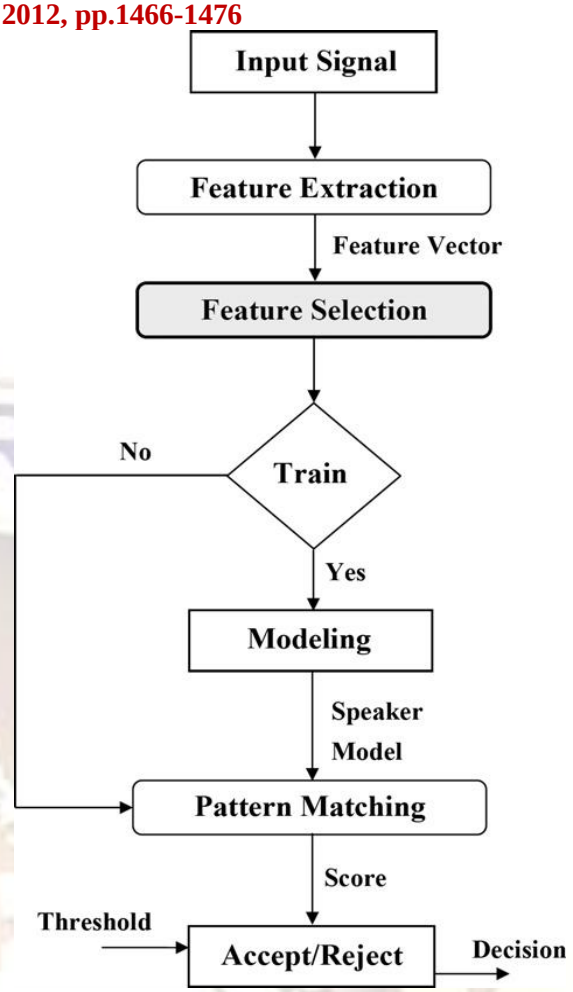

Fig. 1 Overview of the speaker- verification process

Most commonly used features extraction techniques, such as mel-frequency cepstral coefficients (MFCCs) and linear prediction cepstral coefficients (LPCCs) have been particularly popular for ASV systems in recent years. This transforms give a highly compact representation of the spectral envelope of a sound (Cheung-chi, 2004). Delta-features, regardless on what features they are based, can be computed as a one-to-one function of the features themselves. Therefore, the delta-features do not contain more information than is already in the features, and from the theory, no gain can be achieved by using them together with the features. However, the delta-features can be used as a simplified way of exploiting inter-feature dependencies in sub-optimal schemes (Day & Nandi, 2007).

2.2 Speaker modelling

The speaker modelling stage of the process varies more in the literature. The purpose of speaker modelling is characterizing an individual that is enrolled into an ASV system with the aim of defining a model (usually feature distribution values). The three most popular methods in previous works are Gaussian mixture models (GMM) (Cheung-chi, 2004; Reynolds & Rose, 1995), Gaussian mixture models universal background model (GMM-

UBM) (Neiberg, 2001; Reynolds, Quatieri, & Dunn, 2000) and vector quantization (VQ) (Linde, Buzo, & Gray, 1980). Other techniques such as decision trees (Navratil, Jin, Andrews, & Campbell, 2003), support vector machine (SVM) (Wan, 2003) and artificial neural network (ANN) (Wouhaybi & Al-Alaou, 1999) have also applied. In this paper, GMM-UBM is used for speaker modelling.

2.3 GMM-UMB approach

GMM-UBM is the predominant approach used in speaker recognition systems, particularly for text-independent task (Kanan et al., 2007). Given a segment of speech Y and a speaker S, the speaker verification task consists in determining whether Y was spoken by S or not. This task is often stated as basic hypothesis test between two hypotheses: Y is from the hypothesized speaker S(H0), and Y is not from the hypothesized speaker S(H1). A likelihood ratio (LR) between these two hypotheses is estimated and compared to a decision threshold U. The LR test is given by:

$$LR(Y,H_O,H_1)=\frac{P(Y|HO)}{P(Y|H1)} \quad (1)$$

where Y is the observed speech segment, $p(Y|H_0)$ is the likelihood function for the hypothesis H_0 evaluated for Y, $p(Y|H_1)$ is the likelihood function for H_1 and Φ is the decision threshold for accepting or rejecting H_0. If $LR(Y, H_0, H_1) > \Phi$, H_0 is accepted else H_0 is rejected. A model denoted λ_{hyp} represents H_0; it is learned using an extract of speaker S voice. The model λ_{UBM} represents the alternative hypothesis, H_1, and is usually learned using data gathered from a large set of speakers. The likelihood ratio statistic becomes Often, the logarithm of this statistic is used giving the logLR (LLR):

$$LLR(Y)=\log p(Y|\lambda_{hyp}) - \log p(Y|\lambda_{UBM}) \quad (2)$$

3 Feature selection approaches

Feature selection is included in discrete optimization problems. The whole search space for optimization contains all possible subsets of features, meaning that its size is:

$$\sum_{s=0}^{n}\binom{n}{s}=\binom{n}{0}+\binom{n}{1}+\cdots+\binom{n}{n}=2^n \quad (3)$$

where n is the dimensionality (the number of features) and s is the size of the current feature subset (Mladenic´,2006). Usually FS algorithms involve heuristic or random search strategies in an attempt to avoid this prohibitive complexity. However, the degree of optimality of the final feature subset is often reduced (Jensen, 2005).

The objectives of feature selection are manifold, the most important ones being:

- To avoid over fitting and improve prediction performance.
- To provide faster and more cost-effective models.
- To gain a deeper insight into the underlying processes that generated the data.

