Improved Text-Independent Speaker Recognition using Gaussian Mixture Probabilities

Balakrishnan Narayanaswamy
B.E. M.S.R.I.T under VTU, 2003

Advisors:
Professor Raj Reddy
Professor Richard Stern

A Report in Candidacy for the Degree of Master of Science (M.S.)
Department of Electrical and Computer Engineering
Carnegie Mellon University
May, 2005

Abstract

Given a speech signal there are two kinds of information that may be extracted from it. On one hand there is the linguistic information about what is being said, and on the other there is also speaker specific information. This report deals with the task of speaker recognition where the goal is to determine which one of a group known speakers best matches the input voice sample. The problem is made harder when the speakers are not constrained to a particular word sequence,when there is only a very small amount of train and test data, or when the train and test data are collected across different channels.

In this work, we concentrate on the task of improving the performance of Gaussian Mixture models for speaker identification, without a substantial increase in computation, by extracting other features from Gaussian Mixture probabilities that are better indicators of who the speaker is. This report looks closely at the outputs of the mixtures when presented with a voice sample and some properties of the distribution of these probabilities are noted. Using this knowledge, frame based voting to improve speaker identification accuracy is suggested. This is shown to decrease the error rate by a factor of 5 or more (from 1.73% to .09% in the best case), with little or no increase in computation. The voting method is also successfully applied to a two-speaker segmentation task and decreases segmentation errors by a factor of 4. A disadvantage of voting is identified and a way to improve it's performance, which combines voting with conventional evaluation of GMMs, or an estimator that uses the median of the Gaussian probabilities, is suggested. These methods result in a decrease in error rate by a factor of 10 or more (from 1.07% to .05% in the best case).

Contents

Chapter 1

Introduction

1.1 What is Speaker Recognition ?

As speech interaction with computers becomes more pervasive in activities such as financial services and information retrieval from speech databases, the utility of automatically recognizing a speaker based solely on vocal characteristics increases. Given a speech sample, speaker recognition is concerned with extracting clues to the identity of the person who was the source of that utterance. Speaker recognition is divided into two specific tasks: verification and identification. In speaker verification the goal is to determine from a voice sample if a person is whom he or she claims. In speaker identification the goal is to determine which one of a group of know voices best matches the input voice sample. In either case the speech can be constrained to a known phrase (text-dependent) or totally unconstrained (text-independent). As telephones become more pervasive, they may become the tool of choice for interacting with computers. The focus of this work is on achieving significantly higher speaker identification rates using short utterances from unconstrained conversational speech and robustness to degradations produced by transmission over a telephone channel.

1.2 Why can Speaker Recognition be Hard ?

There are many algorithms and models that can be used for speaker recognition including Neural Networks[1], unimodal Gaussians, Vector Quantization[2], Radial Basis Functions[3], Hidden Markov Models and Gaussian Mixture Models(GMMs)[4]. These perform well under clean speech conditions, but in many cases performance degrades when test utterances are corrupted by noise, mismatched conditions or if there are only

small amounts of training and testing data. Among these methods GMMs are usually preferred because they offer high classification accuracy while still being robust to corruptions in the speech signal[4].
When speech is corrupted by noise or by the limited bandwidth of telephone lines, speaker recognition accuracy degrades. The feature vectors generated from corrupted speech are no longer similar to the class distributions learned from the training data. Because of the channel effects, there is inherently more variability in the training data, and as a result, the variance of the distributions of the speaker classes increases. This broadening of the class distributions leads to increased classification errors over the case where the training and test speech are both clean. There is also an intrinsic variation in a person's voice which is more pronounced when the voice samples are collected at widely separated times.

1.3 What have people tried before?

This report focuses on robust text-independent speaker recognition of telephone quality conversational speech, with small amounts of testing and training data recorded in different sessions. There are many methods that have been suggested to overcome mismatched conditions and limited data such as Cepstral Mean Normalization(CMN)[5], stochastic feature transformations [6][7], model adaptation [8][9] and score based compensation [10] [11].

1.4 So why do we need more methods?

All the methods listed in the previous section involve additional processing of the features, models, or scores on top of the computation involved in evaluating Gaussian Mixtures. CMN is known to degrade performance when the amount of data available is small. The other adaptive algorithms are also susceptible to this problem. The methods described in this report work only with the probabilities output by conventional evaluation of GMMs. Thus, there is no extra processing involved during training and little or no extra processing during the testing phase. The methods proposed here also suggest different ways of looking at Gaussian probabilities and can be used in addition to other more complex schemes which have previously been shown to be successful. The methods suggested are evaluated on telephone quality speech with different amounts of training and testing data, and they are shown to provide a useful improvement in identification accuracy.

1.5 Overview of report

The rest of this thesis is organized as follows. Chapter 2 gives a more detailed description of the problem, describes the databases used for evaluating the various algorithms. It also discusses the different situations over which the methods are to be evaluated. The standard Gaussian Mixture Model for speaker recognition is described and tested. Chapter 3 describes a voting based classifier to alleviate the effect of outlier frames. Chapter 4 shows how voting can be used to improve performance on a simple two-speaker segmentation task. Chapter 5 shows a simple method to improve the classifier of Chapters 3, by combining it with the conventional Gaussian mixture models, while still not requiring a substantial increase in computation. Chapter 6 describes an alternate view of Gaussian Mixture probabilities as indicators of the correct speaker. Chapter 7 has the conclusions and some directions for future research.

Chapter 2

Gaussian Mixtures Models for Speaker Identification

2.1 Introduction

As described earlier various models have been applied to the task of text independent speaker identification, such as Neural Networks[1], unimodal Gaussians, Vector Quantization[2], Radial Basis Functions[3], Hidden Markov Models and Gaussian Mixture Models(GMMs)[4]. Of these, GMMs have been the most successful, leading to the extensive use of GMM based speaker recognition systems. The use of common corpora for evaluation of speech and speaker recognition systems has proved to be invaluable in comparing different approaches, sharing results, and generally advancing the technology state-of-the-art. In this chapter we first describe the baseline GMM from [4] for speaker identification. We describe the databases used in evaluating the algorithms presented here, and select the parameters of the baseline system that provide the best performance.

2.2 Feature Extraction

There are a number of different speech features that have been shown to be indicative of speaker identity. These include pitch related features, Linear Prediction Cepstral Coefficients (LPCCs)[12] and Maximum AutoCorrelation Value(MACV)[13] features. Although there are no exclusively speaker distinguishing features, the speech spectrum has been shown to be very effective for speaker recognition. Here we use Mel Frequency

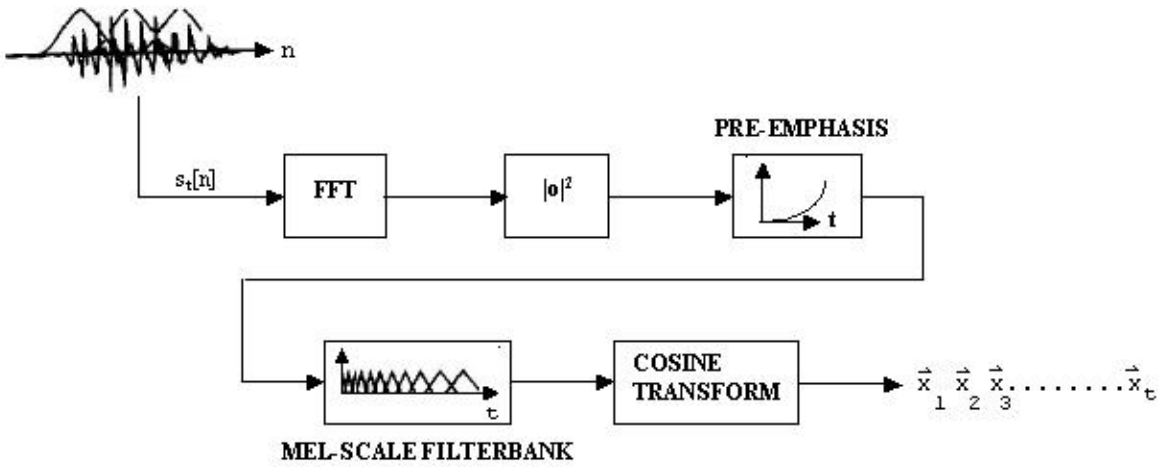

Figure 2.1: Extraction of MFCCs

Cepstral Coefficients[14] (MFCCs) extracted from the spectrum. The main reason for this is that in many applications speaker identification is a precursor to further speech processing, especially speech recognition, to identify what is being said. Among the possible features MFCCs have proved to be the most successful and robust features for speech recognition. So, to limit computation in a possible application, it makes sense to use the same features for speaker recognition. Also, the focus of this report is on extracting additional information from Gaussian Mixture probabilities irrespective of the features used.

Figure 2.1 shows the block diagram of the procedure used for feature extraction in the front end. The speech signal is divided into 30 msec long segments overlapping by 15 msec using a Hamming window. The magnitude spectrum of this short time segment is passed through a simulated mel-scale filter bank consisting of 30 filters. The filter bank is similar to the one described in [14]. The log of the output energy of each filter is calculated and collected into a vector. This is then cosine transformed into cepstral coefficients. The cepstral coefficients are truncated to obtain MFCCs. In all the experiments in this report, 20 MFCC features per frame are used, since they were found to give the best performance in most cases.

