Initialization of Speaker Segmentation System Using Friends and Enemies Algorithm

S I P C

D E S

Xi Fu

August 22nd 2007

A U

Aalborg University

Department of Electronic Systems

Fredrik Bajers Vej 7 ▪ DK-9220 Aalborg East Telephone +45 96 35 86 00

Title: Initialization of Speaker Segmentation Using Friends and Enemies Algorithm

Theme: Methods and Algorithms of Initialization for Speaker Segmentation

Project period: 1st September 2006 - 22nd August 2007

Project group:

1093

Group members:

Xi Fu

Supervisors:

Zheng-hua Tan

Morten Højfeldt Rasmussen

Publications: 4
Pages in
Main Report: 43
Appendix: 9
Finished: 22nd August 2007

Abstract

The aim of this project is to build a speaker segmentation system, applying a non-uniform initialization, Friends and Enemies (FE) algorithm. This algorithm has been proposed in [26].
FE algorithm first does the speaker change point detection via a standard technique based on Bayesian information criterion and put the friend segments which have close likelihoods together to create the initial models for the system.
The parameters in the FE algorithm were fixed in the published paper, but some of them, like the number of friend segments and the number of initial models, did not seem to have well supporting theoretical background for the selection of their values. And the paper did not point out whether the FE algorithm is suitable for different domain data or not.
We test the FE algorithm with different parameters and obtain a purity score of 99.52% together with a low diarization error rate (DER) of 0.48% for the meeting domain data.
We conclude that the FE algorithm is better to use than the uniform segmentation when dealing with data from the meeting domain, but it does not fit data from the broadcast news domain so well.

Preface

This document is written at the Department of Electronic Systems at Aalborg University, Denmark and documents a master thesis for the 9th-10th semesters made by Xi Fu at the signal and information processing in communication systems during the period from September 1st, 2006 to August 22nd, 2007.

This report is written in LaTeX. Formulas, figures and tables are numbered in succession inside each chapter.

All the references to source material are specified with square brackets after the part of the text, where they are used, e.g. [2].

The program code produced while creating the software for the system can be found on the CD-ROM attached to this report together with the audio data and an electronic version of the report.

The software tools for this project are the followings: LabVIEW 8.2, Perl version 5.8.x, Dev-C++ C-compiler, MATLAB 7.1, and the Hidden Markov Model toolkit, HTK, version 3.2.1.

This report applies to the supervisors, the censors, students from the institute and others with interest in this subject.

I would like to thank Zheng-hua Tan and Morten Højfeldt Rasmussen for their patient explanation of the theory and helpful guidance of the implementation. I also would like to thank Asbjørn and my family in China for their big support.

Aalborg University, 22nd August 2007

Xi Fu

Contents

Chapter 1 Introduction

In this introduction part, background and motivation for this project are given in the first section. The second section describes current techniques on speaker segmentation clustering. A problem description about this project is represented in the last section of this chapter.

Motivation

Speech is one of the most important ways for people to communicate with each other. This produces large amounts of information in audio form every day from sources like phone conferences, voice mails, radio broadcast news, meetings, lectures, etc. Due to the decrease in the cost of processing power, storage capacity and network bandwidth, an increasingly part of information gets stored in digital form, so it can be accessed later on. Since it is very time-consuming to do segmenting and indexing of a large amount of data manually, there is a growing need to automatically facilitate the searching and indexing of the stored information. For example, to find and select the useful and important information from a large amount of audio data without listening to thousands of hours from the beginning to the end of the record becomes more and more essential and practical. Sometimes the particular content spoken by a single speaker might be of interest and wanted to be looked at in a certain place of the record using some search tools. Speaker segmentation is one of the tools to identify speech which belongs to different speakers spoken in a audio recording. This is especially useful when there is no knowledge about the number of different speakers in the recording or the speech characteristics of the speakers.

Technologies for Speaker segmentation

Speaker segmentation is an important subproblem of audio diarization. In general a recorded speech signal is a single-channel recording which contain multiple audio sources. These sources could be different speakers, music, or different kind of noises, etc. The aim of audio diarization is to mark and categories the audio sources into different audio groups. If the groups are made according to different speakers, the process is called speaker segmentation. And the task is to mark where speaker changes occur in the detected speech and associate segments of speech (a segment is a section of speech data limited by a certain time) coming from the same speaker. Unlike audio diarization techniques that aim to find what a person is saying, speaker segmentation focus on finding out when a person is speaking. But unlike speaker detection or tracking tasks [11], there is no prior knowledge about the speech characteristics of the speakers before the process starts, so these have to be derived from the same data that is going to be used to find the speaker changing points. Speaker segmentation can be done word-dependant or word-independent. When word-dependant we not only look at when a person is speaking, but

also when some exact words are being spoken. In this report, we use a word-independent technique as we have no a priori knowledge of the words in the speech signal.
There are three primary domains which have been used for speaker segmentation research and development [22]: meeting domain, broadcast domain and telephone conversation domain. The segmentation challenges are different for the different audio data according to their quality of recordings, number of speakers, the speaking duration of each speaker and the sequence of speaker changes, etc. But usually high-level system techniques work well over different domains [27], [24].
The predominant approach used in segmentation systems is using agglomerative clustering (a cluster is defined to be a set of segments, not necessarily contiguous, but share some acoustic similarity) with a Bayesian Information Criteria (BIC) based stopping criterion [4] consisting of the following steps [22]:

1. Divide the speech data segments into initial clusters;
2. Compute the pair-wise distances between every two clusters;
3. Merge closest clusters to a new cluster;
4. Update the distances of remaining clusters to the new cluster;
5. Iterate step 2 to 4 until stopping criterion is achieved.

To generate the initial clusters, uniform segmentation is often used as a simple initialization technique [14], while non-uniform segmentation techniques [13], [26] have also been developed to try to get a better performance.
The audio data in every cluster in the system is generally represented by a single full covariance Gaussian [27], [18], [15], while Gaussian Mixture Models (GMMs) have also been used widely to get higher resolution for cluster representation [3], [19], [5].
An alternative approach described in [10] for cluster training is to use a Euclidean distance between MAP-adapted GMMs. This is highly correlated with a Monte Carlo estimation of the Gaussian Divergence distance while also being an upper bound to it. The stopping criterion uses a fixed threshold, chosen on the development data, on the distance metric. This method is more conventional comparing to the BIC method.

Problem statements

As mentioned in the previous section that the initial segmentation is the first and an important step for the whole clustering process. Prior work [26], pointed out that a non-uniform segmentation called Friends and Enemies (FE) algorithm could improve the system performance. The concept of the FE algorithm is to divide the segments for the initial clusters as "different enemies" with their "friends segments". This process should guarantee that one cluster only contains one speaker, which is different from the uniform segmentation with a much higher probability to make initial clusters consist of segments from multiple speakers.
There are several parameters to be considered in the FE algorithm, such as the number of initial clusters, the number of friendly segments within a cluster and the parameters in preprocess before the clusters division. This project is aiming at testing different parameters in the cluster initialization step using the FE algorithm, and analyze the performance of the FE for both the meeting domain and the broadcast domain.

Chapter 2 Analysis

The following sections describe the speaker segmentation (Section 2.1) and the concepts behind it in greater detail. Section 2.2 gives explanations about the basic mechanic of the speech production and speech features which can be used to distinguish different speakers and even different sounds. In the next section, 2.3, Hidden Markov Models (HMMs) and GMMs will be introduced to model the characteristics of individual speakers in a mathematical way. To classify between different models the log-likelihood is used, explained in Section 2.4. For the merging decision we use the Bayesian Information Criteria, this will be outlined in the last Section 2.5 of this chapter.

2.1 Speaker Segmentation

An overview of the process of speaker segmentation is shown in Figure 2.1.

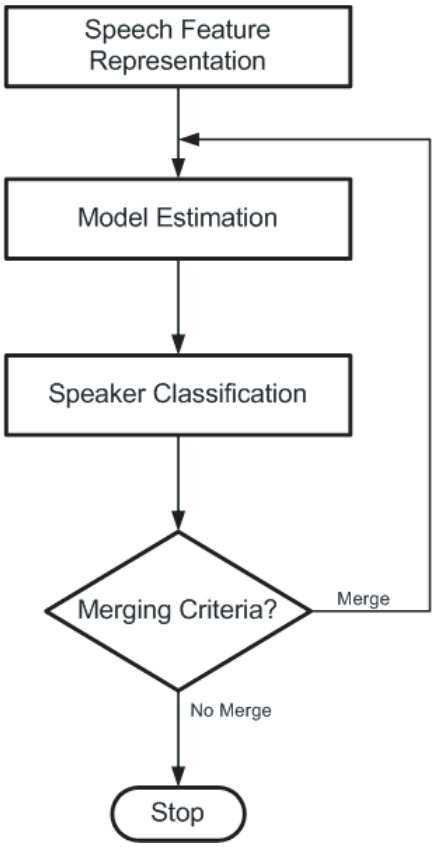

Figure 2.1: The block diagram for speaker segmentation.

The first part of Speaker segmentation requires a transformation of the speech signal into recognizable features. These features can be used to train a number of models representing different speakers. The models are then used to classify an unknown speech signal containing these speakers. After that, the task is to find out when every single speaker is speaking. As the speech signal for training the models and classification is the same, it is necessary to make the number of models greater than the real number of speakers. A merging criteria can then be decided and two models are merged if they satisfy this criteria. This will lead to a retraining of the models and a new merging decision. If no models are merged, the loop will end.

2.2 Speech Feature Representation

In this section, the speech analysis is divided into three parts, the mechanic principle of human speech production (2.2.1), the theory of speech modeling (2.2.2) and typical feature representations (2.2.3).

2.2.1 Speech production

In the view of anatomy the speech production system of human beings includes the lungs, larynx, pharyngeal cavity, oral cavity, and nasal cavity. In technical discussions, the pharyngeal cavity, oral cavity, and nasal cavity are often attributed to one unit which is called the vocal tract. This leads to a division into, the lung, larynx, and vocal tract. The lung is the power supply and provide the airflow into the larynx. The larynx modulates the airflow by vibration which result in sounds. These sounds are what make up a person's voice. The vocal tract further modulates the airflow to produce recognizable words, primarily with the usage of tongue and lips. Figure 2.2 shows the anatomy architecture of speech production from a simplified view.