4 Ant colony optimization (ACO)

In the early 1990s, ant colony optimization was introduced by Dorigo and colleagues as a novel nature-inspired meta-heuristic for the solution of hard combinatorial optimization (CO) problems. An ant colony optimization algorithm (ACO) is essentially a system based on agents which simulate the natural behaviour of ants, including mechanisms of cooperation and adaptation. The inspiring source of ACO is the foraging behaviour of real ants (Dorigo & Blum, 2005).

The first ACO algorithm developed was the ant system (AS) (Dorigo, 1992), and since then several improvements of the AS have been devised (Gambardella & Dorigo, 1995, 1996; Stützle & Hoos, 1997). The ACO algorithm is based on a computational paradigm inspired by real ant colonies and the way they function. The underlying idea was to use several constructive computational agents (simulating real ants). A dynamic memory structure incorporating information on the effectiveness of previous choices based on the obtained results, guides the construction process of each agent.

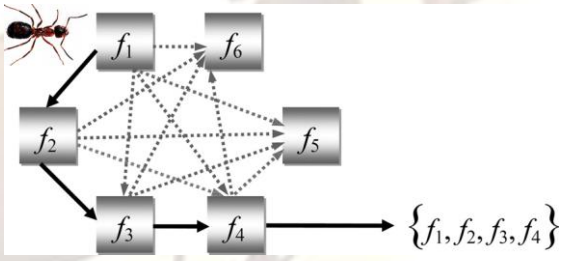

Fig.2. ACO problem representation for FS.

In order to exchange information about which path should be followed, ants communicate with each other by means of a chemical substance called pheromone. As ants move, a certain amount of pheromone is dropped on the ground, creating a pheromone trail. The more ants follow a given trail, the more attractive that trail becomes to be followed by other ants.

4.1. *Ant colony optimization for feature selection*

As mentioned earlier given a feature set of size n, the FS problem is to find a minimal feature subset of size s(s < n) while retaining a suitably high accuracy in representing the original features. Therefore, there is no

concept of path. A partial solution does not define any ordering among the components of the solution, and the next component to be selected is not necessarily influenced by the last component added to the partial solution (Blum & Dorigo, 2004; Leguizamon & Michalewicz, 1999). Furthermore, solutions to an FS problem are not necessarily of the same size. To apply an ACO algorithm to solve a feature selection problem, these aspects need to be addressed. The first problem is addressed by redefining the way that the representation graph is used. 4.1.1. Graph representation The feature selection problem may be reformulated into an ACO-suitable problem. The main idea of ACO is to model a problem as the search for a minimum cost path in a graph. Here nodes represent features, with the edges between them denoting the choice of the next feature. The search for the optimal feature subset is then an ant traversal through the graph where a minimum number of nodes are visited that satisfies the traversal stopping criterion. Fig. 2 illustrates this setup. Nodes are fully connected to allow any feature to be selected next. The ant is currently at node f1 and has a choice of which feature to add next to its path (dotted lines). It chooses feature f2 next based on the transition rule, then f3 and then f4. Upon arrival at f4, the current subset {f1, f2, f3, f4} is determined to satisfy the traversal stopping criterion (e.g. suitably high classification accuracy has been achieved with this subset).
The ant terminates its traversal and outputs this feature subset as a candidate for data reduction (Basiri et al., 2008). Based on this reformulation of the graph representation, the transition rules and pheromone update rules of standard ACO algorithms can be applied. In this case, pheromone and heuristic value are not associated with links. Instead, each feature has its own pheromone value and heuristic value.

4.1.2. *Heuristic desirability.*

The basic ingredient of any ACO algorithm is a constructive heuristic for probabilistically constructing solutions. A constructive heuristic assembles solutions as sequences of elements from the finite set of solution components. A solution construction starts with an empty partial solution. Then, at each construction step, the current partial solution is extended by adding a feasible solution component from the set of solution components (Dorigo & Blum, 2005). A suitable heuristic desirability of traversing between features could be any subset evaluation function for example, an entropy-based measure (Jensen, 2005) or rough set dependency measure (Pawlak, 1991). In proposed algorithm, classifier performance is mentioned as heuristic information for feature selection. The heuristic desirability of traversal and node pheromone levels are combined to form the so-called probabilistic transition rule, denoting the probability that ant k will include feature i in its solution at time step t:

$$P_i^k(t) = \begin{cases} \frac{[\tau_i(t)]^\alpha \cdot [\eta_i]^\beta}{\sum_{u \in J^k} [\tau_u(t)]^\alpha \cdot [\eta_u]^\beta} & \text{if } i \in J^k \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where J^k is the set of feasible features that can be added to the partial solution; τ_i and η_i are respectively, the pheromone value and heuristic desirability associated with feature *i*. α and β are two parameters that determine the relative importance of the pheromone value and heuristic information.
The transition probability used by ACO is a balance between pheromone intensity (i.e. history of previous successful moves), τ_i, and heuristic information (expressing desirability of the move), η_i. This effectively balances the exploitation-exploration trade-off. The best balance between exploitation and exploration is achieved through proper selection of the parameters α and β. If a = 0, no pheromone information is used, i.e. previous search experience is neglected. The search then degrades to a stochastic greedy search. If β = 0, the attractiveness (or potential benefit) of moves is neglected.