2.3 Gaussian Mixture Models for Speaker Recognition

The use of Gaussian Mixture models for modeling speaker identity is motivated by the interpretation that the Gaussian components represent some general speaker dependent spectral shapes and the capability of

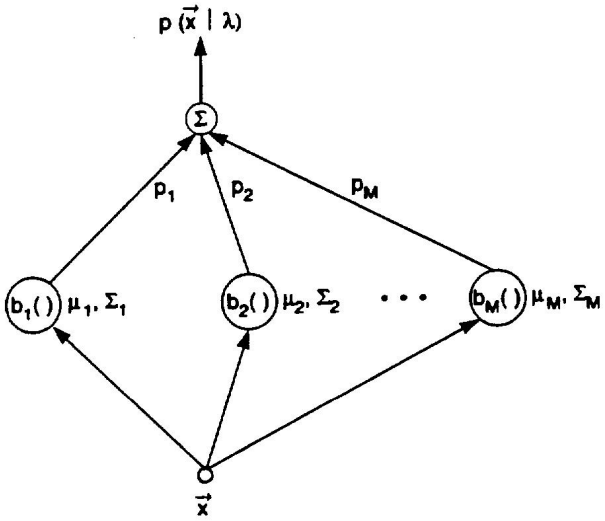

Figure 2.2: GMMs for speaker recognition: from [4]

Gaussian mixtures to model arbitrary densities. A GMM is the weighed sum of M component densities as shown in Figure 2.2, given by the equation,

$$p(\vec{X}|\lambda) = \sum_{i=1}^{M} p_i b_i(\vec{x}) \tag{2.1}$$

Where $\vec{X}$ is a sequence of feature vectors from the audio data, $\vec{x}$ 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 variate Gaussian function of the form,

$$b_i(\vec{x}) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{\frac{1}{2}}} exp\left\{-\frac{1}{2}(\vec{x}-\vec{\mu_i})^{'}\Sigma_i^{-1}(\vec{x}-\vec{\mu_i})\right\} \tag{2.2}$$

with mean vector $\vec{\mu_i}$ and covariance matrix Σ_i. The mixture weights are such that $\sum_{i=1}^{M} p_i = 1$.

For speaker identification, each speaker is represented by a GMM λ_i which is completely parameterized by its mixture weights, means and covariance matrices collectively represented as,

$$\lambda_i = \{p_i, \vec{u_i}, \Sigma_i\} \tag{2.3}$$

For computational ease and improved performance, the covariance matrices are constrained to be diagonal. There are two principal motivations for using GMMs to model speaker identity. The first is that the compo-

nents of such a multi-modal density may represent some underlying set of acoustic classes. It is reasonable to assume that the acoustic space corresponding to a speakers voice can be characterized by a set of acoustic classes representing some broad phonetic events such as vowels, nasals or fricatives. These acoustic classes reflect some general speaker-dependent vocal tract configurations that are useful for characterizing speaker identity. The spectral shape of the i^{th} acoustic class can in turn be represented by the mean μ_i and covariance matrix Σ_i. Because all the training or testing speech is unlabeled, the acoustic classes are hidden in that the class of an observation is unknown.

The second motivation for using Gaussian mixture densities for speaker identification is that a linear combination of Gaussian basis functions is capable of modeling a large class of sample distributions. A GMM can form smooth approximations to arbitrarily shaped densities.

There are several techniques that can be used to estimate the parameters of a GMM, λ, which describes the distribution of the training feature vectors. By far the most popular and well-established is Maximum Likelihood (ML) estimation.

These GMMs are trained separately on each speaker's enrollment data using the Expectation Maximization (EM) algorithm [15]. The update equations that guarantee a monotonic increase in the model's likelihood value are:

Mixture Weights:

$$\overline{p_i} = \frac{1}{T}\sum_{t=1}^{T} p(i|\vec{x_t}, \lambda) \tag{2.4}$$

Means:

$$\overline{\vec{\mu_i}} = \frac{\sum_{t=1}^{T} p(i|\vec{x_t}, \lambda)\vec{x_t}}{\sum_{t=1}^{T} p(i|\vec{x_t}, \lambda)} \tag{2.5}$$

Variances:

$$\overline{\sigma_i}^2 = \frac{\sum_{t=1}^{T} p(i|\vec{x_t}, \lambda)\vec{x_t}^2}{\sum_{t=1}^{T} p(i|\vec{x_t}, \lambda)} - \overline{\vec{\mu_i}}^2 \tag{2.6}$$

where σ_i^2, x_t and μ_i are elements of $\vec{\sigma_i}^2, \vec{x_t}$ and $\vec{\mu_i}$, respectively. The *a posteriori* probability for acoustic class i is given by,

$$p(i|\vec{x_t}, \lambda) = \frac{p_i b_i(\vec{x_t})}{\sum_{k=1}^{M} p_k b_k(\vec{x_t})} \tag{2.7}$$

In speaker identification, given a group of speakers S = $\{1, 2....M\}$, represented by GMMs $\lambda_1, \lambda_2, \lambda_3....\lambda_S$, the objective is to find the speaker model which has the maximum *a posteriori* probability for a given test

sequence,

$$\hat{S} = \arg \max_{1 \leq k \leq M} p(\lambda_k) = \arg \max_{1 \leq k \leq M} \frac{p(\vec{X}|\lambda_k)p(\lambda_k)}{p(\vec{X})} \tag{2.8}$$

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

$$\hat{S} = \arg \max_{1 \leq k \leq M} p(\vec{X}|\lambda_k) = \arg \max_{1 \leq k \leq M} [\prod_{t=1}^{T} (p(\vec{x_t}|\lambda_k)] \tag{2.9}$$

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.

2.4 Databases used for Experimental Evaluation

The use of common corpora for evaluation of speech and speaker recognition systems has proved to be invaluable in comparing different approaches, sharing results, and generally advancing the technology state-of-the-art. Within the last five years the number of publicly available speech corpora has increased dramatically. Two databases were used during the evaluations described in this report, to show that the results are not specific to a certain database or certain voice collection conditions. The databases are KING[16] and SPIDRE[17]. The characteristics that made these databases appropriate are:

- They both have 40-50 speakers. The speaker recognition problem is not easy, and it is similar to the one expected in a typical application.
- They both have speech corrupted by different telephone handsets and channels. They test the ability of a speaker recognition system to generalize to different channel conditions.
- Results on these databases are available for a number of different algorithms allowing easy comparison of the results across systems

KING-92

The KING corpus was collected at ITT in 1987 under a US Government research contract. The version now available from LDC, referred to as KING-92, is based on a 1992 reprocessing of the original recordings. It contains recorded speech from 51 male speakers in two versions, which differ in channel characteristics: one from a telephone handset and one from a high-quality microphone. The speakers are further subdivided into two groups, 25 in one and 26 in the other, who were recorded at different locations. For each speaker and

channel, there are 10 files, corresponding to sessions of about 30 to 60 seconds duration each. Of these 51 speakers, 49 were used for evaluation.

SPIDRE

This corpus is a 2-CD subset of the Switchboard-I collection selected for speaker ID research, and with special attention to telephone instrument variation [18]. Combining the two sides of the conversations also permits speaker change detection or speaker monitoring experiments. There are 45 target speakers; four conversations from each target are included, of which two are from the same handset. Since all conversations are two-sided, this results in 180 target sides. However, for this evaluation, only 44 speakers were used.

2.5 Train and Test data selection

There are 2 test settings that were defined to evaluate different aspects of any speaker recognition algorithm. These are:

Case 1: Training data is recorded on two channels, and test data could be from either of these two channels This setting is to test how well the models can train when data is from different channels. The problem is still not very hard since we have training data on every channel that we have testing data for.

Case 2: Training data is recorded on two channels, and test data is from a completely different channel This setting is hard and most algorithms perform poorly in this test. This is because we do not have any ideas as to the properties of the test channel during training. This tests the ability of the algorithm to generalize to new channels.

In each setting we can have different amounts of data for training and testing. The training data lengths were 5,10,20 sec enrollment for KING 5,10,20 and 30 sec for SPIDRE. All the remaining data, depending on Case 1 or 2, was used for testing. This data was split into segments of length 5,10,20 and 30 sec as described below. Since this covers most of the expected usage conditions, it should give us a good idea of the variation in performance of an algorithm when it has different amounts of training and testing data.

Table 2.1: Error Rate on KING with 49 Speakers for different model orders: Case 1

Train (sec)	Test (sec)	M = 8 (%ER)	M = 16 (%ER)	M = 32 (%ER)
5	5	58.18	56.30	62.02
5	10	55.46	52.66	60.26
5	20	52.82	51.34	59.98
10	5	35.37	29.89	31.13
10	10	32.77	25.17	27.73
10	20	30.41	23.85	24.57
20	5	15.21	12.85	11.20
20	10	11.08	8.64	7.92
20	20	9.32	6.48	3.00

Testing involved at least 50 test utterances per speaker per train-test length combination, so that the results are statistically significant. The sequence of feature vectors was divided into overlapping segments of T feature vectors similar to [4]

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

$$\vec{x_n}, \overbrace{\vec{x_{n+1}}, \vec{x_{n+2}}.....\vec{x_{n+T-1}}, \vec{x_{n+T}}}^{Segment2}, \vec{x_{T+T+1}}....$$

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) =

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

2.6 Setting up the Baseline

Model Order:

The initial base line is as described earlier. The first parameter that needs to be set is the number of Gaussians used to model each speaker. Determining the number of components M in a mixture needed to model a speaker adequately is an important but difficult problem. There is no theoretical way to determine the number of mixture components *a priori*. The results of the experiments with different training and testing lengths, for different model orders is shown in the Tables 2.1 and 2.2.

Table 2.2: Error Rate on SPIDRE with 44 Speakers for different model orders : Case 1

Train (sec)	Test (sec)	M = 8 (%ER)	M = 16 (%ER)	M = 32 (%ER)
5	5	20.17	19.70	30.53
5	10	15.78	16.85	16.43
5	20	11.39	12.37	11.90
5	30	10.36	4.62	8.73
10	5	10.46	7.84	7.80
10	10	7.19	5.88	5.70
10	20	4.20	2.94	2.94
10	30	3.17	1.72	2.43
20	5	8.17	5.60	4.76
20	10	5.88	3.45	2.80
20	20	4.34	2.19	1.54
20	30	2.19	1.70	0.75

When more training data is available, 32 or sometimes only 16 Gaussians are needed. When less data is available, 8 Gaussians gives the best performance. 16 Gaussians were found to give similar or slightly worse performance than 32 Gaussians per speaker in most cases of interest, at half the computational cost. When 64 or more Gaussians per speaker were used, performance was found to fall off. 16 Gaussians are used to model each speaker in all the experiments in this report.