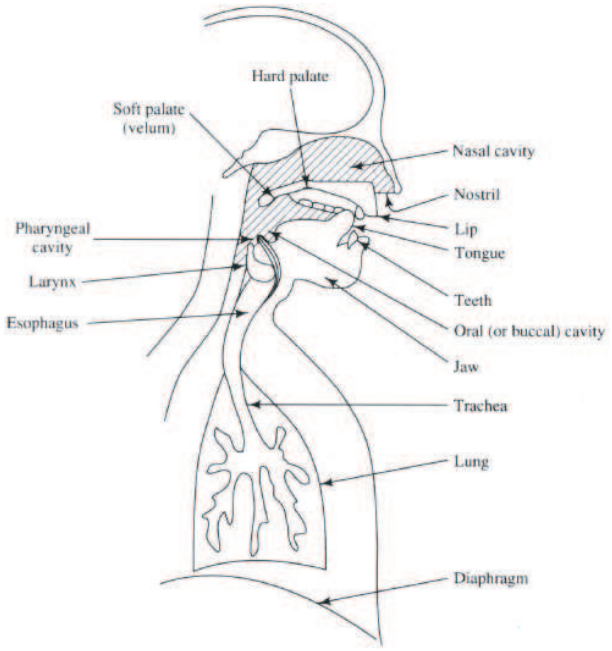

Figure 2.2: The anatomical structure of human speech production system. This figure is taken from [9].

2.2.2 Speech Modeling

During normal speech the lungs and larynx (the source) generates airflow of three different characteristic properties: periodic, noisy, and impulsive. Combinations of the three are often present. Speech from a specific person is not only determined by the product of the source but also, what is more important, by the vocal tract modulation. This can be modeled as a filter. We can recognize different sounds from an individual speaker by

both the properties of the source and the filtering in vocal tract. These sounds are the spoken words that make up the speech.
A block diagram of the source-filter model is shown in Figure 2.3, where $e(n)$ is the excitation signal from the source, $h(n)$ is the filter transfer function and $s(n)$ is the modeled speech signal, all three are in the time domain. $E(w)$, $H(w)$ and $S(w)$ are the same signals in the frequency domain.

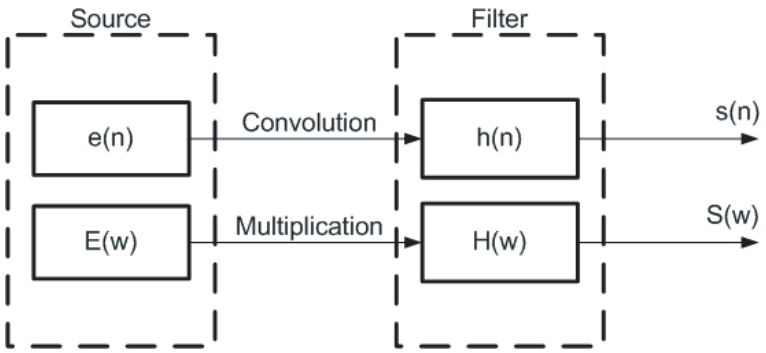

Figure 2.3: The source-filter model of speech production.

While the functionality of the source is mostly related to the amplitude of the voice, the modulation of the filter gives the voice frequency related characteristics. This gives that the discriminating features between different speakers have to be found in the parameters of the filter ($H(w)$).

2.2.3 Feature Selection

In order to do any speaker classification, it is necessary to find a way to transform the speech data, so the characteristics of an individual speaker become as clear as possible. These characteristics can be described as the features of the given data. We say the values of these features represent different classes or models, one for every individual speaker.
For speaker classification, Cepstral Coefficients (CC) are often used features [2], [21], [23] to distinguish one person's voice from another.

Cepstrum Method

Cepstral Coefficients, derived from cepstrum plots, are useful to separate the the voice characteristics of the larynx and vocal tract from the lungs [9]. Giving the previous description on how the different organs used in speech production, this can be represented by a source-filter model, the following section describes a method to get a mathematical description of the filter parameters.
As shown in Figure 2.3, a discrete speech signal $s(n)$ can be given in the time-domain by the convolution of the excitation signal $e(n)$ and the impulse response of the joint filtering of the larynx and vocal tract $h(n)$.

$$s(n) = e(n) * h(n) \tag{2.1}$$

Where "$*$" notes the convolution. Taking the Discrete Fourier Transform (DFT) of 2.1, we will get,

$$S(w) = E(w)H(w) \tag{2.2}$$

Using logarithms, we will get

$$log|S(w)| = log|E(w)| + log|H(w)| \tag{2.3}$$

To get the cepstrum $c_s(n)$ of the speech signal $s(n)$, taking the Inverse Discrete Fourier Transform (IDFT) of Equation 2.3

$$IDFT(log|S(w)|) = IDFT(log|E(w)|) + IDFT(log|H(w)|) \tag{2.4}$$

$$c_s(n) = c_e(n) + c_h(n) \tag{2.5}$$

The whole process is visualized in a block Figure 2.4

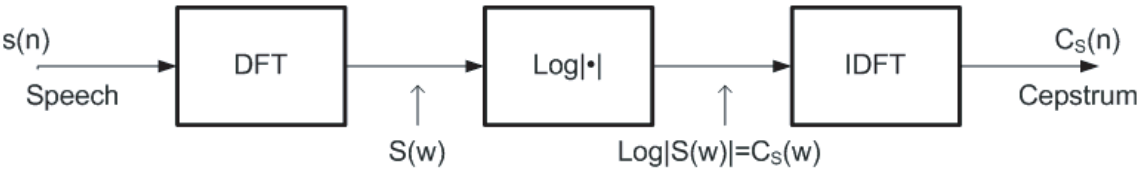

Figure 2.4: The process of cepstrum method.

An overview of the cepstrum method is illustrated by plotting all the figures in every step as shown in Figure 2.5.

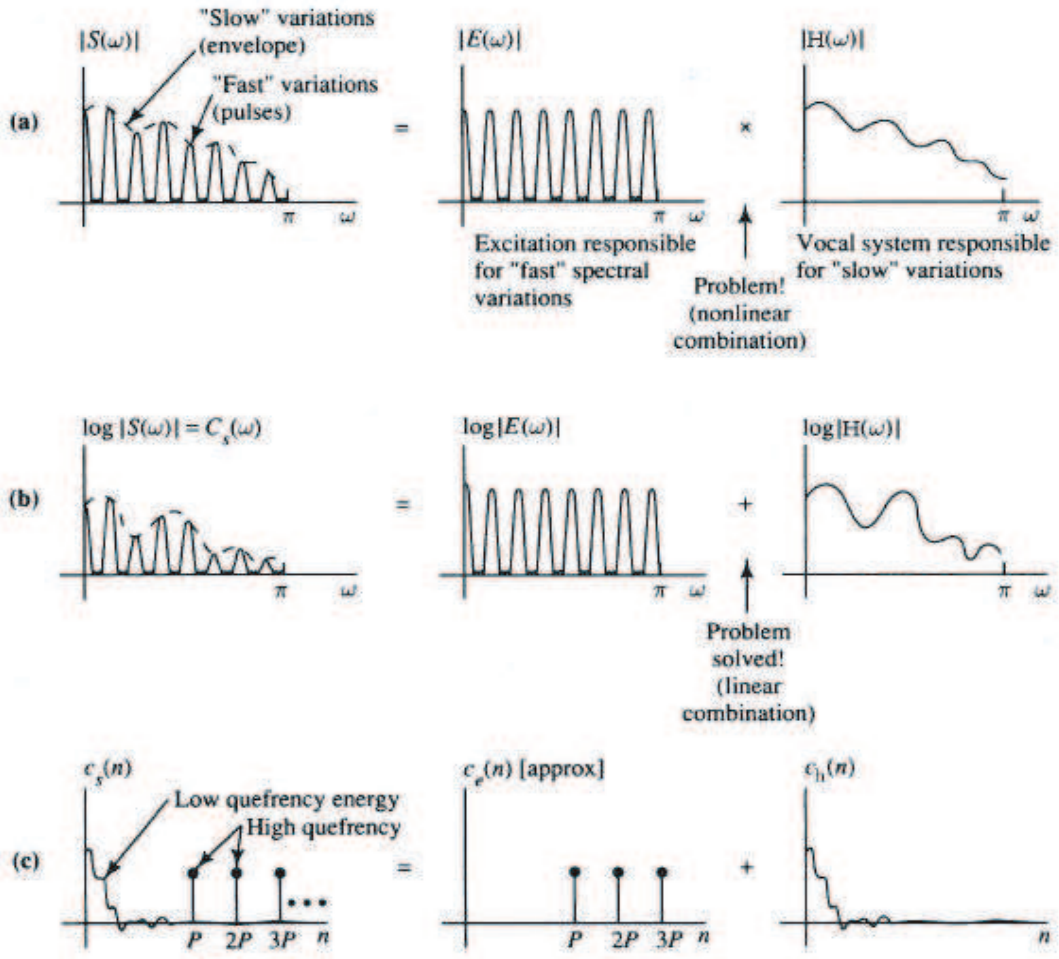

Figure 2.5: The process of cepstrum analysis. (a) In the speech magnitude spectrum, two components, $H(w)$ as a "slowly varying" part (envelope) and $E(w)$ as a "quickly varying" part can be identified and combined by multiplication. (b) Once taking the logarithm, the two convolved signal components are combined by addition. (c) When applied by IDFT, "slowly varying" components are shown in the low quefrencies and "quickly varying" components are shown in the high quefrencies. The figure is taken from [9].

Viewing the logarithmic speech spectrum in the figure 2.5, $\log|S(w)|$ contains two components, a slowly varying part, which is seen as the envelop in a log-magnitude plot, and a fast varying part which is seen as the ripples. As it is desired to separate these two parts from $\log|S(w)|$, the IDFT is used. Since the "signals" $\log|S(w)|$ is already in the frequency domain, a new word "quefrency" is used to describe "frequencies" in this new "frequency domain". The cepstrum plot $c_h(n)$ can then be separated from $c_s(n)$ as the low quefrency part.

CCs are cepstrum plot values in relation to n. If, for example, a wanted number of CCs are 20, the first 20 $c_h(n)$ values will become the CC values. The higher the number of CCs is, the better the characteristic of the larynx and vocal tract will be represented.

Mel-Frequency Cepstrum

To calculate CCs, a method called mel-based cepstrum can be applied. The CCs obtained from this method are called Mel-Frequency Cepstral Coefficients (MFCCs). A mel is a measurement unit which can be applied to measure the perceived pitch or frequency of a tone. The measurement is related to human hearing behavior. The mel-frequency does not correspond linearly to the real world frequency since the human ear perceives physical frequency non-linearly. Investigators [9] has been able to determine a mapping between the real frequency and the mel-frequency. An approximation of this mapping can be written as

$$F_{mel} = 1000 \cdot log_2[1 + \frac{F_{Hz}}{1000}] \tag{2.6}$$

where F_{mel} is the mel-frequency and F_{Hz} is the physical frequency both in the unit Hz.
A plot of equation 2.6 is seen in figure 2.6. It shows that the mapping from the real frequency to the mel-frequency is approximately linear below 1kHz and logarithmic above.