4.1.3. *Pheromone update rule*

After all ants have completed their solutions, pheromone evaporation on all nodes is triggered, and then according to Eq. (7) each ant *k* deposits a quantity of pheromone, $\Delta\tau_i^k(t)$ on each node that it has used.

$$\Delta\tau_i^k(t) = \begin{cases} \phi \cdot \gamma(S^k(t)) + \frac{\varphi.(n-|S^k(t)|)}{n} & \text{if } i \in S^k(t) \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where $S^k(t)$ is the feature subset found by ant *k* at iteration *t*, and $|S^k(t)|$ is its length. The pheromone is updated according to both the measure of the classifier performance, $\Upsilon(S^k(t))$, and feature subset length. Φ ϵ [0, 1] and φ = 1-ϕ are two parameters that control the relative weight of classifier performance and feature subset

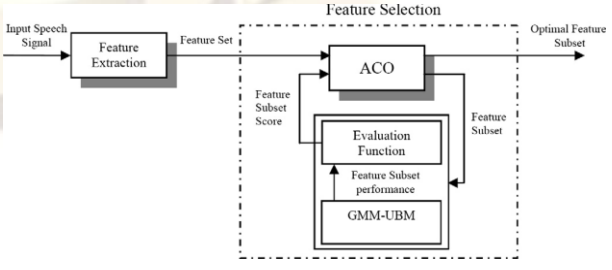

Fig. 3. Overall process of Aco feature selection for ASV.

length. This formula means that the classifier performance and feature subset length have different significance for feature selection task. In our experiment we assume that classifier performance is

more important than subset length, so they were set as $\emptyset = 0.8$, $\varphi = 0.2$.

In practice, the addition of new pheromone by ants and pheromone vaporation are implemented by the following rule applied to all the nodes:

$$\tau_i(t+1) = (1-\rho)\tau_i(t) + \sum_{k=1}^{m} \Delta\tau_i^k(t) + \Delta\tau_i^g(t) \quad (6)$$

where m is the number of ants at each iteration and $p\epsilon(0,1)$ is the pheromone trail decay coefficient. The main role of pheromone evaporation is to avoid stagnation, that is, the situation in which all ants constructing the same solution. g indicates the best ant at each iteration. All ants can update the pheromone according to Eq. (8) and the best ant deposits additional pheromone on nodes of the best solution. This leads to the exploration of ants around the optimal solution in next iterations.

4.1.4. *Proposed feature selection algorithm*

Typically, an ASV system consists of several essential parts including feature extraction and feature selection. After pre-processing of speech signals, feature extraction is used to transform the input signals into a feature set (feature vector). Feature selection is applied to the feature set to reduce the dimensionality of it. This process is shown in Fig. 3. ACO is used to explore the space of all subsets of given feature set. The performance of selected feature subsets is measured by invoking an evaluation function with the corresponding reduced feature space and measuring the specified classification result. The best feature subset found is then output as the recommended set of features to be used in the actual design of the classification system. The main steps of proposed feature selection algorithm are as follows:

1. Initialization.
- Determine the population of ants.
- Set the intensity of pheromone trial associated with any feature.
- Determine the maximum of allowed iterations.

2. Solution generation and evaluation of ants.
- Assign any ant randomly to one feature and visiting features, each ant builds solutions completely. In this step, the evaluation criterion is equal error rate (EER) of the classifier. If an ant is not able to decrease the MSE of the classifier in 10 successive steps, it will finish its work and exit.

3. Evaluation of the selected subsets.

- Sort selected subsets according to classifier performance and their length. Then, select the best subset.

4. Check the stop criterion.

- Exit, if the number of iterations is more than the maximum allowed iteration, otherwise continue.

5. Pheromone updating.

Decrease pheromone concentrations of nodes then, all ants deposit the quantity of pheromone on graph. Finally, allow the best ant to deposit additional pheromone on nodes.

6. Generation of new ants.

- In this step, previous ants are removed and new ants are generated.

7. Go to 2 and continue.

The process begins by generating a number of ants that are then placed randomly on the graph i.e. each ant starts with one random feature. Alternatively, the number of ants to place on the graph may be set equal to the number of features within the data; each ant starts path construction at a different feature. From these initial positions, they traverse nodes probabilistically until a traversal stopping criterion is satisfied. The resulting subsets are gathered and then evaluated. If an optimal subset has been found or the algorithm has executed a certain number of times, then the process halts and outputs the best feature subset encountered. If none of these conditions hold, then the pheromone is updated, a new set of ants are created and the process iterates once more.

5. Genetic algorithm (GA)

The GAs are stochastic global search methods that mimic the metaphor of natural biological evolution (Srinivas & Patnik, 1994). These algorithms are general-purpose optimization algorithms with a probabilistic component that provide a means to search poorly understood, irregular spaces. Instead of working with a single point, GAs work with a population of points. Each point is a vector in hyperspace representing one potential (or candidate) solution to the optimization problem. A population is, thus, just an ensemble or set of hyperspace vectors. Each vector is called a chromosome in the population. The number of elements in each vector (chromosome) depends on the number of parameters in the optimization problem and the way to represent the problem. How to represent the problem as a string of elements is one of the critical factors in successfully applying a GA (or other evolutionary algorithm) to a problem.

5.1. *Genetic algorithm for feature selection*

Several approaches exist for using GAs for feature subset selection. The two main methods that have been widely used in the past are as follow. First is due to Siedlecki and Sklansky (1989), of finding an optimal binary vector in which each bit corresponds to a feature (Binary vector optimization (BVO) method). A '1' or '0' suggests that the feature is selected or dropped, respectively. The aim is to find the binary vector with the smallest number of 1's such that the classifier performance is maximized. This criterion is often modified to reduce the dimensionality of the feature vector at the same time (Yang & Honavar, 1998). The second and more refined technique uses an m-ary vector to assign weights to features instead of abruptly dropping or including them as in the binary case (Punch, Goodman, Pei, Hovland, & Enbody, 1993). This gives a better search resolution in the multidimensional space (Raymer, Punch, Goodman, Kuhn, & Jain, 2000).