Spectral Shape Compensation:

When the speech signal passes through a linear filter *h[n]* representing the telephone channel, it's magnitude spectrum is multiplied by the magnitude response of the filter. If it is assumed that the magnitude of the spectrum is relatively smooth,it can be shown that the effect of the filter is an additive component on the mel-cepstral feature vector.

$$\vec{z} = \vec{x} + \vec{h} \tag{2.10}$$

where $\vec{z}$ is the observed cepstral vector, $\vec{h}$ is the channel filter cepstral vector, and $\vec{x}$ is the input speech cepstral vector.

The method of mean normalization – Cepstral Mean Normalization (CMN) has been used in many speaker recognition systems. Essentially, it consists of removing the bias component by subtracting the global average vector from each feature vector. The global average vector is

$$\vec{m} = \frac{1}{T}\sum_{t=1}^{T}\vec{z} \tag{2.11}$$

Table 2.3: Comparison of results with and without CMN on SPIDRE for Case 1

Train (sec)	Test (sec)	With CMN (%ER)	Without CMN (%ER)
5	5	41.18	19.70
5	10	36.32	16.85
5	20	36.32	12.37
5	30	63.91	4.62
10	5	16.2	7.84
10	10	10.45	5.88
10	20	7.42	2.94
10	30	6.72	1.72
20	5	7.33	5.60
20	10	4.20	3.45
20	20	2.80	2.19
20	30	2.94	1.07

and the channel compensated vectors are given by,

$$\vec{z}^{comp} = \vec{z}_t - \vec{m} \tag{2.12}$$

However, if the amount of data available per channel is low, then the $\vec{m}$ is not a good estimate of the mean and CMN can do more harm than good. Also, in two of the settings data is available from the test channels and hence cmn is not required, and again using CMN results in an increase in error rate. The comparative results with and without CMN are presented in Table 2.3 which has results for Case 1: Training data taken from first two files and all remaining data from first four files used for testing on SPIDRE.

In most of the settings of importance here, not only does CMN not help, but actually makes things worse. So, all the results in this report here are without CMN.

Chapter 3

Voting for speaker estimation

3.1 Introduction

In this chapter we look at a method to improve speaker identification accuracy using GMMs, while not increasing the computational burden significantly. We look at the output probabilities of the GMMs and hypothesize that the probabilities of outlier frames are an important cause of speaker identification errors. The probabilities of the outlier frames are themselves outliers, which may be caused by the fact that speech from a single speaker is not actually generated as a mixture of Gaussians. We propose frame by frame voting[19] as an alternative to estimate the correct speaker, that normalizes the contribution of each frame, and thus mitigates the effect of outliers.

3.2 Looking at Gaussian Probabilities

Evaluation of GMMs for speaker recognition is done as per equation

$$\hat{S} = \arg \max_{1 \leq k \leq M} p(\vec{X}|\lambda_k) = \arg \max_{1 \leq k \leq M} [\prod_{t=1}^{T} (p(\vec{x_t}|\lambda_k)] \tag{3.1}$$

Taking both sides, to the power $\frac{1}{N}$, we obtain

$$\hat{S} = \arg \max_{1 \leq k \leq M} (p(\vec{X}|\lambda_k))^{\frac{1}{N}} = \arg \max_{1 \leq k \leq M} [\prod_{t=1}^{T} ((p(\vec{x_t}|\lambda_k))]^{\frac{1}{N}} \tag{3.2}$$

Thus, in conventional GMM evaluation, we find the geometric mean of the output probabilities of the

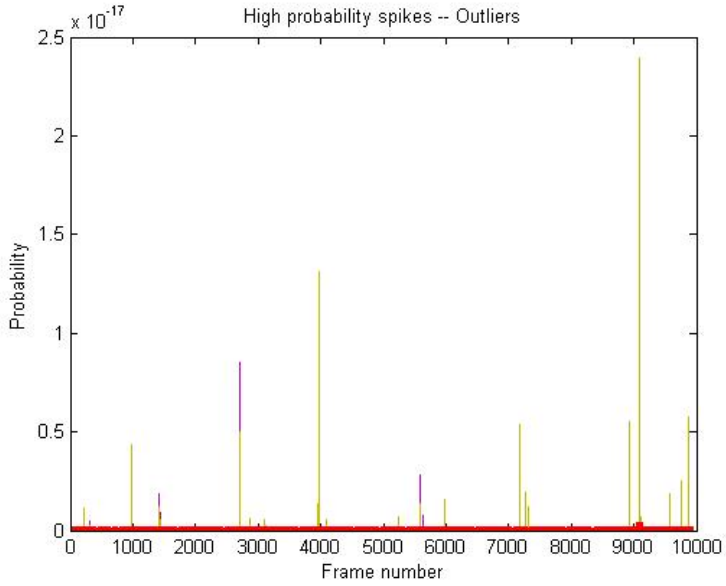

Figure 3.1: Output probabilities of the Gaussian Mixtures for different speakers: Incorrect speaker in Yellow

GMM and use this as an indicator of the correct speaker. To decide what possible alternatives exist to the geometric mean, we explore the properties of the output probabilities.

From the Figures 3.1 and 3.2, we see that the output probabilities are very spiky. The incorrect speaker in yellow in 3.1, is some times many orders of magnitudes higher than the correct speaker in red.
From the Figure 3.3, we see that there are a few frames where all the speakers have very low probabilities (the figure is in log domain). These frames may be noise or non speech frames which match all the models poorly, that is they lie in the tail of most distributions. The fall in probability is often more than -20 in the log domain which corresponds to 10^{-20} in the probability domain. If the probabilities are multiplied, then even a few such outliers could dominate the final result, resulting in identification errors. 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. We would like to somehow limit the effect of these outliers frames. One simple way to do this is frame by frame voting as described below.

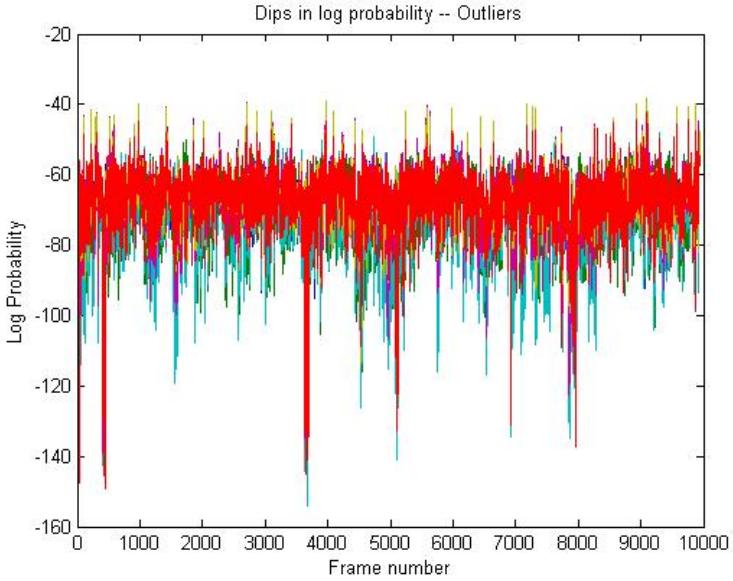

Figure 3.2: Output log probabilities of the Gaussian Mixtures for different speakers: Correct speaker in red

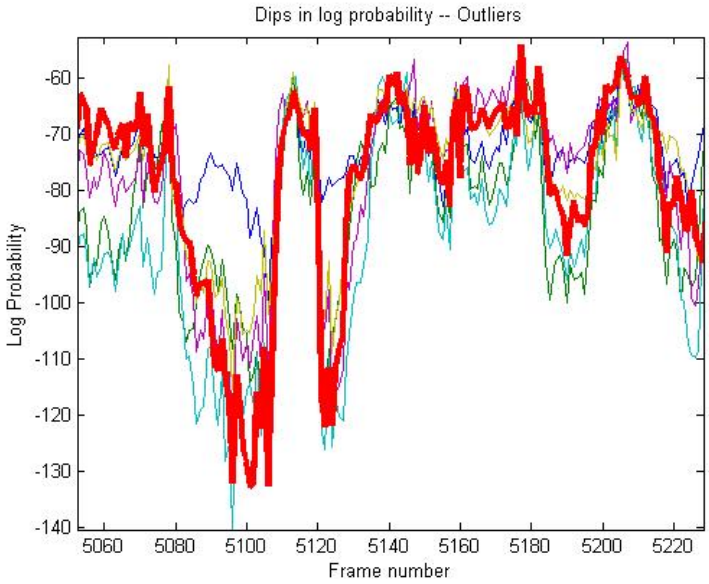

Figure 3.3: Output log probabilities of the Gaussian Mixtures for different speakers: Correct speaker in red