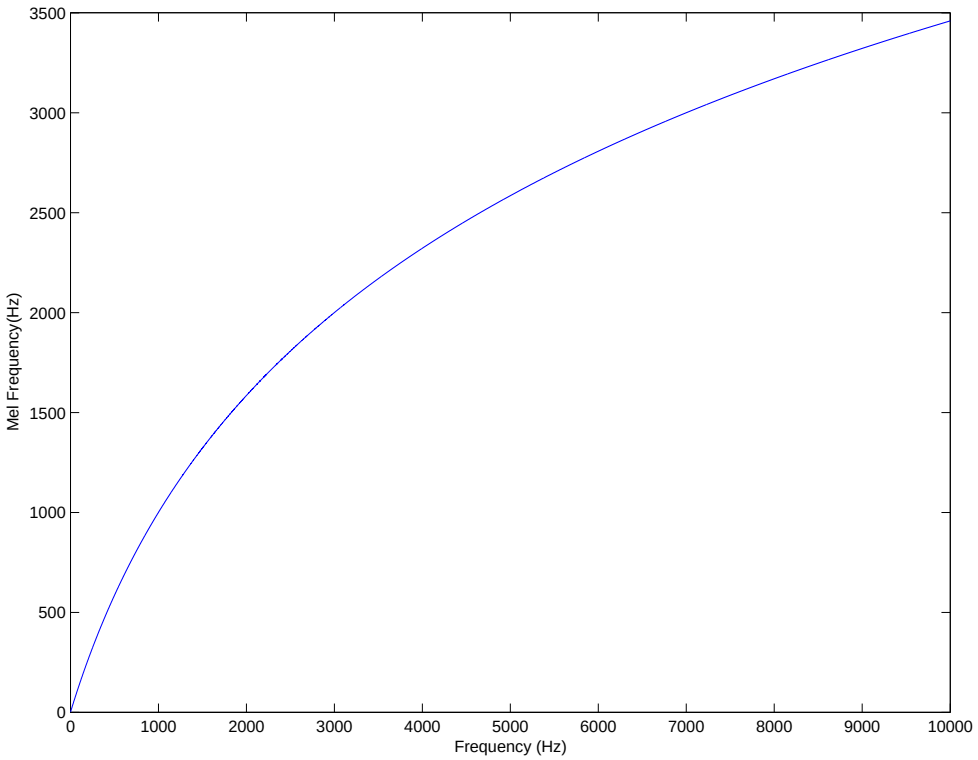

Figure 2.6: The mel frequency scale.

Previous works [9], [6] show that using a filter-bank to map physical frequency onto the mel frequency together with speech analysis provides a good performance in the speech research field, e.g. speech recognition and identification. These filter-banks can be explained in 2.7

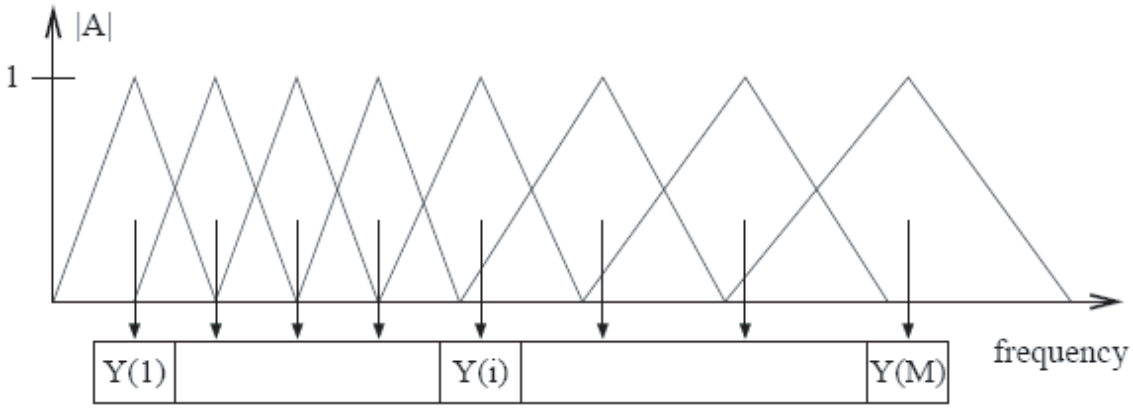

Figure 2.7: The mel frequency filter bank, this is taken from [14].

where the number of triangulars M is the number of filters it is using and $Y(i)$ denotes the sum of the weighted log-magnitude spectrum $log|S(k)|$ within the ith critical band filter which also means the log energy in the ith critical band. This can be expressed as [14]

$$Y(i) = \sum_{k=0}^{\frac{N}{2}} log|S(k)|H_i(k) \tag{2.7}$$

In this equation, $H_i(k)$ denotes the ith conceptual critical band filter and N is the length of the DFT. In order to get the MFCCs, the IDFT and the Discrete Cosine Transform (DCT) can be applied afterward, such as:

$$c_{mf}(n) = \sum_{i=1}^{M} Y(i)exp(jk\frac{2\pi}{M}n) = \sum_{i=1}^{M} Y(i)cos(k\frac{2\pi}{M}n) \tag{2.8}$$

2.3 Model Estimation

After having found a way to compute discriminating features it is still required to decide on a model representation of these features. A probabilistic way to approach this is using Gaussian Mixture Models (GMM) and Hidden Markov Models (HMM). While GMM is a model representation of a features probability density function (pdf), HMM is a model of feature "behavior" at time instances (called states) following each other. Both model techniques can be used independently but in speaker segmentation they are often used together [17], [20]. In this section, the general idea of HMM will be explained, followed by the introduction of GMM. The combined usage of both HMM and GMM for speaker modeling will be also described.

2.3.1 Hidden Markov Model

In this section, the principles of Hidden Markov Models will be outlined from the concept extension of a Markov Model.

Markov Model

A Markov Model (MM) contains a number of states $S_1, S_2, S_3, ..., S_N$, and the parameters of the model consist of the probabilities to shift (transit) from one state to another and even to stay in the same state. A Markov chain is a stochastic model whose probabilistic value is truncated to the current and the previous state. If we denote q_t as the current state at time t, the transition probability a_{ij} from state i to state j is expressed as:

$$a_{ij} = P(q_t = S_j | q_{t-1} = S_i) \tag{2.9}$$

To obey the standard stochastic constrains, the state transition coefficients have the following properties

$$a_{ij} \geq 0 \tag{2.10}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{2.11}$$

This leads to a transition matrix of a MM which is defined as

$$\mathbf{A} = a_{ij} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1,N-1} & a_{1,N} \\ a_{21} & \ddots & & & a_{2,N} \\ \vdots & & & & \vdots \\ a_{N-1,1} & & & \ddots & a_{N-1,N} \\ a_{N,1} & a_{N,2} & \cdots & a_{N,N-1} & a_{N,N} \end{pmatrix} \tag{2.12}$$

Following the condition given in equation 2.11, the summation of each row in the matrix equals to one. A MM which contains three states is illustrated in Figure 2.8. It is possible to have a zero transition probability between states as in the example shown in the figure, as there is no connection from state 2 to state 3. If all the transition probabilities are higher than zero, the MM is called an ergodic MM.

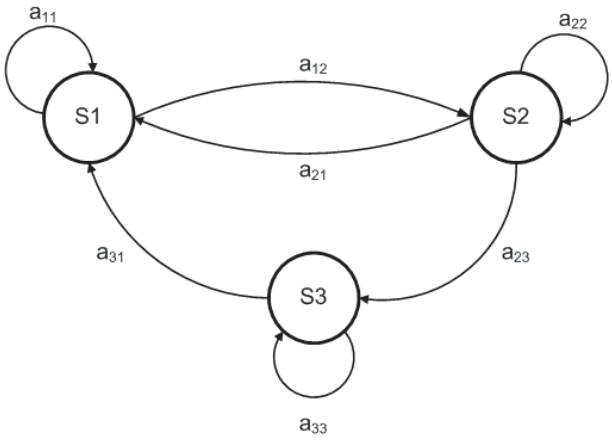

Figure 2.8: A 3 states MM marked with transition probabilities.

An observation sequence O is defined as a group of symbols going through the states. An example could be $O = S_2, S_2, S_1, S_1, S_2, S_3$ at $t = 1, 2, 3, 4, 5, 6$. The probability of this observation sequence given the Markov Model is

$$P(O|Model) = P(S_2, S_2, S_1, S_1, S_2, S_3|Model) = P(q_1 = S_2)a_{22}a_{21}a_{11}a_{12}a_{23} \tag{2.13}$$

The initial state probabilities can be noted as

$$\pi_i = P(q_1 = S_i), 1 \leq i \leq N \tag{2.14}$$

A Markov Model can be shortly expressed by both the transition matrix **A** and the initial state probabilities for each state π_i, as 2.15.

$$\gamma = (\mathbf{A}, \pi_i) \tag{2.15}$$

Extension from Markov Models to Hidden Markov Models

In a MM, each state corresponds to an observable event. If the connection between the state and the observation is not observable, a pdf has to be used to estimate the relation between observation and state value. This kind of Markov Model is called a Hidden Markov Model.
An example can be seen in Figure 2.9. Showing in the figure, the states can only be observed by different stochastic processes tied to each state, which is illustrated as the pdf plots.
There are five things which characterize an HMM [16]:

1. N, the number of states in the model. Different states can be denoted the same way as MM, $S = S_1, S_2, ..., S_N$
2. M, the number of distinct observation events per state. The individual symbols can be denoted as $\mathbf{V} = \mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_M$
3. **A**, the transition probability matrix, same as in the MM.
4. $\mathbf{B} = b_j(\mathbf{x})$, the observation symbol pdf, which is given by

$$b_j(\mathbf{x}) = P(v_k \; at \; time \; t \; |q_t = S_j), \;\; 1 \leq j \leq N \; and \; 1 \leq k \leq M \tag{2.16}$$

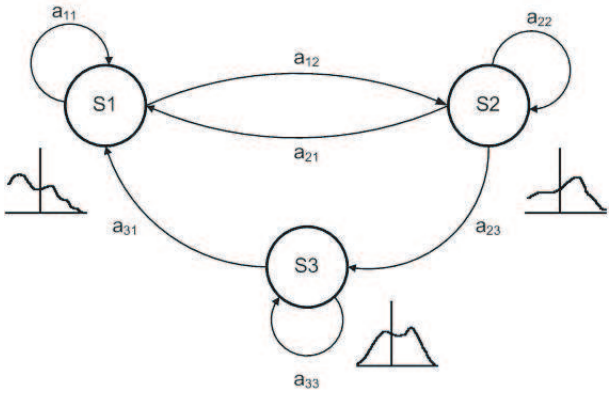

Figure 2.9: A 3 states HMM marked with transition probabilities and one pdf per state.