6. Experimental results

A series of experiments was conducted to show the utility of proposed feature selection algorithm. All experiments have been run on a machine with 3.0 GHz CPU and 512 MB of RAM. We implement proposed ACO algorithm and GA-based feature selection algorithm in Matlab R2006a. The operating system was Windows XP Professional. The following sections describe TIMIT dataset and implementation results.

6.1. *TIMIT dataset*

The TIMIT corpus (Garofolo et al., 1990) is used in this paper This corpus contains 630 speakers (438 male and 192 female) representing eight major dialect regions of the United States, each speaking ten sentences. There are two sentences that are spoken by all speakers and the remaining eight are selected randomly from a large database.

The speech signal is recorded through a high quality microphone with a sampling frequency of 16 kHz in a quiet environment, with no session interval between recordings. Eight sentences (SX, SI) were used to develop each speaker model, and the remaining 2 SA sentences were used to test each speaker. The 40 speakers included in both the test and train directories were used during the TIMIT (40) trials.

6.2. *Evaluation measure*

The evaluation of the speaker verification system is based on detection error tradeoff (DET) curves, which show the tradeoff between false alarm (FA) and false rejection (FR) errors. Typically equal error rate (EER), which is the point on the curve where FA = FR, is chosen as evaluation measure. We also used detection cost function (DCF) defined as (Reynolds & Rose, 1995):

$$DCF = C_{miss} \cdot FRR \cdot P_{target} + C_{FA} \cdot FAR \cdot (1 - P_{target}) \quad (7)$$

6.3. *Experimental setup*

Various values were tested for the parameters of proposed algorithm. The experiments show that the highest performance is achieved by setting the parameters to values shown in Table 1. Experiments were conducted on a subset of TIMIT corpora consist of 24 male and 16 female speakers of different accent that were selected randomly. Data were processed in 20 ms frames (320 samples) with 50% overlaps. Frames were segmented by Hamming window and pre-emphasized with $\alpha = 0.97$ to compensate the effect of microphone's low pass filter. At first, feature vector was created by extracting MFCCs from silence-removed data for each frame. In the next step, delta coefficients were calculated based on the MFCCs and appended to existing feature vector. Furthermore, two energies were applied to input vectors as described earlier. Then we consider the LPCCs and their delta coefficients respectively and append them to the feature vector. The final feature set contains $F = 50$ features. Table 2 shows the overall set of features.

Finally, verification process was performed using the GMMUBM approach. The performance criterion is due to EER and DCF according to an adopted decision threshold strategy. A single 2048 GMM was trained by pooling all the training data together.

6.4. *Results and discussion*

The verification quality and feature subset length are two criteria that are considered to assess the performance of algorithms. Comparing the first criterion, we noted that both ACO-based and GA-based algorithms reduce the dimensionality of feature space. Furthermore, the ACO-based algorithm selects a smaller subset of features than the GA-based algorithm. Table 3 shows the number of selected

Table 1
GA and ACO parameter settings.

	Population	Iteration	Crossover probability	Mutation probability	Initial pheromone	α	β	ρ
GA	50	100	0.7	0.005	–	–	–	–
ACO	50	100	–	–	1	1	0.1	0.2

features by ACO-based and GA-based approaches. As we can see in Table 3, ACO can degrade dimensionality of features over 80%.

Table 2
The overall feature set.

#	Feature name	Order
1	Mel frequency cepstral coefficient (MFCC)	12
2	Linear prediction cepstral coefficient (LPCC)	12
3	First diff. of MFCC (Δ-MFCC)	12
4	First diff. of LPCC (Δ-LPCC)	12
5	MFCC energy	1
6	Delta MFCC energy	1
	Total	50

Table 3
Selected features of ACO-GMM.

Selection method	Number of selected features	Percentages of selected features (%)
ACO-GMM-UBM (16)	10	20
ACO-GMM-UBM (32)	9	18
ACO-GMM-UBM (64)	10	20
GA-16	16	32
GA-32	17	34
GA-64	16	32

The second performance criterion is due to EER and DCF according to an adopted decision threshold strategy. The EER and DCF for ACO-based and GMM-based algorithms with different number of Gaussian (16, 32 and 64) were shown in Table 4.

Table 4
EER and DCF for GMM-UBM, ACO-based and GMM-based algorithms with different number of Gaussians.

Number of Gaussians	GMM-UBM		ACO		GA	
	EER	DCF	EER	DCF	EER	DCF
16	5.08	0.0563	4.318	0.0401	4.966	0.0554
32	4.56	0.0505	2.634	0.0309	3.679	0.0389
64	6.667	0.0707	4.23	0.0386	4.961	0.0532

Ideally the number of mixtures, M, should approximate the number of natural classes in data. If M is less than the number of natural classes, closely placed clusters will fuse to form larger clusters whose variance is higher. This results in a larger percentage of frames, both the true speaker and imposter, getting average scores. The performance goes down because these frames cannot be discriminated. This is a case of trying to under-fit training data. When M is more than the number of natural classes, larger clusters are broken up into smaller sub-clusters or spurious frames get represented as separate clusters. The new clusters will have lower variance. This would lead to increase in average and low scoring frames and hence would lead to lower performance. This is a case of trying to over-fit training data. As could be seen in Table 4, speaker verification performance is found to increase as M is increased but begins to drop after a point at which over-fitting starts taking place.