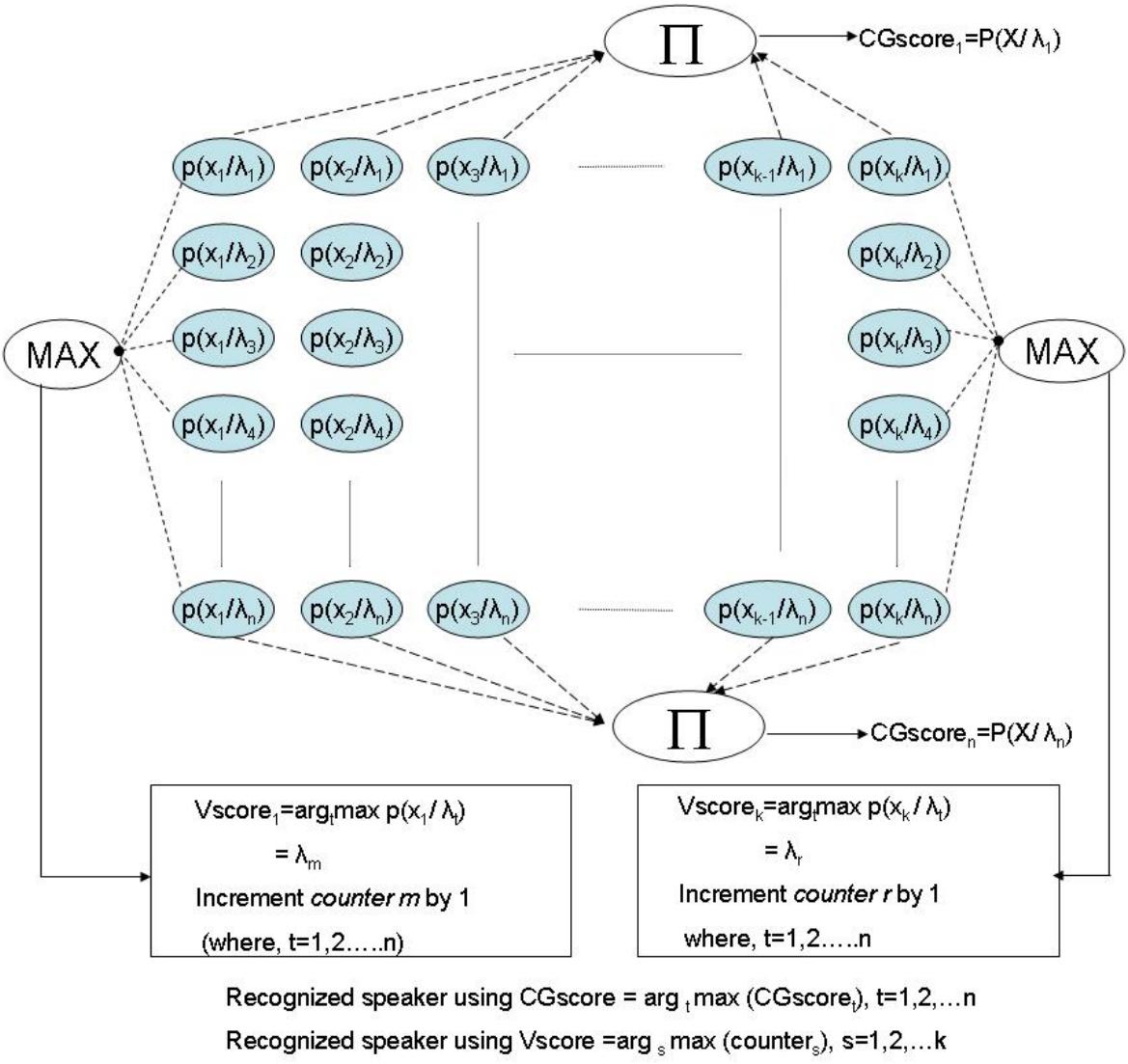

Figure 3.4: Comparison of voting and conventional evaluation of GMMs

3.3 Frames as classifiers

An alternate view of the speaker recognition problem is that each frame is 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. But there are alternative methods to combine the output of multiple classifiers. Since we believe that the errors are being made because of a few outlier frames with small probability values dominating the final results, we are looking for a method that weighs each classifier (frame) equally. In the proposed method, the decisions of all the classifiers (frames) are combined by voting. The difference is shown in Figure 3.4. Thus, in the voting scheme, for each frame we find the most likely speaker $\hat{S}$ for that frame by,

$$\hat{S} = \arg \max_{1 \leq k \leq M} p(\vec{x_i}|\lambda_k) \tag{3.3}$$

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. This is also a good way to prevent a few bad frames from having a unreasonably large effect on the result, as each frame has an equal contribution to the final result. 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
```

$$p(\vec{x_j}|\lambda_i) = \sum_{k=1}^{M} p_k b_k(\vec{x_j})$$

```
50    End For (LOOP 2)
60    Find the speaker v with maximum probability for the frame j
```

$$v = \arg \max_{1 \leq k \leq M} p(\vec{x_j}|\lambda_k)$$

