EXTRACTING ADDITIONAL INFORMATION FROM GAUSSIAN MIXTURE MODEL PROBABILITIES FOR IMPROVED TEXT-INDEPENDENT SPEAKER IDENTIFICATION

B. Narayanaswamy

Carnegie Mellon University
ISRI and ECE
5000 Forbes Avenue,Pitt, PA 15213

Rashmi Gangadharaiah

Indian Institute of Science
SERC
Bangalore 560012

ABSTRACT

This paper addresses the problem of robust text-independent speaker identification. A voting mechanism is proposed to combine probabilities generated using Gaussian Mixture Models (GMMs). This algorithm is evaluated on standard data sets and shown to improve performance. This method is found to decrease error rate by up to **68.6%** relative on KING database and **34.9%** relative on SPIDRE. An analysis is performed and a hypothesis is proposed as to why this algorithm does not give as good an identification rate in certain cases. A method of using voting along with the standard GMM method is described which overcomes this limitation. This second method is evaluated and found to decrease error rate by as much as **45.67%** relative on the SPIDRE databases. It is found to give a substantial improvement over conventional GMMs in all the experiments performed. Both the proposed algorithms achieve increased accuracy with negligible increase in computational cost.

1. INTRODUCTION

Speaker recognition is the process of recognizing who is speaking on the basis of information extracted from the speech signal. It has a number of applications including verification of control access permissions to services such as banking over the telephone, corporate database search and voice mail. Speaker identification is the process of determining which registered speaker was the source of a given utterance whereas speaker verification is the process of accepting or rejecting the identity claim of a user based on speech alone. A speaker recognition system is said to be text independent if the registering and test voices are not restricted to speak a particular word, phrase or sentence.

Various models have been applied to the task of text independent speaker identification, such as Vector Codebooks [1], Radial Basis Functions [2], Auto Associative Neural Networks [3] and Gaussian Mixture Models (GMMs) [4]. Of these, GMMs have been the most successful, while still being computationally not very expensive leading to the extensive use of GMM based speaker recognition systems. This paper concentrates on improving the performance of a simple GMM based text independent speaker identification system which involves almost no increase in the computational cost.

The remainder of this paper is organized as follows. Section 2 describes the features used along with an explanation of the classical Gaussian Mixture Model as applied to speaker identification. Section 3 describes the first scheme, a voting based method to improve the performance of GMMs. Section 4 details the evaluation setup used to compare the various methods. Section 5 gives the results when voting is used. Section 6 analyzes why voting shows a loss of performance in certain cases and presents a new method to overcome the limitations of the voting based classifier. Section 7 gives the results of the evaluation of this combination method. Section 8 has some of the conclusions made from the experiments presented.

2. THE CLASSICAL GAUSSIAN MIXTURE MODEL

The features used in this paper for speaker recognition are Mel Frequency Cepstral Coefficients (MFCCs) [5]. In the setup used, the magnitude spectrum from a short frame is processed using a Mel-scale filter-bank. The log energy filter outputs are then cosine transformed to produce cepstral coefficients. The processing is repeated every frame resulting in a series of feature vectors.

The use of GMMs for speaker recognition is described in [4]. A GMM is the weighed sum of M component densities given by the equation,

$$p(\vec{X}|\lambda) = \sum_{i=1}^{M} p_i b_i(\vec{x}) \quad (1)$$

Where $\vec{x}$ is D dimensional speech feature vector, $b_i(\vec{x})$, i=1....M are component densities and p_i,i=1....M are the mixture weights. Each component density is a D dimensional Gaussian pdf of the form,

$$b_i(\vec{x}) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{\frac{1}{2}}} exp\left\{-\frac{1}{2}(\vec{x}-\vec{\mu_i})'\Sigma_i^{-1}(\vec{x}-\vec{\mu_i})\right\} \quad (2)$$

with mean vector $\vec{\mu_i}$ and covariance matrix Σ_i. The mixture weights are such that $\sum_{i=1}^{M} p_i = 1$. Each speaker is represented by a GMM λ_i which is completely parameterized by its mixture weights, means and covariance matrices collectively represented as,

$$\lambda_i = \{p_i, \vec{u_i}, \Sigma_i\} \quad (3)$$

These GMMs are trained separately on each speaker's enrollment data using the Expectation Maximization (EM) algorithm [6]. For computational ease the covariance matrices are constrained to be diagonal.