DET curves for GA-based and ACO-based algorithms with 16, 32 and 64 Gaussians are shown in Figs. 4–6. From the results, it can be seen that ACO-GMM yields a significant improvement in speed than the baseline GMM approach. The improvement is due to the selection of optimal feature set by ACO algorithm. To further highlight the search process, we graph percent selected features of every ant's current iteration (horizontal coordinate) against classifier performance (vertical coordinate). Each point in the figure is an ant. The process of the ant colony searching for optimal solutions for TIMIT dataset with 32 Gaussian (best results obtained with 32 Gaussian) is given in Figs. 7a–7k.

From the results and figures, we can see that, compared with GA, ACO is quicker in locating the optimal solution. In general, it can find the optimal solution within tens of iterations. If exhaustive search is used to find the optimal feature subset in the TIMIT dataset, there will be tens of billiards of candidate subsets, which is impossible to execute. But with ACO, at the 100th iteration the optimal solution is found.

ACO has powerful exploration ability; it is a gradual searching process that approaches optimal solutions. The running time of ACO is affected more by the problem dimension (feature numbers), and the size of data. For some datasets with more features, after finding a sub-optimal solution, the GA cannot find a better one. However, ACO can search in the feature space until the optimal solution is found. Number of features greatly affects the GA.

ACO comprises a very simple concept, and the ideas can be implemented in a few lines of computer code.

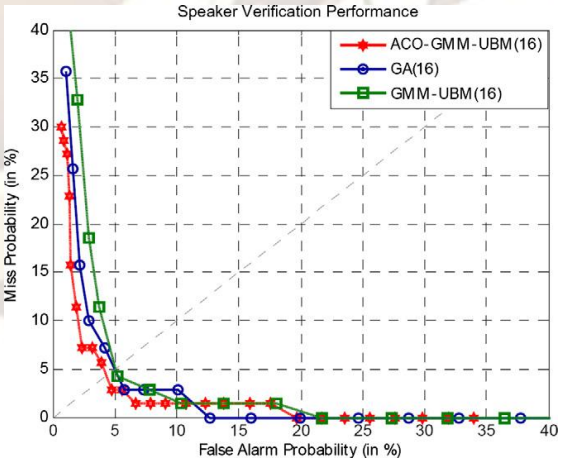

Fig.4. DEsT curves for GMM-UBM,ACO-GMM-UBM and GA with 16 Guassians.

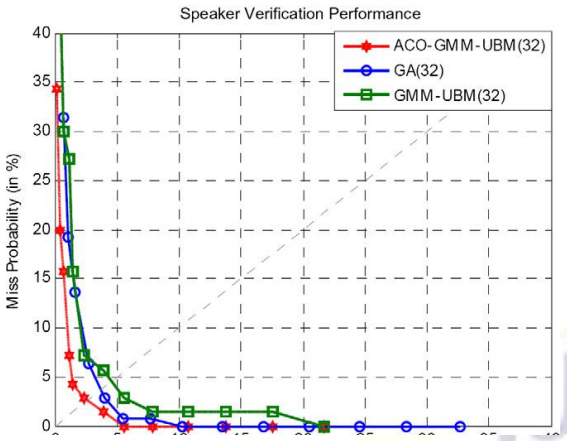

fig.5. DET curves for GMM-UBM,ACO-GMM-UBM and GA with 32 Guassians.

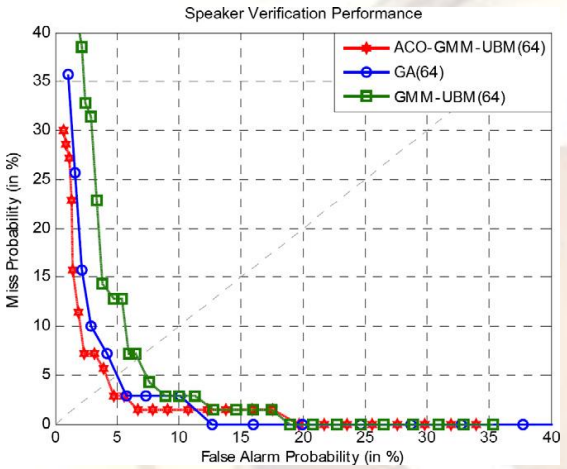

Fig.6. DET curves for GMM-UMB,ACO-GMM-UMB and GA with 64 Gaussians.

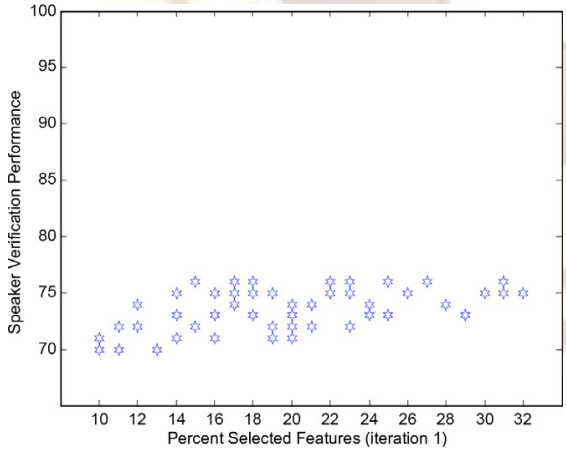

Fig.7.a. iteration 1 of ACO on TIMIT dateset.