```
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.
```

Table 3.1: Voting on KING Case 1

Train (sec)	Test (sec)	Conv. (%ER)	Voting (%ER)	Improvement (%)
5	5	56.30	44.82	20.40
5	10	52.66	39.18	25.61
5	20	51.34	33.93	33.90
10	5	29.89	24.45	18.21
10	10	25.17	17.09	32.11
10	20	23.85	13.85	41.95
20	5	12.85	13.00	-1.25
20	10	8.64	7.92	8.33
20	20	6.48	5.24	19.14

3.4 Evaluation

The voting method is evaluated in all four settings described before:

- Table 3.1 has results for Case 1: Training data taken from first two files and all remaining data from first two files used for testing on KING.
- Table 3.2 has results for Case 2: Training data taken from first two files and all remaining data from first four files used for testing on KING.
- Table 3.3 has results for Case 1: Training data taken from first two files and all remaining data from first two files used for testing on SPIDRE.
- Table 3.4 has results for Case 2: Training data taken from first two files and all remaining data from first four files used for testing on SPIDRE.

3.5 Conclusions

From the results presented, we can see that frame by frame voting gives a substantial increase in speaker identification accuracy in almost all the cases, even with many different test and train lengths.
The best performance is seen when there is only a small amount of training data. This method seems to overcome deficiencies in the model which arise due to insufficient training data. Models trained with very little data, tend to be sharp and give spikes when faced with outliers. Also, speech from a single speaker may not be well modeled by a mixture of a small number of Gaussians (which is all we can train with finite

Table 3.2: Voting on KING Case 2

Train (sec)	Test (sec)	Conv. (%ER)	Voting (%ER)	Improvement (%)
5	5	77.39	74.35	3.93
5	10	74.27	69.87	5.93
5	20	72.71	66.11	9.08
10	5	58.14	55.66	4.27
10	10	57.18	48.34	15.47
10	20	55.30	41.86	24.31
20	5	36.93	41.21	1.34
20	10	36.93	32.77	11.27
20	20	34.89	27.37	21.56

Table 3.3: Voting on SPIDRE for Case 1

Train (sec)	Test (sec)	Conv. (%ER)	Voting (%ER)	Improvement (%)
5	5	19.7	12.09	38.63
5	10	16.85	7.42	55.96
5	20	12.37	3.73	69.85
5	30	4.62	2.85	38.38
10	5	7.84	5.18	33.93
10	10	5.88	1.91	67.52
10	20	2.94	0.56	80.95
10	30	1.73	0.09	94.8
20	5	5.6	4.06	27.5
20	10	3.45	2.19	36.52
20	20	2.19	0.84	61.64
20	30	1.07	0.56	47.66
30	5	4.62	2.85	38.38
30	10	2.75	1.91	30.55
30	20	2.57	0.84	67.32
30	30	1.07	0.33	69.16

Table 3.4: Voting on SPIDRE for Case 2

Train (sec)	Test (sec)	Conv. (%ER)	Voting (%ER)	Improvement (%)
5	5	64.75	57.38	11.38
5	10	63.12	54.34	13.91
5	20	60.5	50.14	17.12
5	30	73.25	68.53	6.44
10	5	58.45	54.25	7.19
10	10	56.26	49.49	12.03
10	20	54.11	44.35	18.04
10	30	52.94	39.26	25.84
20	5	48.74	46.27	5.07
20	10	44.82	42.11	6.05
20	20	41.69	39.31	5.71
20	30	40.1	36.51	8.95
30	5	47.67	47.67	0
30	10	44.07	44.02	0.11
30	20	41.92	38.75	7.56
30	30	40.1	35.39	11.75

training data), leading to some test data falling in the tails of these distributions. When voting is performed, each frame is normalized to have a total contribution of 1 since each frame is allowed to vote for only one speaker. Hence, outliers do not have the same large effect in a voting scheme. This is one possible reason as to why this model works.

The improvement (while still noticeable) is not as great when the test data is from a different channel. This method does not increase the generalization of the model. This encourages us to try different adaptation methods to get better models, but then use voting to increase the accuracy. As described, voting performs best when there is very little training data(resulting in sharp models) and larger amounts of test data (with higher probability of having an outlier frame in the test utterance).

Chapter 4

Voting for speaker segmentation

4.1 Introduction

This chapter describes an application of the voting algorithm of the previous chapter to a two-speaker segmentation problem. In speaker identification applications, it is often assumed that the speech file contains data of a single speaker. However, in many applications such as identifying participants in a telephone conversation or in a conference, speech from different speakers is intermixed. Under such circumstances, classification of speech according to who is speaking becomes important. To do this, the beginning and end points of each speakers voice are required. The process of locating the end points of each speakers voice in an audio file is called speaker segmentation. In this chapter a novel method for two-speaker segmentation is presented that assumes assumes no prior knowledge of the characteristics of the speakers. It is also shown that frame based voting improves the performance of the segmentation [20].

4.2 Background

A good segmentation algorithm should meet the following requirements:

- It should not be overly sensitive to parameters such as window length, window overlap, and window shift so that they can be selected as a trade off between speed and accuracy.
- It should have the ability to detect the speaker change points accurately.
- It should result in segments with a single speaker.

- It should have optional refinement stages, which allow increased accuracy at the cost of speed.

4.3 Previous Work

Current methods to detect speaker changes are based on decoder based splitting, model based splitting or metric based splitting. Several cluster distances have been tested in [21],[22].
Model based splitting uses models for each speaker, which are trained beforehand and is preferred when prior audio information is available about the speakers. Metric based splitting finds speaker changes based on maxima of some distance measure between two adjacent windows shifted along the speech signal. These segmentation algorithms suffer from a lack of stability since they rely on thresholding of distance values.
The GLR is the most computationally expensive distance measure but produces the best results [23], showing high and narrow peaks at speaker change points. The segmentation algorithm based on Bayesian Information Criterion (BIC), cannot detect two speaker changes closer to one another in time, as BIC has been shown to require longer speech segments [24],[25]. The content based indexing proposed in [26],[27] combines the Generalized Likelihood Ratio(GLR) distance measure to detect the speaker change points and the BIC technique to refine the results in order to fight over-segmentation.

4.4 Computing the Generalized Likelihood Ratio (GLR) distance

Take two segments of speech characterized by a sequence of spectral feature vectors, which we will denote by x_n, $n = 1, ...N_1$ and y_n, $n = 1...N_2$, with a total of $N = N_1 + N_2$ vectors. Assume that the vectors in each of these segments were generated by a multivariate Gaussian distribution, and that the segments are statistically independent. We form the likelihood ratio of the observations with the unknown model parameters replaced by their Maximum Likelihood estimates. L1 is the likelihood that the two segments were generated by different speakers and L2 is the likelihood that the two segments were generated by the same speaker. The ratio of L1 and L2 is called the likelihood ratio λ, and it can be expressed as

$$\lambda = \lambda_{cov} * \lambda_{mean} = \frac{L1}{L2} \tag{4.1}$$

If $a = \frac{N_1}{N}$, S1 and S2 are the sample covariances for each of the two segments and W is their frequency weighted average,

$$W = \frac{N_1}{N} * S_1 + \frac{N_2}{N} * S_2 \tag{4.2}$$

Then we have,

$$\lambda_{cov} = \left(\frac{|S_1|^{\alpha} * |S_2|^{1-\alpha}}{|W|}\right)^{\frac{N}{2}} \tag{4.3}$$

From these we obtain a measure of distance between segments by taking the negative logarithm of the likelihood ratio λ from Equation 4.1.

4.5 Proposed Method

The proposed method consists of two major parts, the initial speaker change detection and the refining of models and change points. In many speech and speaker recognition tasks, models can be trained from a flat start, but in such cases, the models used for segmentation may converge to speech classes (say consonants and vowels) rather than to different speakers. To force the models to converge to models of the speakers, we need spectral features across longer segments, (at least 1-2s in length), to capture the long term speaker information but average out the short term speech information, that is, an initial segmentation that is more likely to contain data of a single speaker. An effective solution to this problem is to use the GLR metric. An algorithm to derive good initial speaker models is described in Section 4.5.3.

4.5.1 Silence Removal

A drawback of training HMMs on unlabelled data with silences is that some of the models may converge to these silence regions. The method used for silence removal in this paper is similar to the second method in the NIST stnr routine [28],a technique suggested by Ned Neuberg, Jordan Cohen and others. To save computation, only the first 5-10s is used for detecting the speech-silence threshold. A signal energy histogram is generated by computing the root mean squared (RMS) power, in decibels, over a 20ms window and then updating the appropriate histogram bin. The window is then shifted by 10 ms and the next power is computed. A plot of these power coefficients is seen to be bi-modal, with a sharp low energy silence mode and a flatter higher energy speech mode. The point of inflection between these two modes is the boundary between speech and silence. However, during the course of experimentation it was found that the threshold obtained at this point was too high and resulted in an increased number of deletion errors. The threshold was chosen to be the peak of the mode corresponding to the noise or silence region, since the method is robust to small silence regions.

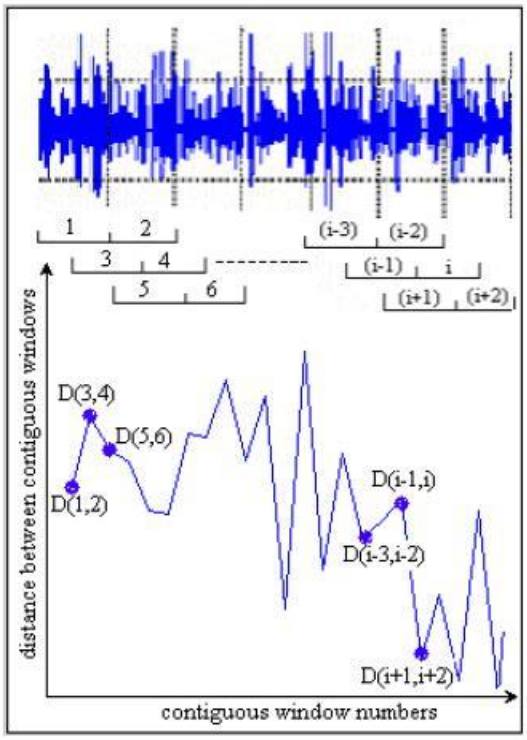

Figure 4.1: Distance computation for the GLR metric

4.5.2 Initial Segmentation

In this step, the dissimilarity between two contiguous windows of the parameterized signal is calculated. The GLR distance is computed for a pair of adjacent windows of the same size, and the windows are then shifted by a fixed step along the whole parameterized speech signal [27] as shown in Figure 4.1. A large distance indicates change in speaker, whereas low values signify that the two portions of the signal correspond to the same speaker.