5. π_i, which is the same notation as the initial state probabilities in MM.

Combining these five elements, a HMM can be explained using a compact notation

$$\boldsymbol{M} = (\mathbf{A}, \mathbf{B}, \pi_i) \tag{2.17}$$

2.3.2 Gaussian Mixture Model

There are different ways to describe the pdfs of the states in a HMM, one way is by using Gaussian Mixture Models. In this section, the form of the GMM will be explained as well as its usage for speaker identification.

Model description

A Gaussian Mixture Model is a probability distribution that is a combination of multiple Gaussian distributions. This is given by the equation 2.18.

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{I} c_i N(\mathbf{x}; \mu_i, \mathbf{\Sigma}_i) \tag{2.18}$$

In the equation, I is the number of Gaussian mixtures, $\mathbf{x}$ is a D-dimensional random vector, c_i are the mixture weights that satisfy the constraints $0 < c_i < 1$ and $\sum_{i=1}^{I} c_i = 1$. $N(\mathbf{x}; \mu_i, \mathbf{\Sigma}_i)$, is the pdf of a single Gaussian distribution, which obeys a D-variate Gaussian function form

$$N(\mathbf{x}; \mu_i, \mathbf{\Sigma}_i) = \frac{1}{(2\pi)^{\frac{D}{2}} |\mathbf{\Sigma}_i|^{\frac{1}{2}}} exp\{-\frac{1}{2}(\mathbf{x} - \mu_i)'(\mathbf{\Sigma}_i)^{-1}(\mathbf{x} - \mu_i)\} \tag{2.19}$$

with mean vector μ_i and covariance matrix Σ_i.
From these definitions, a GMM can be parameterized by the mixture weights, the mean vectors and covariance matrices from all the component densities. We can use the compact notation to represent a GMM as

$$\lambda = (c_i, \mu_i, \mathbf{\Sigma}_i), \ i = 1, \ldots, I \tag{2.20}$$

Motivation to use GMM

As described previously is it desired to make a probabilistic model of the Cepstral Coefficients that can represent their pdfs. The histogram of the values of a single CC over multiple speech segments produced by the same speaker might look like what is shown in Figure 2.10 (a). This kind of distribution is clearly not one of the "standard" distributions (Gaussian, Laplacian, etc.), but with the help of GMM it is possible to make an representative model of the pdf that can be described mathematically. Figure (b) shows how the pdf of an 8 component GMM and the underlying component densities, and figure (c) shows how the GMM approximates the histogram of the CC.

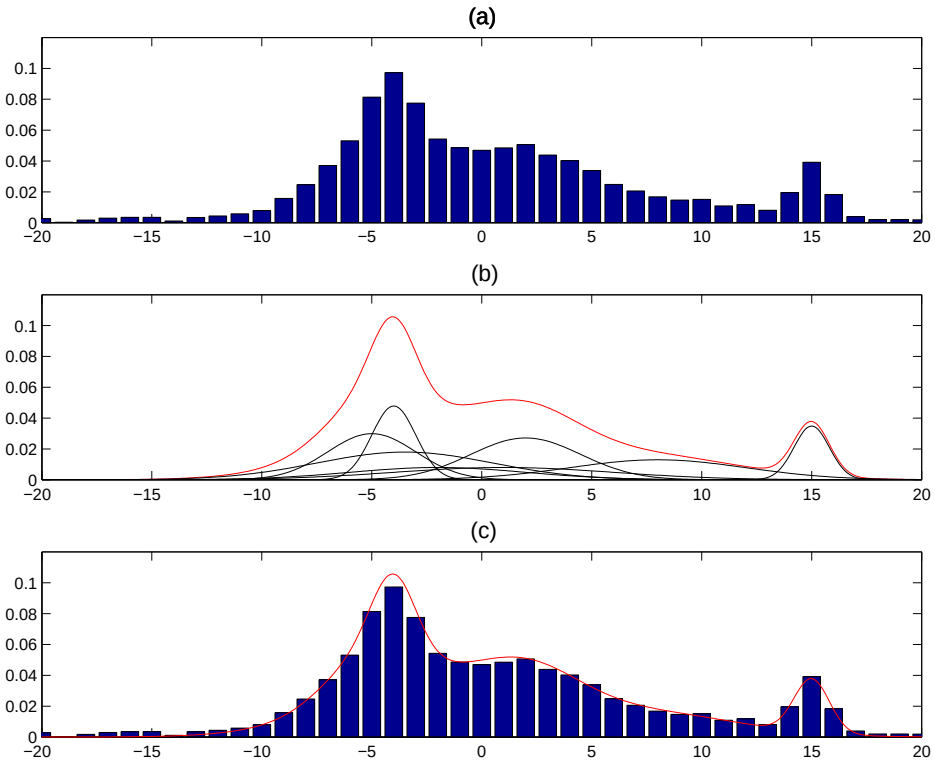

Figure 2.10: Histogram distribution modeling: (a) Histogram of CC; (b) GMM and its 8 component densities; (c) How GMM approximates the histogram of the CC.

Training GMM

The speech characteristics for any individual speaker who was previously found to be most distinctive by converting the speech data to the cepstrum domain can now be used to train the GMM's. That requires a sequence of K training vectors $\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K$ that are known all to belong to the same speaker. These can now be used to compute the parameter values $c_i, \mu_i, \mathbf{\Sigma}_i$ of the model λ. There are different methods to estimate these parameter values of a GMM [12], by far the most popular way is maximum likelihood (ML) estimation.
An ML estimation is to find a model which contains parameter values which maximize

the likelihood of the GMM, given the training data. For the sequence of training vectors, the GMM likelihood can be calculated as

$$p(\mathbf{X}|\lambda) = \prod_{k=1}^{K} p(\mathbf{x}_k|\lambda). \tag{2.21}$$

ML parameter estimation can be done iteratively using a special case of the expectation-maximization (EM) algorithm [1].
The general idea of the EM algorithm is to begin the process with an initial model λ_0, using the parameter values of this initial model to create another model λ_1, which will satisfy the expression $p(\mathbf{X}|\lambda_1) > p(\mathbf{X}|\lambda_0)$. Indicating that the new model λ_1 "fits" the training data better than the old model λ_0. Then the parameter values of λ_1 are used to compute λ_2 which also have to satisfy the evaluation expression, in general terms: $p(\mathbf{X}|\lambda_{new}) > p(\mathbf{X}|\lambda_{current})$. Ideally should these iterations continue until the evaluation expression no longer is fulfilled. Though in practical applications a convergence threshold will be used instead. The final model will become the last model from the process which contains the most ideal parameters that best describe the training data for the individual speaker. The EM algorithm shares the same basic technique used for estimating HMM parameters when using the Baum-Welch algorithm [7]. The reason why this report does not give much insight how to estimate all the parameters of a HMM, in particular the **A** matrix will become clear in the implementation part of this report, see Chapter 3. This information can be found in [16].
The mathematical algorithms used for every EM iteration are described by the following equations. These guarantee a monotonic increase in the model's likelihood value:

- Mixture Weight:

$$c_{i,new} = \frac{1}{K} \sum_{k=1}^{K} p(i|\mathbf{x}_k, \lambda_{current}) \tag{2.22}$$

- Means:

$$\mu_{i,new} = \frac{\sum_{k=1}^{K} p(i|\mathbf{x}_k, \lambda_{current})\mathbf{x}_k}{\sum_{k=1}^{K} p(i|\mathbf{x}_k, \lambda_{current})} \tag{2.23}$$

- Variances:

$$\sigma^2_{i,new} = \frac{\sum_{k=1}^{K} p(i|\mathbf{x}_k, \lambda_{current})\mathbf{x}_k^2}{\sum_{k=1}^{K} p(i|\mathbf{x}_n, \lambda)} - \mu^2_{i,new} \tag{2.24}$$

The *a posteriori* probability for acoustic class i is given by

$$p(i|\mathbf{x}_k, \lambda_{current}) = \frac{c_{i,current} N(\mathbf{x}_k; \mu_{i,current}, \mathbf{\Sigma}_{i,current})}{\sum_{j=1}^{J} c_{j,current} N(\mathbf{x}_k; \mu_{j,current}, \mathbf{\Sigma}_{j,current})} \tag{2.25}$$

It is important to point out that there is no good theoretical way to decide on the parameter values of the initial model λ_0.

2.4 Speaker Classification

After having trained multiple GMMs based on the characteristics of different speakers it is now possible to identify these speakers in a speech signal where their voice is present. The speech signal is divided into segments covering small time windows and the classification procedure classify the most probable speaker of every segment one by one. The result becomes a label file with start and end time for the segments and the speaker ID for every segment. The following section describes the classification procedure on a single segment.
A group of speakers S is represented by trained GMM models as $\lambda_1, \lambda_2, ..., \lambda_S$. The CC values of the test segment represented as the vector $\mathbf{x}_{test}$ of same dimension as the previous training vectors. The goal is simple to find the model that will give the highest likelihood value computed with the test vector. This is described as, [17]:

$$\hat{S} = \arg \max_{1 \leq s \leq S} Pr(\lambda_s | \mathbf{x}_{test}) = \frac{p(\mathbf{x}_{test} | \lambda_s) Pr(\lambda_s)}{p(\mathbf{x}_{test})} \tag{2.26}$$

second equation is due to Bayes' rule. $\hat{S}$ is the speaker number of the most likely speaker of the segment. If we assume equally likely speakers, the $Pr(\lambda_s) = 1/S$ and note that $p(\mathbf{x}_{test})$ is the same for all speakers the classification rule is simplified to:

$$\hat{S} = \arg \max_{1 \leq s \leq S} p(\mathbf{x}_{test} | \lambda_s) \tag{2.27}$$

Taking the logarithm to this provides the log likelihood probability:

$$\hat{S} = \arg \max_{1 \leq s \leq S} \log p(\mathbf{x}_{test} | \lambda_s) \tag{2.28}$$

2.5 Merging Criteria

While modeling a certain amount of data, and applying the EM algorithm to select the best fit model, it is reasonable to believe that a model of high complexity will score a high likelihood with the data. On the other hand a highly complex model takes long computational time while maybe not really improve the likelihood. The Bayesian Information Criteria (BIC) can be applied to make a trade off between the model complexity and computational time.
If the sequence of data $\mathbf{X} = x_1, x_2, ..., x_K$ is going to be modeled by model $M = (M_s)$, $s = 1, 2, ..., S$ with a fixed number of free parameters, the definition of BIC is given in [4] as:

$$BIC(M_s) = \log L(\mathbf{X}|M_s) - \xi\frac{1}{2}\sharp(M_s)\log(K) \tag{2.29}$$