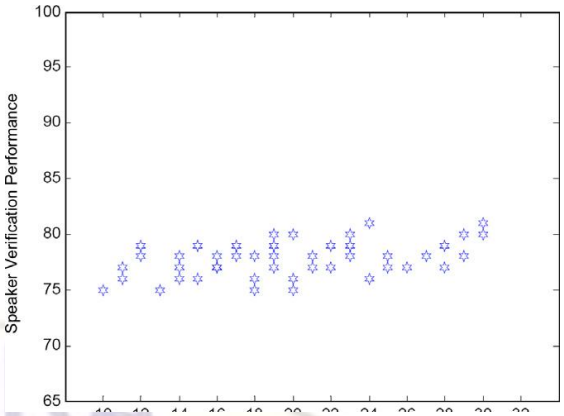

Fig.7b. iteration 10 of ACO on TIMIT dateset.

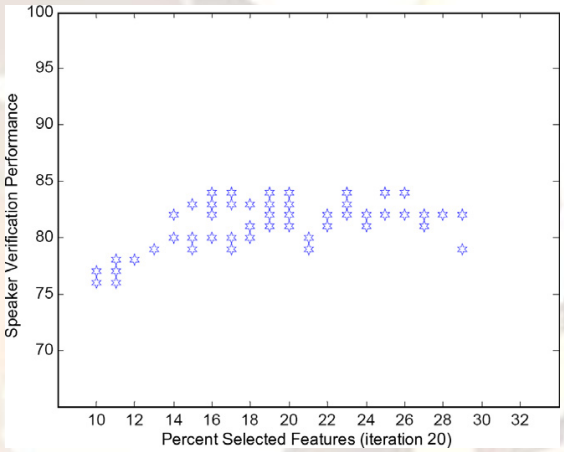

Fig.7c. iteration 20 of aco on TIMIT dataset

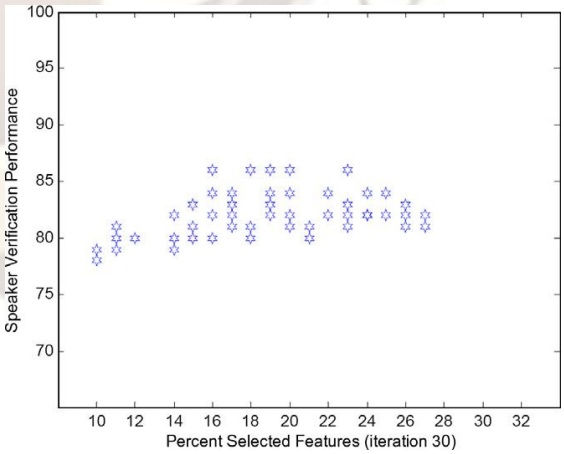

Fig.7d. iteration 30 of ACO on TIMIT dataset

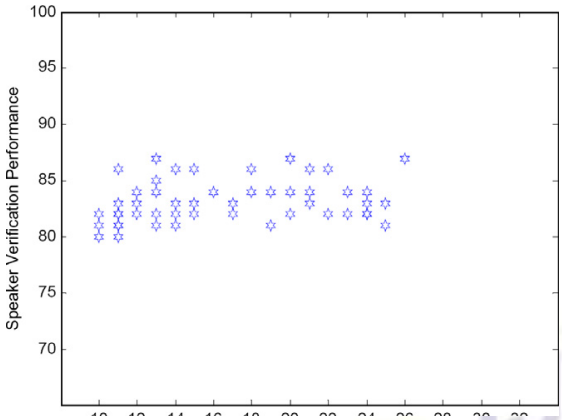

Fig.7e. iteration 40 of ACO on TIMIT dataset

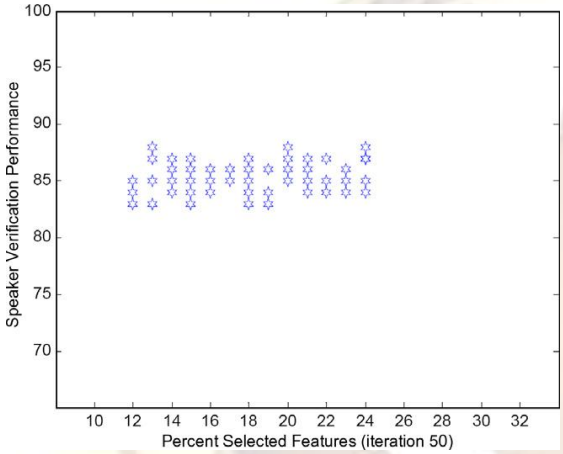

Fig.7f. iteration 50 of ACO on TIMIT datase

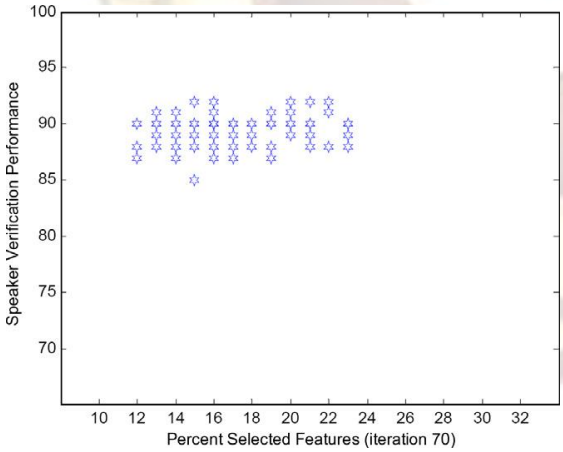

Fig.7g. iteration 60 of ACO on TIMIT dataset

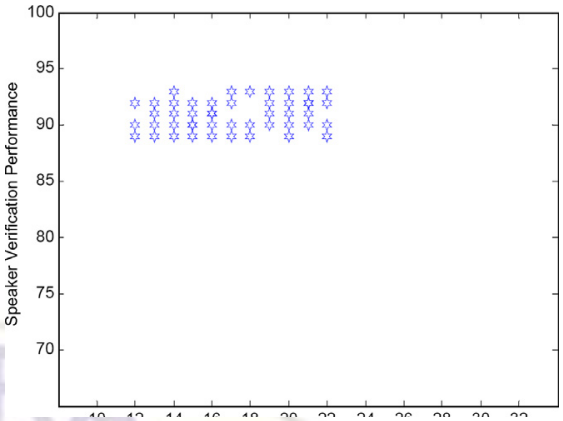

Fig.7h. iteration 60 of ACO on TIMIT dataset

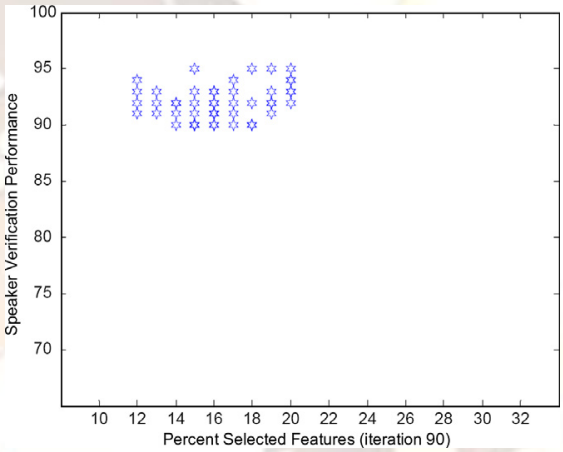

Fig.7i. iteration 70 of ACO on TIMIT dataset

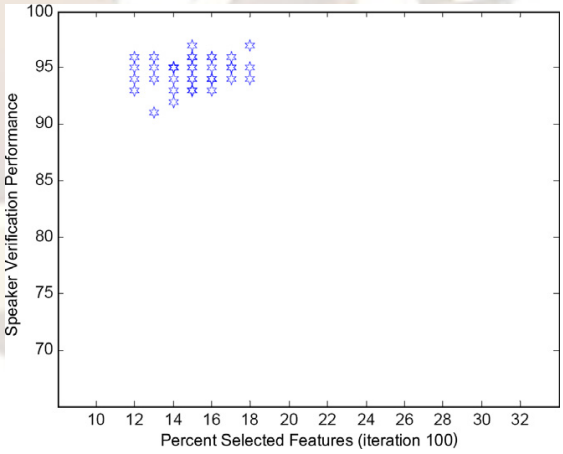

Fig.7j. iteration 80 of ACO on TIMIT dataset

This optimization technique does not suffer, however, from some of the difficulties of GAs; interaction in the colony enhances rather than detracts from progress toward the solution. Further, an ant colony system has memory, which the genetic algorithm does not have.

Changes in genetic populations result in the destruction of previous knowledge of the problem. In ant colony optimization, ants that past optima are tugged to return towards them; all ants retain knowledge of good solutions.

7. Conclusion and future research

In this paper, we have addressed the problem of optimizing the acoustic feature set by ACO technique for text-independent speaker verification system based on GMM-UBM. In our previous work we have proposed an ACO algorithm for feature selection in GMM-based ASV systems (Nemati et al., 2008). In this paper we propose some modifications to the algorithm and apply it to larger feature vectors containing MFCCs and their delta coefficients, two energies, LPCCs and their delta coefficients. ACO selected the most relevant features among all features in order to increase the performance of our ASV system. We compare its performance with another prominent