Let D(i-1,i) denote the GLR distance between the $(i-1)^{th}$ and i^{th} speech sub-segment. The initial segmentation is performed at peaks of this distance measure, which are above a threshold Th_i. The difficulty lies in setting the threshold, Th_i without any prior knowledge. A robust threshold can be set based on the previous N successive distances as follows [29],

$$Th_i = \alpha \frac{1}{N} \sum_{n=1}^{N} D(i-2n-1, i-2n) \tag{4.4}$$

Where, N is the number of previous distances used for predicting the threshold, and α is a coefficient used

as an amplifier and it is set to 1.0. The threshold determined in this way was found to work satisfactorily. Thus, the threshold is set automatically and need not be derived experimentally on each new data set. The above procedure splits the audio data into a series of segments, $SEG_1, SEG_2, ..., SEG_n$ defined by the change points located at the peaks of the GLR distance metric.

4.5.3 Initialization of the GMMs for the two speakers/states

The segment between the starting point of the conversation and the first detected change point is assumed to represent the first speakers data. The feature vectors of this segment are modeled as a GMM, λ_a.
The feature vectors of each segment are modeled using one GMM per segment as $\lambda_2, \lambda_3, ...\lambda_n$.

$$\hat{S} = \arg \min_{2 \leq k \leq n} p(\lambda_k | SEG_1) \tag{4.5}$$

By Bayes rule this becomes,

$$\hat{S} = \arg \min_{2 \leq k \leq n} \frac{p(SEG_1|\lambda_k)p(\lambda_k)}{p(SEG_1)} \tag{4.6}$$

where, $p(\lambda_k) = \frac{1}{n}$ is probability of choosing a particular model, $p(SEG_1)$ remains same for models λ_k, and the set of feature vectors $SEG_1 = se\vec{g}_{11}, se\vec{g}_{12}, ...se\vec{g}_{1T}$ therefore,

$$\hat{S} = \arg \min_{2 \leq k \leq n} p(SEG_1|\lambda_k) \tag{4.7}$$

$$\hat{S} = \arg \min_{2 \leq k \leq n} \sigma_{t=1}^{T} log(p(se\vec{g}_{1t}|\lambda_k)) \tag{4.8}$$

The model, $\hat{S}$ having the minimum a posteriori probability for SEG_1, found using 4.8, is assumed to represent the second speakers data. This step results in the initialization of the parameters of GMMs representing the two states, to the values λ_a and the λ_k corresponding to the segment, $\hat{S}$. In this way, each state is initialized to a segment which is most likely to be from a different speaker.

4.5.4 Refinement of the clusters and change points

The two clusters created in the initialization step are now used as the two reference models (λ_a and λ_b) and 4.9 is computed for all the segments created by the speaker change detection step. The objective here is to find the model which has the maximum aposteriori probability for each segment SEG_y, where, y=2,3..n. If the reference model, λ_a, shows a higher a posteriori probability compared to λ_b for SEG_y, then SEG_y will

be labeled as the first speaker's data, otherwise SEG_y will be labeled as the second speaker's data. The segments are clustered based on the labels.

$$\hat{S} = \arg\max_{y=a,b} p(\lambda_m | SEG_y) \tag{4.9}$$

$\lambda_m = \frac{1}{2}$ (any segment is equally likely to belong to either speaker). This is similar to the Viterbi training algorithm for HMMs with all transition probabilities fixed and equal. A second iteration is performed to obtain clusters of high purity for the two speakers. This procedure can be repeated till convergence. The performance was found not to increase significantly beyond the fourth iteration.

4.6 Evaluation

A good segmentation task should detect the speaker change points accurately. There are two types of errors related to speaker change detection, an insertion error occurs when a speaker change is detected although it does not exist, a deletion error occurs when a true speaker change is not detected. To compare the proposed approach with [26] [27] the same evaluation tasks, on TIMIT and SWITCHBOARD (published by NIST and distributed by the LDC in 1992-3), are used. A set of 40 speakers (T) is taken from the dialect regions DR1 and DR2 from TIMIT. A conversation is obtained by concatenating sentences of 2s on average from two speakers taken from the set, T (clean speech). Two files from SWITCHBOARD, sw2005 and sw2007 are used in a second part of the comparison. Finally results are presented for some other SWITCHBOARD files to show that the proposed method is robust to noise/ silence regions and other variations in the speech signal.

12th order Mel-cepstral coefficients are computed for every frame of 16 ms length with 6.25ms shift for parameter extraction. For speaker change detection, the length of each window is set to 1s and shifted by 0.5s. A four component GMM with diagonal covariance is used to compute the GLR distance between two consecutive windows. For modeling the segments created, eight component GMM with diagonal covariance is used. The method suggested in [26] involves two passes. In the first pass, a distance based segmentation is done using a 2s window shifted by 0.1s. The BIC is then used during the second pass to refine the previously detected change points. The number of deletion errors increases after the second pass, because when BIC is used for segmentation long speech segments are required. Table 4.1 gives results obtained using the segmentation technique given in [26]. The first two and the next two columns correspond to the results

Table 4.1: Comparison of [26] and the Proposed Method on TIMIT and SWITCHBOARD. I - number of Insertions, D - number of Deletions F-female, M-male, CPs - Change Points

Files	GLR,BIC [26]				Prop.	
	$1^{st}Pass$		$1^{st}Pass$		Meth.	
	I	D	I	D	I	D
TIMIT 29 CPs	26	3	9	7	2	2
TIMIT 27 CPs	23	3	9	7	3	2
sw2005(M-M) 19 CPs	41	6	17	7	0	2
sw2008(F-F)30 CPs	31	17	18	17	4	3

obtained after the first and second pass respectively. The last two columns show the results obtained using the method proposed.

One warning is that this is not a completely fair comparison since BIC is looking for the number of clusters, but that information is already known in this case. It is just used as a baseline to compare performance.

The resolution of the proposed method is t=0.5s, and therefore, if any segment boundary is hypothesized within the time interval, $t_0-\delta t < t < t_0+\delta t$ of the reference boundary t_0, it is regarded as correct. Utterances of less than 0.5s duration were not taken into account while marking the correct change points. As can be seen from Table 4.1, the proposed method decreases number of errors on SWITCHBOARD by 84.75%.

4.7 Voting for speaker segmentation

During the refinement stage of the Sub-section 4.5.4, the two clusters created in the initialization step are now used as the two reference models (λ_a and λ_b) and 4.9 is computed for all the segments created by the speaker change detection step, to find the model which has the maximum a posteriori probability for each segment SEG_y, where, y=2,3..n. Instead of this, we can use a frame based voting on each segment, similar to Chapter 3. We then assign a label to each segment depending on which cluster gets the most number of votes. The segments are clustered based on the labels, and the reference models λ_a and λ_b can be retrained using the clustered segments. This procedure can then be repeated iteratively as before.

4.8 Evaluation of voting for speaker segmentation

The system is evaluated on the a series of SWITCHBOARD files and performance of the methods of Section 4.5 and 4.7 are compared. The algorithm for segmentation improves over the baseline GLR-BIC method. It

Table 4.2: Comparison of conventional GMM evaluation and voting on SWITCHBOARD. I - number of Insertions, D - number of Deletions F-female, M-male, CPs - Change Points

Files	Conv.. GMM		Voting Method		Characteristics
	I	D	I	D	
sw2001(F-F) 34 CPs	3	7	3	3	some background noise
sw2006(F-M) 25 CPs	7	4	2	2	large silences, noisy
sw2010(F-M) 45 CPs	9	4	2	4	some background noise
sw2015(F-M) 24 CPs	2	2	0	3	some background noise
sw2017(F-M) 37 CPs	14	3	6	1	some background noise
sw2018(F-F) 65 CPs	14	5	8	2	overlapped speech, background noise
sw2102(F-M) 52 CPs	7	4	2	2	noisy, few silences
sw2105 (F-M) 52 CPs	3	6	2	2	noisy, large silences

is seen that voting based segmentation results in a substantial improvement in the segmentation accuracy over this new algorithm

4.9 Conclusions

In this chapter a novel method for two speaker segmentation [20] was described. A modification based on the voting scheme of Chapter 3 was evaluated and shown to provide an appreciable improvement in accuracy.

Chapter 5

Combining voting and conventional GMMs

5.1 Introduction

In Chapter 3, a frame by frame voting based classifier was proposed and evaluated. This was found to give substantial improvement in many cases. In this chapter we look in more detail at the performance of the voting based classifier. We note in some vary rare cases voting performs worse than conventional GMM evaluation. We hypothesize that this is because when there are many possible speakers and limited test data, the finite number of votes(=number of test frames) results in no single speaker as a clear cut winner. We suggest a combination scheme that further improves performance of the voting based classifier, by decreasing the number of competing speaker in a first pass using conventional GMM evaluation[19].

5.2 Effect of number of speaker on the voting classifier

One disadvantage of this method is apparent from Figure 5.1. Figure 5.1 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 correct 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.

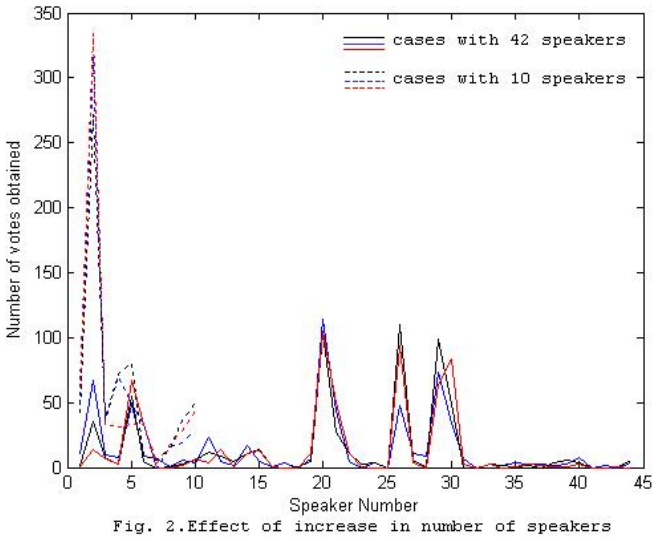

Fig. 2.Effect of increase in number of speakers

Figure 5.1: Effect of increase in number of speakers in the voting classifier

Another way to think of this is that the total number of votes is fixed for a given length of the test utterance, and votes can only be assigned in a discrete manner. On the other hand, when probabilities are used, each frame can contribute any amount (between 0 and 1) to any or all speakers.

5.3 What can we do?

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