In the equation, there are several parameters which can be listed as:

$L(\mathbf{X}|M_s)$ The maximum likelihood of the data given by a certain model M_s

$\sharp(M_s)$ The number of free parameters in the model M_s, in our work, it refers to the number of Gaussian mixtures in the model

K The number of data points in the data $\mathbf{X}$

ξ The penalty weight with an ideal value of 1

In the speaker segmentation case, BIC can be used both to decide if two segments belong to the same speaker and should be merged into one model or if they belong to two different speakers. In that case we put a "changing point" between them. Suppose there are three models noted as M_a, M_b and M_c. M_a is trained on data segment a and M_b is trained on the following data segment b. M_c is trained on both segment a and b, this is illustrated as shown in figure 2.11. The BIC for the combined model c and the BIC for

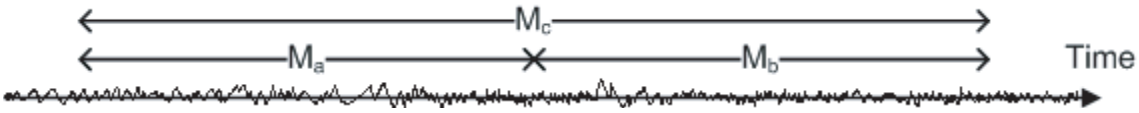

Figure 2.11: Three models structure

individual models a, b are denoted as $BIC(M_c)$ and $BIC(M_a, M_b)$. The interesting part is the difference between these two BIC values, which can be expressed as:

$$\Delta BIC = BIC(M_c) - BIC(M_a, M_b) \tag{2.30}$$

A negative ΔBIC means it is better to keep the data separated in two models which indicates a potential speaker change. On the contrary, a positive ΔBIC means merging two models into one is favored.
To compute the ΔBIC, we can use probability instead of maximum likelihood which gives the equation 2.30.

$$\Delta BIC = logP(\mathbf{X_c}|M_c) - \lambda\frac{1}{2}\sharp(M_c)log(N_c) - [logP(\mathbf{X_a}, \mathbf{X_b}|M_a, M_b) - \lambda\frac{1}{2}\sharp(M_a + M_b)log(N_a + N_b)] \tag{2.31}$$

Since data a and b are independent from each other together with their models M_a and M_b, then

$$P(\mathbf{X_a}, \mathbf{X_b}|M_a, M_b) = P(\mathbf{X_a}|M_a)P(\mathbf{X_b}|M_b) \tag{2.32}$$

And note that

$$N_c = N_a + N_b \tag{2.33}$$

Put 2.32 and 2.33 into 2.31, we will get

$$\Delta BIC = logP(\mathbf{X_c}|M_c) - [logP(\mathbf{X_a}|M_a) + logP(\mathbf{X_b}|M_b)] - \lambda\frac{1}{2}[\sharp M_c - \sharp(M_a + M_b)]logN_c \tag{2.34}$$

In this equation, $\lambda\frac{1}{2}[\sharp M_c - \sharp(M_a + M_b)]logN_c$ is the penalty term for ΔBIC which states that if M_c has a higher number of parameters than the parameter sum of M_a and M_b, the likelihood of the M_c be penalized. If we put the number of parameters for the combined the model equal to the summation number of parameters in the two candidates models, it will make $\sharp M_c - \sharp(M_a + M_b) = 0$, and the penalty term will disappear. And the final expression for ΔBIC will become

$$\Delta BIC = logP(\mathbf{X_c}|M_c) - [logP(\mathbf{X_a}|M_a) + logP(\mathbf{X_b}|M_b)] \tag{2.35}$$

Chapter 3 Implementation

The aim of this chapter is to describe the implementation of our speaker segmentation algorithms with primary focus on the FE cluster initialization algorithm. The remaining parts of the speaker segmentation procedure are taken from a baseline system produced by [14]. The implementation for both baseline system and FE algorithm uses functionality provided by HTKTools. The HTKTools is a toolbox for HMM with GMM state descriptions developed by [20].
In the implementation of FE algorithm, we are only interested in GMM so we simply use several single state HMMs to model the data in different segments. Due to the implementation of HTKTools even a "single" state HMM will contain more than one state as HTKTools will automatically apply two pseudo states to any HMM - a start and an end state.
The start state represents the state before the speech data starts and state "End" represents the state after the speech data ends. Therefore, none of these two states are modeled by a GMM and just represent the begin and termination states of the HMM.
The baseline system uses HMM and its functionality is described in the following section.

3.1 System Overview

In this section, the implementation of the baseline system is explained step by step.

3.1.1 Initial cluster Selection

Before the speaker segmentation starts, the speech data has to be divided into different initial clusters. This is done by assigning model labels to segments of the speech data. Two questions have to be answered at this step: how many labels should be applied and how should they be distributed? One method is uniform segmentation [14] where each label covers an equally large part of the data, another is the FE algorithm which will be described later in Section 3.2 in this chapter.

3.1.2 Baseline system

The speaker segmentation works as it can be seen in Figure 3.1. A speech data file containing different speakers is used as the input, and the output file from this system is a label file containing both segmentation and different speaker IDs.

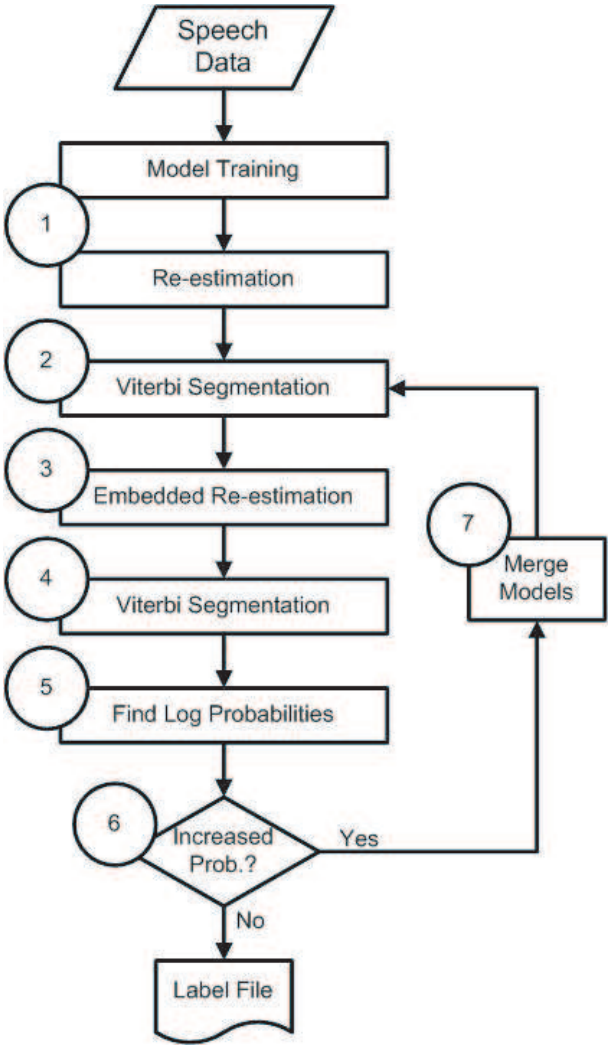

Figure 3.1: Flow graph for speaker segmentation, this is a redrawn figure from [14].

The detail explanations for Figure 3.1 are given as follows:

1. Initial model training

As the first step of the baseline system, Viterbi training and the re-estimation using forward and backward algorithm is used to train parameters of the initial models. The theory of Viterbi training and the forward and backward algorithm are both described in [20] and [16]. They are performed by the HTKTools HInit and HRest.

2. Viterbi Segmentation

Viterbi Segmentation is used [14], [20] to achieve the single best path for a HMM, in a probabilistic sense. It ensures to find the path through the sequence of models that matches the given observations most. This step is done either after the initial training step or after the merging step. HTKTool HVite is used to do the implementation.

3. Embedded Re-estimation

Embedded re-estimation is used after the Vitrbi segmentation. More than one model can be updated at the time with only one transcription file containing no time information [14].
Using embedded re-estimation instead of just using re-estimation is because when the data is segmented and the models are trained before this re-estimation, it might not give the optimal segmentation. Then if this embedded re-estimation is applied, the parameters from the previous models are updated and the optimal parameters will be generated using all the data.
This step is implemented using the HTKTool HERest.

4. Viterbi Segmentation after Embedded Re-estimation

Repeat the Viterbi segmentation as step 2, to make sure that the models match the speech segments most after the embedded re-estimation.
Again this step is performed using HTKTool HVite.

5. Finding Log Probabilities

This step is a preparation for the next merging step. The task is to find the log probabilities of the data given its single model and all the combined data given their combined models. For example, consider three models, M_1,M_2,M_3 and their data X_1,X_2,X_3, then the single log probabilities: $\log P(X_1|M_1)$, $\log P(X_2|M_2)$, $\log P(X_3|M_3)$ and the combined log probabilities: $\log P(X_1, X_2|M_1, M_2)$, $\log P(X_1, X_3|M_1, M_3)$, $\log P(X_2, X_3|M_2, M_3)$ have to be found to continue the next step.
HTKTool HVite is used to find these log probabilities.

6. Merging Criterion

As mentioned in Section 3.1.1, the number of initial models should be bigger than the real number of speaker, that means two or more models could be merged if they are from the same speaker. To decide this, Bayesian Information Criteria is applied. As defined in Section 2.5, two models can be merged if $\Delta BIC > 0$, the log probabilities can be found in step 5. And if more than one pair of models are found which satisfy $\Delta BIC > 0$, the pair with the biggest ΔBIC is merged for this iteration and continue to do the next step. If there is no positive ΔBIC, then stop the process and output the current segmentation.
Note that in order to make the penalty term disappear as mentioned in Section 2.5, we use the sum of the number of Gaussian mixtures in the individual models for the number of Gaussian mixtures in the combined model. For example, for the initial models, we use a GMM with 8 Gaussian mixtures, so the number of Gaussian mixtures in a combined model should be 16. And if later this combined model is merged with another GMM model with 8 Gaussian mixtures, the new combined model should have 16 + 8 = 24 Gaussian mixtures.

7. Merge models

After finding the pair of models which satisfies the merging condition from the last step, discard the two individual models and use the combined model instead. Then go back to step 2 and continue the iteration.

3.2 Friends and Enemies Algorithm

The FE algorithm can be used as the initialization algorithm for the baseline system, it provides non-uniform segments which is different from the uniform segmentation. This cluster initialization algorithm groups the segments which are close to each other and treats different groups as enemies. A log-likelihood metric is used to determine the "friendliness".