In speaker identification, given a group of speakers S = $\{1, 2....M\}$, the objective is to find the speaker model which has the maximum a posteriori probability for a given test sequence,

$$\hat{S} = \arg \max_{1 \le k \le M} p(\lambda_k) = \arg \max_{1 \le k \le M} \frac{p(\vec{X}|\lambda_k)p(\lambda_k)}{p(\vec{X})} \quad (4)$$

Assuming that all speakers are equally likely and that the observations are independent, and since p(x) is same for all speakers, this simplifies to

$$\hat{S} = \arg \max_{1 \le k \le M} p(\vec{X}|\lambda_k) = \arg \max_{1 \le k \le M} [\text{prod}(p(\vec{x_i}|\lambda_k)] \quad (5)$$

Thus each GMM outputs a probability for each frame, which is multiplied across all the frames. The classifier makes a decision based on these product posterior probabilities.

3. VOTING FOR SPEAKER IDENTIFICATION

An analysis of the kinds of mistakes made by the GMM based system was performed. It was seen that in many cases, the correct speaker had very low scores in only a few frames, and despite scoring better than all other speakers in all other frames, was not selected as a right speaker. These few frames could be noisy frames or variant frames where the speaker model did not match properly. This phenomenon could also be due to non-optimal models. The effect of these few frames is thought to be amplified because the probabilities of each frame are multiplied to achieve the final posterior probability, thus possibly giving a higher weight to some frames which have a much lower score.

$$p(\vec{X}|\lambda_k) = [\text{prod}(p(\vec{x_i}|\lambda_k)] \quad (6)$$

for i=1,2...n, where n is the total number of frames.

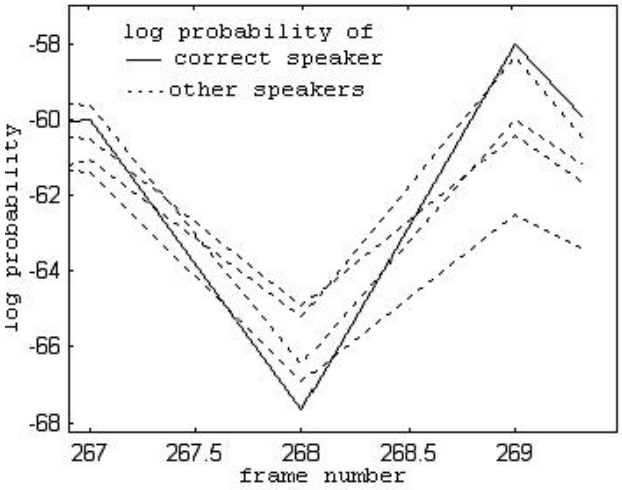

Fig. 1. Dip in log probabilities for a single frame shown for different speakers in a single utterance

An example of the log probabilities for such a frame, is shown in Fig. 1. Many of the utterances which were recognized incorrectly using GMMs had at least five or six such frames. When these few regions were identified and removed by hand many errors made by the system were corrected. To avoid the influence of a few bad frames causing wrong identification there is a need to make the influence of the frames more uniform. So, a voting based combination scheme is suggested, where each frame has a single vote.

In the proposed voting algorithm, each frame is viewed as an independent classifier. Using the GMM parameters each classifier makes an independent decision as to who the speaker is. In the case of classical GMMs the outputs of the frames are the probabilities p($\vec{x_i}|\lambda$) which are then combined by multiplication. In the proposed method the decisions of all the classifiers (frames) are combined by voting. The difference is shown in Fig. 2. Thus in