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 the second pass. At the same time, if it is required that the amount of computation should not be increased, the second pass should not require the calculation of an entirely different set of probabilities.

Table 5.1: Combination I on KING Case 1

Train (sec)	Test (sec)	Conv. (%ER)	Voting N=15 (%ER)	Impr. (%)	Voting N=20 (%ER)	Impr. (%)	Voting N=30 (%ER)	Impr. (%)
5	5	56.30	44.46	21.04	44.26	21.39	44.42	21.11
5	10	52.66	39.58	24.85	40.10	23.86	40.06	23.94
5	20	51.34	35.37	31.10	35.65	30.55	37.17	27.59
10	5	30.85	20.21	34.50	21.89	29.05	24.17	21.66
10	10	27.37	13.93	49.12	15.41	43.71	17.01	37.87
10	20	25.65	10.72	58.19	11.60	54.76	13.77	46.33
20	5	13.37	8.28	38.02	7.40	44.61	9.20	31.14
20	10	9.40	4.52	51.91	3.40	63.83	4.76	49.36
20	20	7.60	2.96	61.05	1.24	83.68	2.16	71.58

The solution proposed is to perform a first pass using classical GMM and pick out the top N speakers, that is the N speakers λ_j which have highest probability $p(\vec{X}|\lambda_j)$ of generating the utterance $\vec{X}$. The probabilities of each frame, for each speaker calculated using the speakers 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 in Chapter 3. Thus, the benefits of both the classical GMM and the voting method may be obtained with negligible increase in computation.

5.4 Evaluation

The combination scheme(Combo1) was evaluated on the KING and SPIDRE databases. The percentage improvement 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 the Tables 5.1, 5.2, 5.3, 5.3.

5.5 Conclusions

We see that in almost all cases, Combination method I with N=15 for KING and N=20 for SPIDRE, performs better than either voting or conventional evaluation of GMMs. Since trivially N=1 is equivalent to conventional GMM evaluation and N = Total Number of Speakers is equivalent to just the voting algorithm, we can always find an N that is as good as the better one of these two.

Table 5.2: Combination I on KING Case 2

Train (sec)	Test (sec)	Conv. (%ER)	Voting N=15 (%ER)	Impr. (%)	Voting N=20 (%ER)	Impr. (%)	Voting N=30 (%ER)	Impr. (%)
5	5	77.39	72.79	5.95	72.63	6.15	74.79	3.36
5	10	74.27	67.27	9.43	67.47	9.16	69.87	5.93
5	20	72.71	63.91	12.11	65.63	9.74	67.39	7.32
10	5	58.14	50.38	13.35	51.62	11.22	53.54	7.91
10	10	55.30	44.06	20.33	44.22	20.04	46.58	15.77
10	20	53.98	38.54	28.61	37.62	30.32	40.78	24.46
20	5	42.78	34.53	19.27	34.45	19.46	37.05	13.38
20	10	37.09	26.09	29.67	26.37	28.91	28.65	22.76
20	20	33.65	20.49	39.12	21.53	36.03	23.09	31.39

Table 5.3: Combination I on SPIDRE for Case 1

Train (sec)	Test (sec)	Conv. (%ER)	Voting N=15 (%ER)	Impr. (%)	Voting N=20 (%ER)	Impr. (%)	Voting N=30 (%ER)	Impr. (%)
5	5	19.70	11.58	41.23	11.53	41.47	11.62	41.00
5	10	16.85	7.42	55.96	7.47	55.68	7.70	54.29
5	20	12.37	3.03	75.47	3.55	71.32	3.59	70.94
5	30	9.71	0.79	91.83	1.40	97.12	1.87	80.77
10	5	7.84	3.92	50.00	4.76	39.29	4.81	38.69
10	10	5.88	1.82	69.05	1.96	66.67	1.87	68.25
10	20	2.94	0.61	79.37	0.70	76.19	0.51	82.54
10	30	1.73	0.09	94.59	0.09	94.59	0.14	91.89
20	5	5.60	3.41	39.17	3.31	40.83	3.22	42.50
20	10	3.45	1.77	48.65	1.54	55.41	2.05	40.54
20	20	2.19	1.03	53.19	0.56	74.47	1.03	53.19
20	30	1.07	0.61	43.48	0.05	95.65	0.84	21.74
30	5	4.30	3.22	25.11	2.99	30.47	3.13	27.21
30	10	2.75	1.73	37.29	1.87	32.20	1.59	42.37
30	20	2.57	0.98	61.82	1.63	36.36	0.75	70.91
30	30	1.07	0.42	60.87	0.70	34.78	0.28	73.91

Table 5.4: Combination I on SPIDRE for Case 2

Train (sec)	Test (sec)	Conv. (%ER)	Voting N=15 (%ER)	Impr. (%)	Voting N=20 (%ER)	Impr. (%)	Voting N=30 (%ER)	Impr. (%)
5	5	64.75	57.05	11.90	56.30	13.05	56.77	12.33
5	10	63.12	53.36	15.46	52.71	16.49	53.78	14.79
5	20	60.50	49.16	18.75	48.93	19.14	49.81	17.67
5	30	73.25	69.70	4.84	43.46	40.66	46.08	37.09
10	5	58.45	51.87	11.26	51.59	11.74	53.31	8.79
10	10	56.26	47.85	14.94	47.67	15.27	49.49	12.03
10	20	54.11	42.62	21.23	42.95	20.62	43.84	18.98
10	30	52.94	36.41	31.22	40.48	23.54	38.56	27.16
20	5	48.74	45.42	6.80	47.48	2.59	47.43	2.68
20	10	44.82	41.55	7.29	43.28	3.44	41.64	7.08
20	20	41.69	37.39	10.30	38.89	6.72	38.38	7.95
20	30	40.10	34.69	13.50	34.87	13.04	35.62	11.17
30	5	47.67	46.45	2.55	46.50	2.45	45.33	4.90
30	10	44.07	43.00	2.44	43.28	1.80	41.78	5.19
30	20	41.92	36.55	12.81	38.89	7.24	37.91	9.58
30	30	40.10	33.80	15.72	34.50	13.97	35.06	12.57

In most of the cases tested however, N as above was found to give either the best or close to the best performance. This is most likely a function of the number of speakers and especially the type and amount of test data used, and needs to be verified on a larger database with more speakers and more data for each speaker.

Chapter 6

Median for speaker estimation

6.1 Introduction

In this chapter we continue the thought process of Chapter 3, which looked at speaker identification as the problem of estimating the correct probability(using voting) from the probabilities of the GMMs when presented with a test utterance. Since we are looking for a robust estimator invariant to outliers we try the median. However, the median does not improve performance, perhaps because the data is not actually very noisy – there are very few outlier frames and most frames are reliable. We then combine the median classifier with the voting method similar to Chapter 5 and obtain very good performance.

6.2 Looking at Gaussian Probabilities again

Evaluation of GMMs for speaker recognition is done as per equation 6.1.

$$\hat{S} = \arg \max_{1 \leq k \leq M} (p(\vec{X}|\lambda_k))^{\frac{1}{N}} = \arg \max_{1 \leq k \leq M} [\prod_{t=1}^{T} ((p(\vec{x_t}|\lambda_k))]^{\frac{1}{N}} \tag{6.1}$$

Thus, we find the geometric mean of the output probabilities of the GMM and use this as an indicator of the correct speaker.

To decide what possible alternatives exist to the geometric mean, we explore the properties of the output probabilities of Gaussian Mixtures. Looking at the Figures 3.3 and 3.3,we can see that though the correct speaker's probabilities are higher in most frames, there are some frames where it is lower than all the speakers and some frames where some other speaker has much higher probability than the correct one. We therefore

need an estimate of central tendency that is not sensitive to outliers.

The problem with the geometric mean is that it depends on each value in the series. Changing any value changes the geometric mean at least a little. A measure of central tendency that is less sensitive to outliers, may perform better.

If confronted with severely corrupted data, the behavior of an estimator can be described by its breakdown point β. This is the smallest fraction of outliers, i.e. of data not obeying the assumed noise model, which can cause the estimator to produce arbitrarily bad results.

As a simple example, consider n measurements $x_i = s + \eta_i$ of a signal s corrupted by additive noise η_i. We further assume the noise to be Gaussian noise, i.e., $P(\eta) \sim \exp(-\eta^2/2\sigma^2)$. The maximum likelihood estimate s^* of the signal is than given by a least-square fit, which in our example yields the mean of the measurements, $s^* = 1/n, \sum_i x_i$, as estimation formula.

However, the widely used assumption of signals corrupted by additive Gaussian noise is slightly questionable. Since extremely low probability values are assigned to large values of noise, these values distort the estimate if they occur. A single large deviation in one the measurements, say x_k, will cause the mean s^* to deviate arbitrarily far from the true value. It can be derived that the estimator has a breakdown point of $\beta = \frac{1}{n}$, and asymptotically as $n \leftarrow \infty$ a breakdown point of 0. This is a property common to all least square based estimators [30].

There exist other classes of estimators, called robust estimators, which can tolerate a non-zero percentage of outliers. They are typically non-linear estimation schemes. A classical example of a robust estimator is the median of n data points, which is insensitive to a few large outliers in its set of measurements. In fact, this estimator has a breakdown point of 0.5, i.e., as much as 50% of the data can be corrupted before this estimator fails.

The advantages of the median as an estimator are:

- The median is highly resistant to the effect of outliers.
- The median does not depend on the exact value of all numbers in the series, just their relative positions. Thus, many values can be changed or corrupted without affecting the median value.
- As shown before, the correct speaker is

$$\hat{S} = \arg \max_{1 \leq k \leq M} p(\vec{X}|\lambda_k) = \arg \max_{1 \leq k \leq M} [\prod_{t=1}^{T} (p(\vec{x_t}|\lambda_k)] \tag{6.2}$$

Taking log to base e on both sides, we obtain

$$\hat{S} = \arg \max_{1 \leq k \leq M} ln(p(\vec{X}|\lambda_k)) = \arg \max_{1 \leq k \leq M} [\sum_{t=1}^{T} (ln(p(\vec{x_t}|\lambda_k))] \tag{6.3}$$

Dividing both sides by N,

$$\hat{S} = \arg \max_{1 \leq k \leq M} \frac{1}{N} ln(p(\vec{X}|\lambda_k)) = \arg \max_{1 \leq k \leq M} [\frac{1}{N} \sum_{t=1}^{T} (ln(p(\vec{x_t}|\lambda_k))] \tag{6.4}$$

Thus, we are calculating the average of the log probabilities. The median of log probabilities is just log of the median of the probabilities, since log is a monotonic function which retains ordering of the terms. The median is a good estimate of mean in noise, so if the probabilities are noisy then median is a better estimator than the mean.

We can use the median of the probabilities as an indicator of the right speaker. We estimate the true speaker as:

$$\hat{S} = \arg \max_{1 \leq k \leq M} median(p(\vec{x_t}|\lambda_k)) \tag{6.5}$$

6.3 Evaluation of Median as an estimator

In spite of the motivation presented above, the median as an estimator does not perform significantly better than conventional evaluation of GMMs. This maybe because the data is not noisy enough that the median would be significantly better. However, It was observed that the errors made by the voting scheme described in Chapter 3, and the voting classifier are quite different. So, maybe similar to the combination scheme described in Chapter 5, we can perform a first pass using the median estimator, and pick out the top N speakers, that is the N speakers λ_j 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 speakers 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 finding the median of the probabilities to find the best speaker, voting is performed as described in Chapter 3. The benefits of both the median and the voting method may be obtained with negligible increase in computation.