The initialization part has often been considered to be of less importance in the past, since later in the process the segmentations and models will be retrained for many iterations which should allow any "pseudo-optimal" initializations to perform as well as any other in the end. But a good initialization should be considered to be the one which does not introduce computational burden to the system and will not propagate the errors all the way to the end of the agglomerative clustering.

The FE cluster initialization is designed to split the acoustic data into K clusters, where K is determined beforehand by running out of the segments or manually set by the user. The initialization is composed of two distinct blocks, as shown in Figure 3.2.

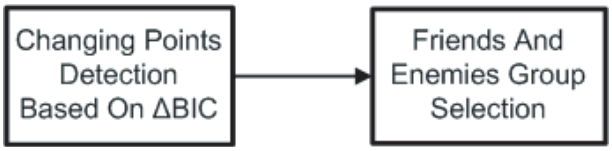

Figure 3.2: Clusters initialization blocks diagram.

3.2.1 BIC Value Calculation and Changing Points Detection

The first block performs a speaker-change detection on the complete acoustic data to identify segments with a high probability of containing only one speaker. This first step is done by using the modified BIC metric [2], the theory is already given in Section 2.5. Model M_a and M_b are generated using the data from two windows with the same size W, and the combined model M_c is made by the data from both these two windows whose length is $2W$. The ΔBIC value is computed over the whole speech data every S frames, where S means a scroll number which controls the step of these three windows. In this report, 0.5 seconds is used for the S value, which means that every 0.5 seconds, a ΔBIC will be computed . This is illustrated in Figure 3.3.

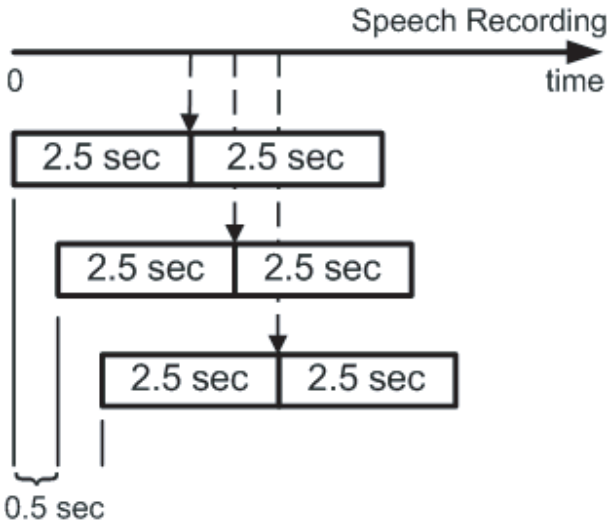

Figure 3.3: ΔBIC calculation using a scroll number of 0.5.

Any point with $\Delta BIC < 0$ is considered as a possible changing point. To restrict the number of possible changing points, we will only select local minimums as the real changing points. The time window for which to look for local minimums is called the minimum duration. In our implementation $W = 2.5$ second windows are used, and each window is modeled using a GMM with 8 Gaussian mixtures, and for the combined model, 16 Gaussian mixtures are used. A 2 seconds minimum duration is selected under the assumption that each speaker is speaking at least for this period at a time. According to the minimum duration, if more than one changing points are found within 2 seconds, the one with the lowest ΔBIC value is chosen as the most possible changing point, and all the remaining points with negative ΔBICs will be ignored. This is shown in Figure 3.4.

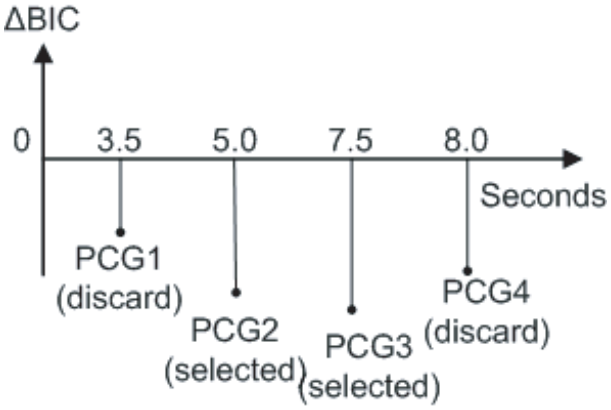

Figure 3.4: Changing points selection using minimum duration.

In the figure, there are four potential changing points named as $PCG1$, $PCG2$, $PCG3$ and $PCG4$, since they all have values lower than zero. But the distance between $PCG1$ and

*PCG*2 is 1.5 seconds, which is smaller than the minimum duration time 2 second, so *PCG*2 with a lower value will be chosen as a more possible changing point and *PCG*1 will be discarded. Using the same comparison, *PCG*3 will be selected instead of *PCG*4 because it has more negative ΔBIC and its distance to *PCG*3 is also less than 2 second.
The ideal case will be that all the changing points right now should be the speaker change points, but according to the reasons: the window size is small and the speech sound variates, there are still more than expected changing points. A threshold can be considered to be set to reduce the number of found changing points. The threshold is set as: first, find the changing point with the lowest ΔBIC, and then the values from 0% to 100% of this lowest ΔBIC are set to be different thresholds. A good threshold is the one which can reduce the number of already existing changing points and still make enough changing points for the later processing, and should include the detected changing points that are closest to the real changing points.
The segments for the FE selection are then set as the data between every two neighboring changing points.

3.2.2 Friends and Enemies Selection

The second block in the initialization algorithm will group the segments from previous step into different clusters as friends and enemies. It is defined that the given acoustic segments are friends if they contain acoustically homogeneous data, and only the best friends can be put together to form a cluster. On the contrary, it is considered that two segments are enemies if their data are acoustically very different from each other. Each cluster contains F segments which are friends of each other.
The whole process can be described with the following steps:

1. First of all, a general model M is made according to all the acoustic data from the first segment S_1 to the last one S_R. And then the log-likelihood for every segment S_i, $i = 1, 2, ..., R$ given by this world model M will be calculated, which can be noted as $\log L(S_i|M)$. The segment S_H, $1 \leq H \leq R$, which has the highest log-likelihood value will be selected as the first enemy. An example with the segment 2 as the initial enemy with the biggest log-likelihood can be seen in Figure 3.5.

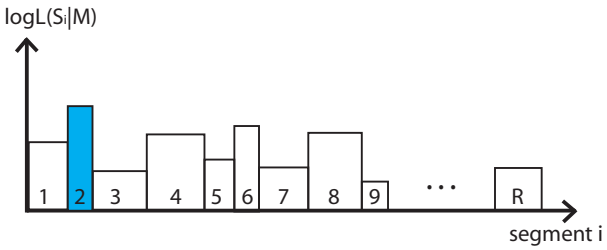

Figure 3.5: Initial enemy segmentation selection.

In the figure, the x-axis refers to the segments from changing points detection, and y-axis shows the log-likelihood values according to every segment given by the global model.

2. Now the task is going to select the friends for the initial enemy segment, first the model M_{S_H} for S_H will be trained using GMM. Afterwards, the log-likelihood of every remaining segment given by model H is calculated, and the $F-1$ segments with higher values are selected as the friends of M_{S_H}. All these F friends will become a initial cluster. Followed by the previous example Figure 3.5, another Figure 3.6 is drawn to illustrate this process. As can be seen in the figure, F is equal to 3 and S_6 and S_8 are chosen to be the friends of S_2.

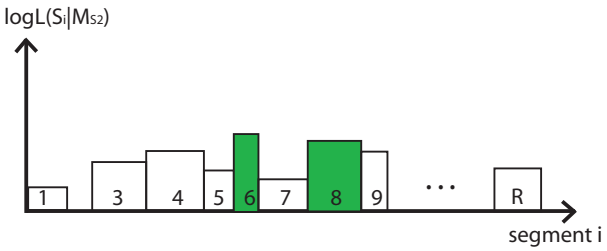

Figure 3.6: Friends selection.

3. The F segments in the first cluster from the previous step will give the source to train a group model M_1, and the log-likelihood of the segments outside the group given by the group model are estimated, using the contrary conception of the friends finding, the segment with the lowest log-likelihood is selected as the enemy of the group. In the following Figure 3.7, an example enemy S_1 is chosen.

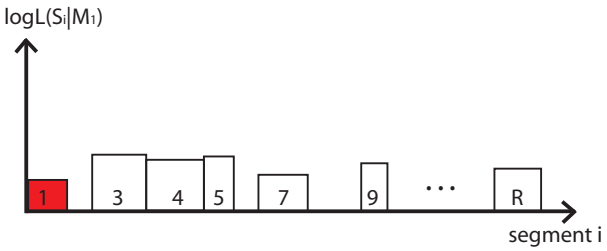

Figure 3.7: Enemy selection.

4. Applying the same way to choose $F-1$ friends for the enemy segment, and build up the second cluster model M_2. The example figure continues being drawn in Figure 3.8.

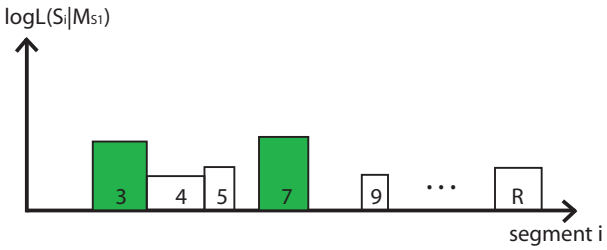

Figure 3.8: Friends selection.

5. A new enemy will be selected, but it is not only the enemy of the first cluster, but also of the second cluster. So the summation of log-likelihood for the segment given by the model of the first and second cluster will give the enemy as the segment with the lowest value. This is seen in Figure 3.9.

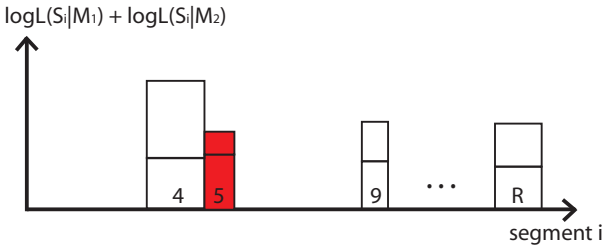

Figure 3.9: Global enemy selection.

6. $F-1$ friends will be found for the new enemy and the process will continue until K clusters are achieved or when it is running out the segments.

Note that if the whole process will stop until K clusters are made, K is also the initial number of models which will go to the baseline system. This value can be set by the users beforehand. Prior work [25], [13] were made by experimenting and pointed out that K could be set either 10 or 16 for the meeting domain, and 40 can be used as in the broadcast news domain.

3.3 Measurement of the Speaker Segmentation

Occurrence matrices, purity index and the diarization error rate (DER) are used as a measurements of performance of speaker segmentation [26], [13]. They are used as to determine the effect of parameter changes in the FE algorithm and are computed by comparing the speaker segmentation result labeling with a reference labeling that gives the exact speaker "who spoke when". An example of a reference labeling is shown in Figure 3.10.