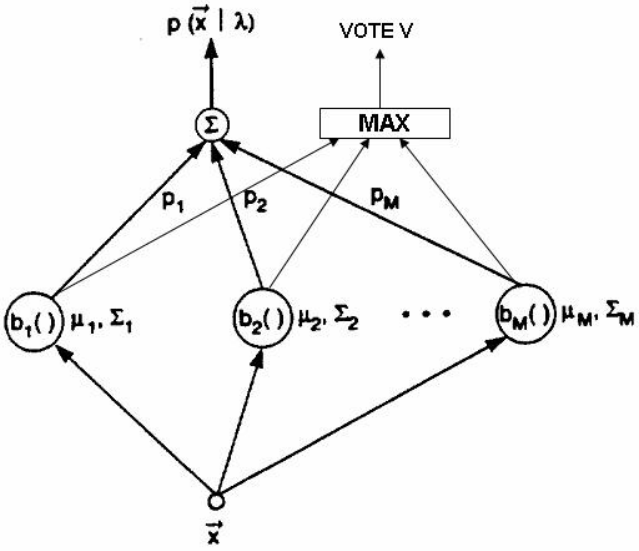

Fig. 2. Recognition on a single frame with Classical GMM and Voting, modified from [4]

the voting scheme, for each frame we find the most likely speaker $\hat{S}$ for that frame by,

$$\hat{S} = \arg \max_{1 \le k \le M} p(\vec{x_i}|\lambda_k) \quad (7)$$

Thus the frames together function as an ensemble classifier. In an ensemble classifier each classifier is run and casts a "vote" as to who the correct speaker is. The votes are then collated and the speaker with the greatest number of votes becomes the final classification. Pseudo code for the algorithm is shown below.

```
10 Initialize a counter for each speaker to 0
20 For each frame j (LOOP 1)
30    For each Speaker i (LOOP 2)
40          Evaluate
                p(x_j|λ_i) = Σ_{k=1}^{M} p_k b_k(x_j)
50    End For (LOOP 2)
60    Find the speaker v with maximum
        probability for the frame j
                v = arg max_{1≤k≤M} p(x_j|λ_k)
70    Increment the counter for speaker v by one
80 End For (LOOP 1)
90 The speaker with the largest counter (i.e. largest
   number of votes) is hypothesized as the
   correct speaker.
```

4. EXPERIMENTAL SETUP AND PARAMETERS

The voting scheme is evaluated similar to [4].The only difference is that the results of the baseline were improved using larger frames than in [4] and the shift between segments is 40 frames.The databases used were KING[7] and SPIDRE[8]. The SPIDRE I and II databases were part of various NIST speaker recognition and tracking evaluations.The length of training was either 30sec or 60sec. In KING there are 10 sessions per speaker. One fourth of the training data vectors were taken from each of the first four files. In SPIDRE there are 4 conversations per speaker. Half the training data vectors were taken from the first conversation and half from the second conversation. All the remaining data vectors were used for testing in both cases. Thus testing involved about 200 sec in SPIDRE and 50 sec in KING split as described below, resulting in atleast 590 and 125 test cases per speaker, so that the results are statistically significant. In the experiments on SPIDRE and KING, 32 gaussians per speaker was found to be optimal, and hence these values was used for all experiments. The test speech was processed by the front end using frames of 30msec length, with 124 frames per second to produce a sequence of MFCC feature vectors $\{x_1, x_2 ... x_t\}$. The sequence of feature vectors was divided into overlapping segments of T feature vectors similar to [4]

$$\overbrace{\vec{x_1}, \vec{x_2} ..... \vec{x_{T-1}}, \vec{x_T}}^{Segment1}, \vec{x_{T+1}} ....$$

$$\vec{x_{40}}, \overbrace{\vec{x_{41}}, \vec{x_{42}} ..... \vec{x_{T+39}}, \vec{x_{T+40}}}^{Segment2}, \vec{x_{T+41}} ....$$

A test segment of 5 sec corresponds to T = 620 feature vectors. Each segment of T feature vectors is treated as a separate test utterance. The error rate is computed as:

% error rate (ER) =

$$\frac{\text{number of incorrectly identified segments}}{\text{total number of segments}} * 100$$

5. EVALUATION OF THE VOTING SCHEME

The voting method was first evaluated on a set of 10 speakers from the KING [7] database and results for a few different testing and training lengths, are summarized in Table.1. It was also evaluated on 44 speakers from SPIDRE I and II databases as shown in Table.2. The method was then evaluated on 48 speakers in the King database as shown in Table.3.