population-based feature selection method, genetic algorithm. The experimental results on a subset of TIMIT database showed that ACO is able to select the more informative features without loosing the performance than GA. The feature vectors size reduced over 80% that led to a less complexity of our ASV system. Moreover, verification process in the test phase speed up because less complexity is achieved by the proposed system in comparison with current ASV systems.

For future work, the authors plan to investigate the performance of proposed ASV system by taking advantage of using VQ and other models instead of GMM-UBM. We also, intend to take the advantage of other swarm intelligence techniques like Particle swarm optimization (PSO) algorithm in ASV systems. Finally, another research direction will involve experiments with different datasets.

REFERENCES

[1] Aghdam, M. H., Ghasem-Aghaee, N., & Basiri, M. E. (2009). Text feature selection using ant colony optimization. Expert Systems with Applications, 36, 6843–6853.

[2] Ani, A. A. (2005). Ant colony optimization for feature subset selection. Transaction on Engineering, Computing and Technology, 4, 3–35.

[3] Basiri, M. E., Ghasem-Aghaee, N., & Aghdam, M. H. (2008). Using ant colony optimization-based selected features for predicting post-synaptic activity in proteins. EvoLNCS. LNCS (vol. 4973). Heidelberg: Springer-Verlag. pp. 12–23.

[4] Bins, J. (2000). Feature selection from huge feature sets in the context of computer vision. Ph.D. dissertation, Department Computer Science, Colorado Stat University.

[5] Blum, C., & Dorigo, M. (2004). The hyper-cube framework for ant colony optimization. IEEE Transactions on Systems, Man, and Cybernetics – Part B 34(2), 1161–1172.

[6] Campbell, J. P. (1997). Speaker recognition: A tutorial. Proceedings of IEEE, 85(9), 1437–1462.

[7] Cheung-chi, L. (2004). GMM-based speaker recognition for mobile embedded systems Ph.D. thesis, University of Hong Kong.