Table 6.1: Combination II on KING Case 1

Train (sec)	Test (sec)	Conv. (%ER)	Median N=15 (%ER)	Impr. (%)	Median N=20 (%ER)	Impr. (%)	Median N=30 (%ER)	Impr. (%)
5	5	56.30	44.58	20.82	43.62	22.53	44.38	21.18
5	10	52.66	39.38	25.23	38.98	25.99	40.62	22.87
5	20	51.34	34.65	32.50	34.89	32.03	37.01	27.90
10	5	30.85	19.93	35.41	21.61	29.96	23.97	22.31
10	10	27.37	14.09	48.54	15.09	44.88	16.93	38.16
10	20	25.65	11.16	56.47	11.48	55.23	13.37	47.89
20	5	13.37	8.08	39.52	7.32	45.21	9.12	31.74
20	10	9.40	4.48	52.34	3.48	62.98	4.80	48.94
20	20	7.60	2.96	61.05	1.20	84.21	2.12	72.11

Table 6.2: Combination II on KING Case 2

Train (sec)	Test (sec)	Conv. (%ER)	Median N=15 (%ER)	Impr. (%)	Median N=20 (%ER)	Impr. (%)	Median N=30 (%ER)	Impr. (%)
5	5	77.39	72.75	6.00	73.19	5.43	74.79	3.36
5	10	74.27	67.67	8.89	67.59	9.00	70.15	5.55
5	20	72.71	64.27	11.61	65.43	10.02	67.51	7.15
10	5	58.14	49.98	14.04	51.18	11.98	53.38	8.19
10	10	55.30	44.14	20.19	43.50	21.35	46.42	16.06
10	20	53.98	37.74	30.10	36.53	32.32	40.38	25.20
20	5	42.78	34.29	19.83	34.05	20.39	36.57	14.50
20	10	37.09	26.33	29.02	26.33	29.02	28.65	22.76
20	20	33.65	21.29	36.74	20.85	38.05	23.05	31.51

6.4 Evaluation of Combination method 2

The combination scheme II(Combo2) was evaluated on the KING and SPIDRE databases. The percentage improvement 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 the tables below.

6.5 Conclusions

We see that in almost all cases, Combination method II with N=20 performs much better than conventional evaluation of GMMs. Since trivially N = Total Number of Speakers is equivalent to just the voting algorithm, we can always find an N that is as good as the voting algorithm. Thus, the median estimator in combination with the voting estimator gives performance as good as or slightly better than the Combination I described in Chapter 5.

Table 6.3: Combination II on SPIDRE for Case 1

Train (sec)	Test (sec)	Conv. (%ER)	Median N=15 (%ER)	Impr. (%)	Median N=20 (%ER)	Impr. (%)	Median N=30 (%ER)	Impr. (%)
5	5	19.70	11.44	41.94	11.58	41.23	11.58	41.23
5	10	16.85	7.33	56.51	7.38	56.23	7.70	54.29
5	20	12.37	3.13	74.72	3.31	73.21	3.59	70.94
5	30	9.71	1.26	87.02	1.49	96.93	1.49	84.65
10	5	7.84	3.83	51.19	4.62	41.07	4.86	38.10
10	10	5.88	1.73	70.63	1.91	67.46	1.87	68.25
10	20	2.94	0.61	79.37	0.70	76.19	0.51	82.54
10	30	1.73	0.00	100.00	0.09	94.59	0.14	91.89
20	5	5.60	3.31	40.83	3.27	41.67	3.17	43.33
20	10	3.45	1.82	47.30	1.54	55.41	2.05	40.54
20	20	2.19	0.98	55.32	0.61	72.34	0.98	55.32
20	30	1.07	0.61	43.48	0.05	95.65	0.84	21.50
30	5	4.62	3.17	31.39	2.94	36.36	3.08	33.33
30	10	2.75	1.77	35.59	1.82	33.90	1.63	40.68
30	20	2.57	0.93	63.64	1.59	38.18	0.79	69.09
30	30	1.07	0.37	65.22	0.75	30.43	0.28	73.91

Table 6.4: Combination I on SPIDRE for Case 2

Train (sec)	Test (sec)	Conv. (%ER)	Median N=15 (%ER)	Impr. (%)	Median N=20 (%ER)	Impr. (%)	Median N=30 (%ER)	Impr. (%)
5	5	64.75	57.00	11.97	55.98	13.55	56.72	12.40
5	10	63.12	53.31	15.53	52.85	16.27	53.83	14.72
5	20	60.50	49.30	18.52	48.88	19.21	49.91	17.52
5	30	73.25	VALUE	VALUE	43.23	40.98	45.33	38.12
10	5	58.45	51.82	11.34	51.87	11.26	53.17	9.03
10	10	56.26	47.57	15.44	47.67	15.27	49.35	12.28
10	20	54.11	42.30	21.83	42.72	21.05	44.07	18.55
10	30	52.94	36.27	31.48	40.34	23.81	38.61	27.07
20	5	48.74	45.38	6.90	47.48	2.59	47.39	2.78
20	10	44.82	41.55	7.29	43.09	3.85	41.69	6.98
20	20	41.69	37.21	10.75	39.26	5.82	38.42	7.84
20	30	40.10	34.50	13.97	34.83	13.15	34.73	13.39
30	5	47.67	46.22	3.04	46.50	2.45	45.33	4.90
30	10	44.07	43.00	2.44	43.09	2.22	41.88	4.98
30	20	41.92	36.65	12.58	39.03	6.90	37.86	9.69
30	30	40.10	33.80	15.72	34.55	13.85	35.01	12.69

Chapter 7

Conclusions and Future work

7.1 Conclusions

- Frame by frame voting gives a substantial improvement in speaker recognition in almost all cases. The decrease in error rate is up to 94.8%(a factor of 19 improvement) and 25.72%(a factor of 1.3 improvement) on average across all the test sets.
- The voting scheme for speaker identification can be used to provide a substantial improvement in the task of two-speaker segmentation.
- If conventional evaluation of GMMs is used to reduce the number of speakers, before applying voting, a further decrease in error rate is obtained in all cases. The decrease in error rate is by as much as a factor of 18 and a factor of 1.5 on average.
  - For N=15 up to 94.59% and 34.75% on average
  - For N=20 up to 94.59%and 33.56% on average.
  - For N=30 up to 91.89% and 32.60% on average.
- Using the median instead of mean of log likelihoods does not improve the accuracy
- Using the median followed by a frame by frame voting, performs the best among all the methods evaluated providing a decrease in error rate by as much as a factor of 20 or a factor of 1.5 on average.
  - For N=15 up to 100.00% and 33.67% on average.

  - For N=20 up to 94.59% and 32.45%on average.
  - For N=30 up to 91.89%and 31.08% on average.

7.2 Future Work

- Training the GMMs using the median may provide improvement when the median is used in the evaluation stage.
- Robust estimators other than the median may be useful in speaker identification
- Detecting outlier frames more systematically may provide an improvement in accuracy.

Bibliography

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

[2] F.Soong et al. A vector Quantization Approach to speaker recognition. In *IEEE ICASSP*, pages 397–390, 1985.

[3] J. Oglesby and J.Mason. Radial basis function networks for speaker recognition. In *IEEE ICASSP*, pages 393–396, May 1991.

[4] D. Reynold and R.C. Rose. Robust text independent speaker identification using gaussian mixture speaker models. *Proc. IEEE Tran. Speech and Audio Processing*, 3:72–83, January 1995.

[5] Atal B. Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. 55:1304–1312, June 1974.

[6] C. H. Lee A. C. Surendran and M. Rahim. Nonlinear compensation for stochastic matching. *Proc. IEEE Tran. Speech and Audio Processing*, 7:643–655, 1999.

[7] M. W. Mak and S. Y. Kung. Combining stochastic feautre transformation and handset identification for telephone-based speaker verification. In *Proc. of IEEE, ICASSP*, pages 1701–1704, 2002.

[8] F. Beaufays and M. Weintraub. Model transformation for robust speaker recognition from telephone data. In *Proc. of IEEE, ICASSP*, volume 2, pages 1063–1066, 1997.

[9] M. W. Mak K. K. Yiu and S. Y. Kung. Environment adaptation for robust speaker verification. In *Proc. of Eurospeech*, pages 2973–2976, 2003.

[10] D. A. Reynolds. Comparison of background normalization methods for text independent speaker verification. In *Proc. of Eurospeech*, pages 963–966, 1997.

[11] M. Carey R. Auckenthaler and H. Lloyd-Thomas. Score normalization for text-independent speaker verification systems. *Digital Signal Processing*, 10:42–54, 2000.

[12] J. Makhoul. Linear predication, a tutorial review. *Proc. IEEE*, 6:561–580, 1975.

[13] B.Wildermoth and K. K. Paliwal. Use of voicing and pitch information for speaker recognition. In *Use of Voicing and Pitch Information for Speaker Recognition*, pages 324–328, 2000.

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

[15] N. M. Laird A. P. Dempster and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38, 1980.

[16] Dr. Alan Higgins and Dave Vermilyea. King speaker verification, 1992. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC95S22.

[17] Ed Holliman Alvin Martin, Jack Godfrey and Mark Przybocki. Spidre, 1996. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC94S15.

[18] Reynolds. D. The effects of handset variability on speaker recognition performance: Experiment on the switchboard corpus. In *Proc. of IEEE, ICASSP*, pages 113–116, May 1996.

[19] B. Narayanaswamy and Rashmi Gangadharaiah. Extracting additional information from gaussian mixture probabilities for improved text-independent speaker identification. In *Proc. of IEEE, ICASSP*, March 2005.

[20] N.Balakrishnan Rashmi Gangadharaiah, B. Narayanaswamy. A novel method for two-speaker segmentation. In *Proc. of ICSLP*, volume 3, pages 2337–2340, September 2004.

[21] Laurent Couvreur and Jean-Marc Boite. Speaker tracking in broadcast audio material in the frame work of the thisl project. In *Proc. of European Speech Communication Association ETRW Workshop on Accessing Information in Spoken Audio, Cambridge (UK)*, pages 84–89, 1999.

[22] Shrikanth Narayanan Soonil Kwon. Speaker change detection using a new weighted distance measure. In *IEEE International Conference on Spoken Language Processing, Denver, USA*, volume 4, pages 2537–2540, 2002.

[23] M. Siu H. Gish and R. Rohlicek. Segregation of speakers for speech recognition and speaker identification. In *Proc. of IEEE ICASSP*, pages 873–876, 1991.

[24] S.Chen and P.Gopalakrishnan. Speaker, environment and channel change detection and clustering via the bayesian information criterion. In *DARPA speech recognition workshop, Lansdowne, Virginia*, 1998.

[25] Alain Tritschler and Ramesh Gopinath. Improved speaker segmentation and segments clustering using bayesian information criterion. In *Sixth European Conference on Speech Communication and Technology, Budapest, Hungary*, pages 679–682, 1999.

[26] David Kryze Perrine Delacourt and Christian J. Wellekens. Speaker-based segmentation for audio data indexing. In *SCA International Speech Communication Association, Cambridge, UK*, pages 78–83, 1999.

[27] Perrine Delacourt and Christian J.Wellekens. Audio data indexing: use of second order statistics for speaker-based segmentation. In *International Conference on Multimedia Computing and Systems, Florence, Italy*, volume 2, pages 959–963, 1999.

[28] Casimir Wierzynski and Jon Fiscus. 'stnr.doc' included with the nist speech quality assurance (spqa) package version 2.3 and speech file manipulation software (sphere) package version 2.5.

[29] Hong-Jiang Zhang Lie Lu and Hao Jiang. Content analysis for audio classification and segmentation. *IEEE transactions on speech and audio processing*, 10(7):504–516, 2002.

[30] P. Meer. Robust high breakdown estimation and consensus. In *Advances in Machine Vision: Strategies and Applications, C. Archibald, E. Petriu (Eds.), World Scientific Publishing Company, Singapore*, pages 23–34, 1992.