Train (sec)	Test (sec)	Classical GMMs(%ER)	Voting (%ER)	%Improvement with Voting
30	5	2.36	1.94	**17.80**
30	6	2.02	1.02	**49.50**
30	7	1.56	0.56	**64.10**
30	10	7.23	2.27	**68.60**

Table. 1.Error Rate on KING with 10 Speakers

Train (sec)	Test (sec)	Classical GMMs(%ER)	Voting (%ER)	%Improvement with Voting
30	5	13.93	11.37	**18.38**
30	10	11.66	9.14	**10.94**
60	5	4.99	3.72	**25.45**
60	10	3.35	2.128	**34.9**

Table. 2.Error Rate on SPIDRE with 44 Speakers

6. PROS AND CONS OF THE PROPOSED METHOD

The voting scheme is found to work especially well in cases where,

- The signal is subjected to burst noise - a few frames are bad but the rest are reliable such as in VOIP communication. Some packets may be lost during transmission leading to certain frames being unreliable.
- There are only a few speakers
- Very few data vectors are available for testing and training

Thus the method has many applications involving authentication of computer users using voice over a wireless or wired IP network.

Train (sec)	Test (sec)	Classical GMMs(%ER)	Voting (%ER)	%Improvement with Voting
60	5	9.03	14.31	-58.47
60	10	6.97	10.39	-49.06

Table. 3.Error Rate on KING with 48 Speakers

Thus, using only the probabilities of the classical GMM we can extract two types of information

- Probability of the utterance given a model using multiplication
- Probable source of each frame and hence the entire utterance, using voting

One disadvantage of this method is apparent from the results in Table.3. Fig 3. shows the number of votes per speaker in two experiments – one with 11 speakers and the other with 48 speakers – in a series of experiments where the corerct speaker is held constant. It is seen that the method is not as effective when there are a large number of speakers on KING database. When the number of speakers increase, the peak in the voting histogram becomes less prominent and hence speakers are more confusable. Though this would be common in any method it appears to be especially pronounced when voting is used.

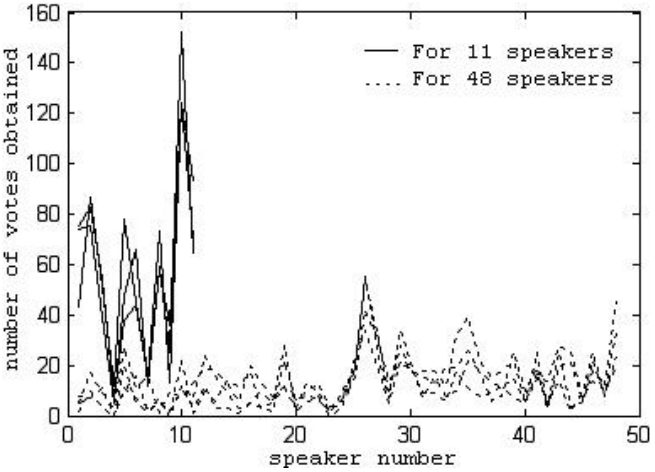

Fig. 3. Effect of increase in number of speakers

Also it was observed that the errors made by the voting scheme and the classical GMM are in some sense orthogonal. Very few

(usually only 1) speaker(s) are common to the N best lists of both systems. This suggests a natural method to overcome the limitations of the voting based identifier.

If voting is to be used in cases where many speakers are enrolled, it is essential that the number of competing speakers is reduced by a first pass and then voting is used in a second pass. At the same time if it is required that the amount of computation should not be increased, the first pass should not require the calculation of an entirely different set of probabilities.

The solution proposed is to perform a first pass using classical GMM, and pick out the top N speakers, that is the N speakers λj which have highest probability $p(\vec{x_i}|\lambda j)$ of generating the frame $\vec{x_i}$. The probabilities of each frame, for each speaker calculated using the speaker's corresponding GMM are stored for later processing. These top N speakers are then compared using the voting mechanism in a second pass. The probabilities of each frame are the same as those calculated in the first pass and hence the stored values may be reused.In this second pass instead of multiplying the probabilities to find the best speaker, which is what was done with classical GMMs, voting is performed as described earlier. Thus the benefits of both the GMM and the voting method may be obtained with negligible increase in computation.