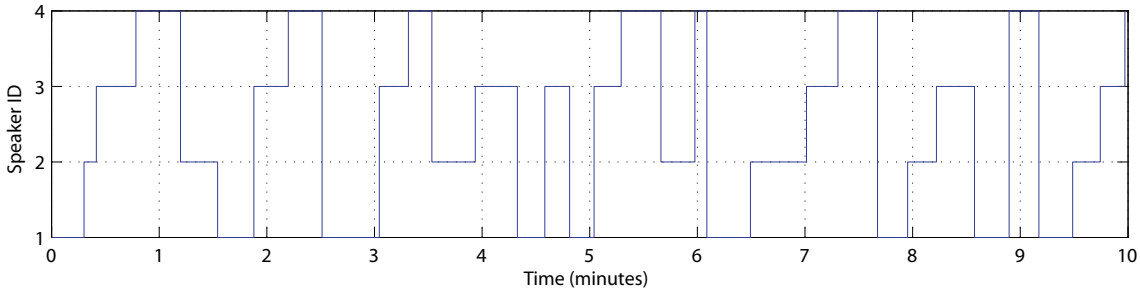

Figure 3.10: Reference labeling which contains 4 different speakers.

The x-axis shows the time in minutes of this speech audio file and the y-axis gives the speaker ID.
The result table contains the occurrence matrix which gives the information of the percentage in time of occurrence between the result labeling and the reference labeling. The purity, DER and number of speakers found are also listed in the result table. An example table is shown as:

```
          R      e      f      e
          S1     S2     S3     S4     | PIDX
R S7      0.2923 0.0011 0.0084 0.0002 | 0.9678
e S2      0.0000 0.1493 0.0001 0.2040 | 0.5774
s S1      0.0000 0.0005 0.2578 0.0001 | 0.9980
u S6      0.0001 0.0000 0.0000 0.0183 | 0.9964
l S4      0.0000 0.0634 0.0000 0.0000 | 1.0000
t S10     0.0000 0.0044 0.0000 0.0000 | 1.0000
          PURITY = 0.840314093211587
          DER = 0.245864821888166
          Number of speakers found = 6
```

Table 3.1: An example of the result table.

3.3.1 Occurrence Matrix and Purity

In the occurrence matrix, the rows correspond to the result labels and the columns correspond to the reference labels. As an example, the value for $S7S1 = 0.2923$ means that percent wise is 29.23% of the speech data assigned to result speaker label $S7$ and at the same time to reference speaker label $S1$.
The column labeled as PIDX contains a purity index which is calculated as:

$$PIDX_i = \frac{max(row_i)}{sum(row_i)} \tag{3.1}$$

This is the purity index for a single result speaker, which is computed as first taking the maximum percentage time this model occurs at the same time as a reference speaker occurs, and then divide by the total percent wise time the model occur.
The PURITY value under the matrix is an averaged purity over all the single model purity index by using the percent wise time the models occur correctly from the real speakers occur. This can be calculated as:

$$PURITY = \sum_i PIDX_i \cdot sum(row_i) = \sum_i \frac{max(row_i)}{sum(row_i)} \cdot sum(row_i) = \sum_i max(row_i) \tag{3.2}$$

From Equation 3.2, we can see the PURITY is the summation of the maximum numbers in every row of the matrix.
Since it is always tried to have more initial models than the number of real speakers, it is possible to have more speakers left than the reference number of speakers. Therefor, the number of speakers found is also an important result which is given under the PURITY value. For example, in the matrix above, there are two more speaker found than the real speaker in the end, which means two models are used for modeling the same speaker.
The reason for the model names $S7$, $S2$, $S1$, $S6$, $S4$ and $S10$ in the result labels is that when two models are merged which is described in the baseline flow chart, one of the models will replace the other model's name. That makes the result model names are not in order and different from the reference speaker IDs. These IDs should not be confused with the HMM states in Chapter2 or the segment notation in Section 3.2

3.3.2 Diarization Error Rate

There exists a problem for the purity score measurement, it does not take the surplus of found speakers over the number of real speakers into consideration. This kind of error is reflected in the number of speakers found. If we end up with: a very large number of speakers found and every found speaker matches a single speech segment. Then the purity score could be as high as 100%.
To account for this weakness, the DER is a performance measurement that gives the speaker error time in system output divided by the total speaker time in the reference. It is defined in the NIST Rich Transcription Evaluations [8].

$$DER = \frac{\sum_{all\ segs}\{dur(seg) \times (\max(N_{Ref}(seg), N_{sys}(seg)) - N_{correct}(seg))\}}{\sum_{all\ seg}\{dur(seg) \times N_{Ref}(seg)\}} \tag{3.3}$$

where the speech data file is divided into contiguous segments at all speaker change points (for both reference and the system) and where, for each segment, seg:

- $dur(seg)$: The duration of seg.
- $N_{Ref}(seg)$: The number of reference speakers speaking in seg. In our case, $N_{Ref}(seg) = 1$, since the data in our system is all speech and there is no overlapping of speakers.
- $N_{Sys}(seg)$: The number of result speakers speaking in seg, in our case, $N_{Sys}(seg) = 1$.
- $N_{correct}(seg)$: the number of reference speakers speaking in seg for whom their matching (mapped) result speakers are also speaking in seg, $N_{correct}(seg) = 1$ *or* 0.

There are three kind of errors which accumulate the DER:

- Miss: speech in reference but test as non-speech in result
- False alarm: Non-speech in reference but test as speech in result
- Speaker-error: Speaker time that is attributed to the wrong speaker

Since the data for our system is all speech, we do not have the "Miss" and "False alarm" errors between speech and non-speech part. Our DER only contains "Speaker-error", it is computed by first finding an optimal one-to-one mapping of the reference label file to the result label file and then obtaining the error as the percentage of time that the result segments assign the wrong speaker ID. The smaller DER, the better performance the system has.
For example, back to Table 3.1, to do the mapping, we find the maximum percentage number of every column (according to the reference speaker IDs, and in this case, that should be four maximum values), then set them all to zero, and the sum of the rest percentage values is the DER. But note that it is possible to find more maximum column values in the same row, as shown in the example, $S2S2 = 0.1493$ and $S2S4 = 0.2040$ are the maximum values for the reference column $S2$ and $S4$, respectively. This indicates that result speaker 2 is a merged model of the two reference speakers, which is also an error for the system. So only the bigger matched part, $S2S4 = 0.2040$ will be regarded as the correct part, which is set to be zero, and the other part will be counted into the DER. In general, the correct one-to-one mapping of the reference speakers in the occurrence matrix, should not only be the maximum value of the column, but also the maximum value of the row, otherwise the value will be added into the DER.

Chapter 4
Test and Discussion

In this chapter, the data for the speaker segmentation system is explained in Section 4.1, and in the following Section 4.2,the results from uniform segmentation and FE algorithm are given and compared, the results by using different parameters in FE algorithm are also listed and analyzed.

4.1 Test Setup

The test data and the development data for threshold selection (explained in Section 3.2.1) is from two different source domains, the meeting and the broadcast news.
Test data for the broadcast domain is from the NIST audio corpus: 940413. The non-speech sections of the file were cut out according to the information in the transcription file for 940413. To reduce computational time, only the first 17 minutes were used. In this part speech from 19 different speakers is present. And the development broadcast domain data for the threshold selecting is also from NIST audio corpus: 940429. The non-speech sections of the file were cut out as the same way we cut out the non-speech part for the test data. The whole data was used to get a proper threshold.
Test data for the meeting domain was recorded by ourselves. Four people, two boys and two girls, talked together for 10.2 minutes. In order to ensure that the file only contained continuous speech in which no other sound exists between two speakers, the recording was made every time after one speaker started speaking and stopped before the speaker finished his or her talking. Overlapping was not present in the data recording and the SNR was around 30dB. The development meeting data for the threshold selection was made the same way as the test meeting data.

4.2 Result and Analysis

In this section, we present and compare the results from the uniform segmentation and the FE algorithm. We also analyze the parameters in the FE algorithm.

4.2.1 Results for Uniform Segmentation

The results of the complete system using the uniform segmentation in the initialization step are given for the meeting domain data in Table 4.1 and the broadcast domain data as shown in Table 4.2.

Table 4.1: Result for speaker segmentation in 4 people speaking meeting domain using uniform segmentation.

Test file	Initial number of models	Purity score	Number of speakers found	DER
Meeting	10	0.7996	5	0.3024
Meeting	16	0.8819	7	0.2371

Table 4.2: Result for speaker segmentation in 19 people speaking broadcast domain using uniform segmentation.

Test file	Initial number of models	Purity score	Number of speakers found	DER
Broadcast	40	0.9397	31	0.2896

From the two tables, we can see that the purity score for the broadcast news domain using uniform segmentation is better than the one with the meeting domain. That could be because that in the broadcast news the speakers are mostly represented a few times but they talk for a long time when present. This makes every initial model rather well fitted to a single speaker. In the meeting domains the same speakers appear many times but only speak for short periods. This would make the amount of speech from every single speaker evenly distributed in all initial models. This is demonstrated in Table 4.3, where the purity scores for the initial models themselves at the very beginning have been computed.

Table 4.3: Purity scores of the initial models for meeting and broadcast domain data.

Test file	Initial number of models	Purity score for the initial models
Meeting	10	0.4151
Meeting	16	0.5407
Broadcast	40	0.7760

But the DERs for both meeting domain data and broadcast data are close to each other, this

is due to different reasons though. As previously mentioned, DER is an accumulation of different kind of errors for an overall performance. If we look at the Occurrence matrices in Table C.1 and C.2, for the results of the meeting domain in Appendix C, it can be seen that the primary reason for the DER is that the labeling of the found speakers largely overlap more than one real speaker. This is because all the initial models were trained on speech from every single real speaker, so chances are higher that two real speakers will get represented by a single found speaker. This is not the problem in the Occurrence matrix from the broadcast domain in Table C.3. There the problem is too many found speakers (31) than real speakers (19).

4.2.2 Results with Different Parameters Using Friends and Enemies Algorithm

Possible changing points selection with different thresholds

In the FE algorithm, the first step is the changing points detection. As explained in Section 3.2.1, after the process of calculating ΔBIC and applying the minimum duration, a threshold is taken into consideration to decrease the number of possible changing points, or in other words, to reduce the number of initial segments which will be used to select friends and enemies groups from. A plot 4.1 is made by development data for the meeting domain to evaluate the method of the threshold setting and is also supposed to find out a possible threshold for the process.