[8] Cohen, A., & Zigel, Y. (2002). On feature selection for speaker verification. In Proceedings of COST 275 workshop on the advent of biometrics on the Internet.

[9] Day, P., & Nandi, A. K. (2007). Robust text-independent speaker verification using genetic programming. IEEE Transactions on Audio, Speech, and Language Processing, 15(1), 285–295.

[10] Dorigo, M. (1992). Optimization, learning and natural algorithms. Ph.D. dissertation, Dipartimento di Electtronica, Politecnico di Milano, Italy.

[11] Dorigo, M., & Caro, G. D. (1999). Ant colony optimization: A new meta-heuristic. In Proceedings of the congress on evolutionary computing.

[12] Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. Theoretical Computer Science, 243–278.

[13] Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony f cooperating agents. IEEE Transaction on Systems, Man, and Cybernetics – Part B, 26(1), 29–41.

[14] Duda, R. O., & Hart, P. E. (1973). Pattern classification and scene analysis. Chichester: John Wiley & Sons.

[15] Gambardella, L. M., & Dorigo, M. (1995). Ant-Q: A reinforcement learning approach to the TSP. In Proceedings of the twelfth international conference on machin learning (pp. 252–260).

[16] Gambardella, L. M., & Dorigo, M. (1996). Solving symmetric and asymmetric TSPs by ant

colonies. In Proceedings of IEEE international conference on evolutionary computation (pp. 622–627).

[17] Ganchev, T., Zervas, P., Fakotakis, N., & Kokkinakis, G. (2006). Benchmarking feature selection techniques on the speaker verification task. In Fifth international symposium on communication systems, networks and digital signal processing, CSNDSP'06 (pp. 314–318).

[18] Garofolo, J. et al. (1990). DARPA TIMIT acoustic–phonetic continuous speech corpus CDROM. National Institute of Standards and Technology.

[19] Jensen, R. (2005). Combining rough and fuzzy sets for feature selection. Ph.D. thesis, University of Edinburgh.

[20] Kanan, H. R., Faez, K., & Hosseinzadeh, M. (2007). Face recognition system using ant colony optimization-based selected features. In Proceedings of the first IEEE symposium on computational intelligence in security and defense applications, CISDA 2007 (pp. 57–62). USA: IEEE Press.

[21] Kwon, S., & Narayanan, S. (2002). Speaker change detection using a new weighted distance measure. In Proceedings of international conference on spoken language processing (ICSLP 2002), Denver, CO (pp. 2537–2540).

[22] Lapidot, I., Guterman, H., & Cohen, A. (2002). Unsupervised speaker recognition based on competition between self-organizing maps. IEEE Transactions on Neural Networks, 13(2), 877–887.

[23] Leguizamon, G., & Michalewicz, Z. (1999). A new version of ant system for subset problems. In Proceedings of IEEE congress on evolutionary computation (pp. 1458– 1465).

[24] Linde, Y., Buzo, A., & Gray, R. M. (1980). An algorithm for vector quantizer design. IEEE Transactions on Communications, 28, 84–95.

[25] Liu, B., Abbass, H. A., & McKay, B. (2004). Classification rule discovery with ant colony optimization. IEEE Computational Intelligence Bulletin, 3(1), 31–35.

[26] Liu, D., & Kubala, F., (1999). Fast speaker change detection for broadcast news transcription and indexing. In Proceedings of 6th European conference speech communication and technology (Eurospeech 1999), Budapest, Hungary (pp. 1031– 1034).

[27] Maniezzo, V., & Colorni, A. (1999). The ant system applied to the quadratic assignment problem. IEEE Transaction on Knowledge and Data Engineering, 11(5), 769–778.

[28] Nemati, S., Boostani, R., & Jazi, M. D. (2008). A novel text-independent speaker verification system using ant colony optimization algorithm. In ICISP2008. LNCS (vol. 5099, pp. 421–429). Berlin, Heidelberg: Springer-Verlag.

[29] Pandit, M., & Kittkr, J. (1998). Feature selection for a DTW-based speaker verification system. IEEE, 796–799.

[30] Raymer, M., Punch, W., Goodman, E., Kuhn, L., & Jain, A. K. (2000). Dimensionality reduction using genetic algorithms. IEEE Transactions on Evolutionary Computing, 4, 164–171.

[31] Reynolds, A., Quatieri, F., & Dunn, R. (2000). Speaker verification using adapted Gaussian mixture models. Digital Signal Processing, 10, 19–41.

[32] Reynolds, D. A., & Rose, R. C. (1995). Robust text-independent speaker identification using Gaussian mixture speaker models. IEEE Transactions on Speech and Audio Processing, 3(1), 72–83.

[33] Srinivas, M., & Patnik, L. M. (1994). Genetic algorithms: A survey. Los lamitos: IEEE Computer Society Press.

[34] Stützle, T., & Hoos, H. H. (1997). MAX–MIN ant system and local search for the traveling salesman problem. In Proceedings of IEEE international conference on evolutionary computation (pp. 309–314).

[35] Susmaga, R. (1998). Parallel computation of reducts. In Proceedings of the first international conference on rough sets and current trends in computing (pp. 450–457). Springer-Verlag.

[36] Wouhaybi, R., & Al-Alaou, M. A. (1999). Comparison of neural networks for speaker recognition. In Proceedings of 6th IEEE international conference electronics, circuits systems (ICECS) (Vol. 1, pp. 125–128).

[37] Xiang, B., & Berger, T. (2003). Efficient text-independent speaker verification with structural Gaussian mixture models and neural network. IEEE Transactions on Speech and Audio Processing, 11(5).