7. EVALUATION OF THE COMBINATION SCHEME

The combination scheme(Combo) was evaluated on the KING database with 10 speakers, with all 48 speakers, and on the SPIDRE I and II databases. Since the KING database was used primarily to compare with the baseline GMM as described in [4].The percentage improvement(%Impr. with Combo.) for different values of N (where N is the the number of best matching speakers remaining after the first pass with classical GMM) is shown in Table.4.

Train (sec)	Test (sec)	GMMs (%ER)	Voting (%ER)	N	Combo (%ER)	%Impr. with Combo.
30	5	13.93	11.37	5	11.82	15.14
30	5			10	11.16	19.88
30	5			20	10.98	**21.17**
30	5			30	11.28	19.02
30	10	11.66	9.14	5	9.78	16.12
30	10			10	9.56	18.01
30	10			20	8.72	**25.20**
30	10			30	9.26	20.58
Train (sec)	Test (sec)	GMMs (%ER)	Voting (%ER)	N	Combo (%ER)	%Impr. with Combo.
60	5	4.99	3.72	5	4.25	14.83
60	5			10	3.33	33.27
60	5			20	3.12	**37.47**
60	5			30	4.00	19.84
Train (sec)	Test (sec)	GMMs (%ER)	Voting (%ER)	N	Combo (%ER)	%Impr. with Combo.
60	10	3.35	2.18	5	3.13	6.23
60	10			10	2.21	34.02
60	10			20	1.82	**45.67**
60	10			30	2.37	29.25

Table. 4.Results on SPIDRE with 44 Speakers

Train (sec)	Test (sec)	GMMs (%ER)	Voting (%ER)	Combo (%ER)	%Impr. with Combo.
60	5	9.03	14.31	8.20	**9.19**
60	10	6.97	10.39	5.22	**25.11**

Table. 5.Results of on KING with 48 Speakers

8. CONCLUSION

The importance of identifying unreliable frames in speaker recognition was motivated. A voting mechanism for combining Gaussian Mixture Probabilities was presented to reduce the effect of few bad frames on identification accuracy. The voting method is seen to provide an improvement over Classical GMMs on almost all datasets evaluated. In only a few cases with a larger number of speakers, on the KING database, it does not perform as well as expected. This motivated the use of a combination method using both voting and Classical GMM testing. This second method is found to always perform better than the Classical GMM while not requiring any extra computation. With optimal choice of N (usually about half the total number of speakers), it is found to always perform better than voting or Classical GMMs.

Also it was observed that the voting scheme and the classical GMM are to some degree orthogonal. Very few (usually only 1) speaker(s) are common to the N best lists of both systems. Thus the voting mechanism is extracting a different kind of information from the Gaussian probabilities than the classical GMM. The combination performs much better than GMMs or voting alone even though all three methods work with the same probabilities.

9. REFERENCES

[1] F.Soong et al., "A vector Quantization Approach to speaker recognition," in *IEEE ICASSP*, 1985, pp. 397–390.

[2] J. Oglesby and J.Mason, "Radial basis function networks for speaker recognition," in *IEEE ICASSP*, May 1991, pp. 393–396.

[3] B.Yegnanarayana and S. Kishore, "AANN: an alternative to GMM for pattern recognition," *Neural Networks*, pp. 459–469, 2002.

[4] D. Reynold and R.C. Rose, "Robust Text Independent Speaker Identification Using Gaussian Mixture Speaker Models," *Proc. IEEE Tran. Speech and Audio Processing*, vol. 3, pp. 72–83, Jan. 1995.

[5] S.B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. on ASSP*, vol. 28, pp. 357–366, 1980.

[6] N. M. Laird 'A. P. Dempster and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, pp. 1–38, 1977.

[7] D Graff J. Godfrey and A.Martin, "Public Databases for Speaker recognition and verification," in *Proc. ESCA Workshop Aut. Spk. Recog.,Ident. and Ver.*, 1994, pp. 39–42.

[8] "SPIDRE: A User's Manual," *http://wave.ldc.upenn.edu/ Catalog/docs/LDC94S15/manual.txt*, 1995.