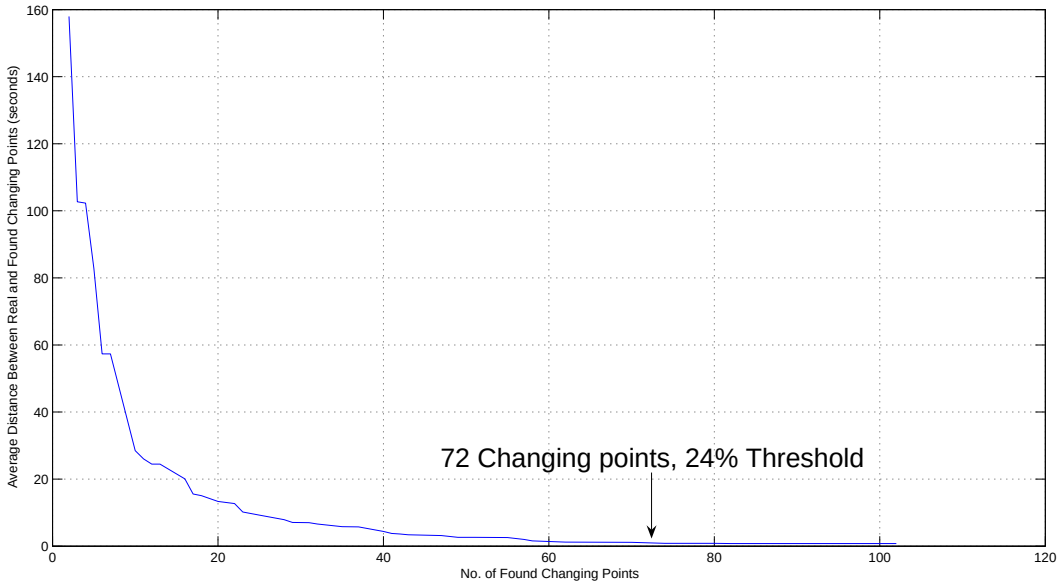

Figure 4.1: Threshold measurement for the meeting development data.

In the Figure 4.1, the x-axis is the number of changing points according to different thresholds, which are from 100% to 0% of the lowest ΔBIC. The y-axis is the average distance from a real speaker change point (RCP_i) to its closest found changing points

(FCP_i), which can be expressed as:

$$y = \frac{\sum_{i=1}^{N_{RCP}} |FCP_i - RCP_i|}{N_{RCP}} \tag{4.1}$$

where N_{RCP} is the number of real changing points.
From the Figure 4.1, it can be seen that the more changing points are found, the smaller average distance is. We note the point where the average distance goes below 0.5 second and uses the corresponding threshold for the further testing. Read out at this point, it gives a 24% threshold with 72 changing points, so it removes many redundant changing points (around 30). Therefor, there should be 73 initial segments which go to the next step.
Using the same idea, another Figure 4.2 for the thresholds of broadcast data is also made with a point marking the proper threshold which should be chosen.

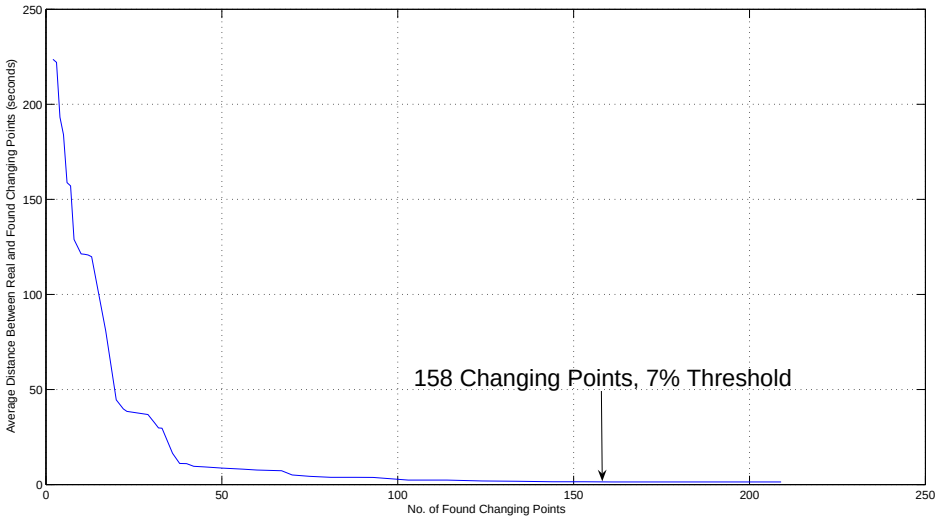

Figure 4.2: Threshold measurement for the broadcast news development data.

From the figure, we can see that for the broadcast data, 158 changing points using a threshold at 7% can be used for the next process.
24% and 7% are regarded as proper thresholds for the meeting domain data and for the broadcast news domain data, respectively. So for the test data, these two values are applied to constrain the number of changing points. The number of changing points for the broadcast domain is 133, and for meeting data, it is 73. Therefor, 134 segments and 74 segments are generated for the later process of FE for the two domain data.

Initial Number of Models Selection and Friends Number Selection

After changing points detection, the initial clusters are generated. This is different from the uniform segmentation: here every initial cluster has different length from each other according to the FE algorithm.

As described in Section 3.2.2, the initial number of models K can be set both beforehand by the users or is obtained when it is running out of available segments. In this project, both of these two methods are used, the number is set as: the recommend values 10 and 16 for the meeting domain, 40 for the broadcast news domain and the returned values when the system runs out of the segments according to different number of friends in a group.
The number of friends in a group F is also tested as a parameter of the FE algorithm, it starts with 2 friends per group and increases the number by 1 friend every time. But note that when the initial number of models are fixed as 10 or 16 for the meeting and 40 for the broadcast news, the number of segments available from the previous changing points detection step is also fixed. These two fixed number constrain the maximum of the possible F, for example, in our meeting data, if 10 is selected as the number of initial models, and there are 74 segments available from the last step, then the maximum F should be $[\frac{74}{10}] = 7$. Same for the 16 initial models case, the maximum should be $[\frac{74}{16}] = 4$, and for the broadcast news data, the maximum F should be $[\frac{134}{28}] = 4$.

Results for Speaker Segmentation using FE algorithm

The results for the speaker segmentation in both meeting and broadcast news domains using FE algorithm by tuning the parameters can be seen in Tables 4.4, 4.5, 4.6 and 4.7. The occurrence matrices are given in Appendix C.

Table 4.4: Results for speaker segmentation in meeting domain with 4 real speakers using FE initialization algorithm, the number of initial models 10 or 16 is set beforehand. For convenience, the results from uniform segmentation is also listed.

Algorithm	No. of initial models	No. of friends	Purity score	No. of speakers found	DER
FE	10	2	0.8403	6	0.2459
FE	10	3	0.8093	5	0.2626
FE	10	4	0.9937	4	0.0063
FE	10	5	0.9913	6	0.0420
FE	10	6	0.9329	5	0.1608
FE	10	7	0.9950	4	0.0050
Uniform	10	None	0.7996	5	0.3024
FE	16	2	0.8753	9	0.2848
FE	16	3	0.9040	7	0.1300
FE	16	4	0.9952	4	0.0048
Uniform	16	None	0.8819	7	0.2371

Table 4.5: Results for speaker segmentation in meeting domain with 4 real speakers using FE initialization algorithm, the number of initial models is returned by the number when running out of the segments.

Algorithm	No. of initial models	No. of friends	Purity score	No. of speakers found	DER
FE	37	2	0.9835	13	0.1482
FE	24	3	0.8639	6	0.3147
FE	18	4	0.9952	4	0.0048
FE	14	5	0.8733	4	0.2256
FE	12	6	0.9805	5	0.1107

Table 4.6: Results for speaker segmentation in broadcast domain with 19 real speakers using FE initialization algorithm, the number of initial models 40 is set beforehand. For convenience, the results from uniform segmentation is also listed.

Algorithm	No. of initial models	No. of friends	Purity score	No. of speakers found	DER
FE	40	2	0.7302	27	0.4362
FE	40	3	0.9537	34	0.2484
Uniform	40	None	0.9397	31	0.2896

Table 4.7: Results for speaker segmentation in broadcast domain with 19 real speakers using FE initialization algorithm, the number of initial models is returned by the number when running out of the segments.

Algorithm	No. of initial models	No. of friends	Purity score	No. of speakers found	DER
FE	66	2	0.9450	45	0.2868
FE	44	3	0.9537	37	0.2482
FE	33	4	0.9167	26	0.2225
FE	26	5	0.8926	24	0.3001

4.2.3 Analysis of the Results

Comparison of Uniform Segmentation and Friends and Enemies Algorithm

Comparing the results for the meeting data using uniform segmentation and FE algorithm, when the initial models are set beforehand as 10 or 16, the purity scores for the FE Algorithm are generally better than the results from uniform segmentation. As can be seen in the result Tables 4.4 and 4.1, using the FE algorithm, the purity score can be as high as 99.50% for 10 initial models, this is a 24.44% relative improvement and 19.54% absolute improvement comparing to the uniform segmentation. The highest purity score for the 16 initial models is 99.52% in FE which is 12.85% relative improvement and 11.33% absolute improvement comparing to the uniform segmentation. And both of these highest purity values are together with an exact speaker number found, which is also better than the uniform segmentation.
The reason for the improvement from the meeting domain could be that using FE algorithm, the errors from the initialization step will not propagate all the way down to the end, and affect the final results. It is more precise and reasonable to decide the initial models using FE, which constrains only one speaker existing in an initial model. If using the uniform segmentation, the initial models are very likely trained by more than one speakers, which gives some initial errors to the system.
From the result of FE algorithm in the broadcast domain, FE does not seem to work better than the uniform segmentation. That could be because some people in the broadcast data speak only once, and all of this speech gets assigned to a single segment by the changing point detection, which means there is no real friend for this segment. But FE algorithm would forcedly assign friends for it, which makes the initial model contain more than one speaker. But in the meeting domain, we normally have people speaking for many times, this kind of problem will less likely happen.
So, FE algorithm is suitable for the meeting domain data, but not well for the broadcast domain data.

Analysis of Different Parameters in FE

Using different friends a group for a beforehand decided number of initial models, also means that it is using different amount of data from the speech file to train the initial models. These data percentages for different friends numbers together with their achieved purity scores can be seen in Table 4.8. From the table, we can see that using more friends a group, alternatively, using a higher percentage of the data from the speech file to train the initial models, a lower DER will be achieved for a better performance. For example, in the meeting domain, when the number of initial models is 10, using 2 friends a group, it will cover only 23% of the whole speech data, to train the initial models; if using 7 friends per group, it will cover 95% of the speech data, to train the initial models, and the 7 friends group can get a much lower DER (0.0050) than the one (0.2459) from 2 friends group. Since the whole speech file will be assigned to the initial models which are also trained by part of the same speech file, general saying, the more information it is using to train the models, the more possibly precise the initial models will be. Though when using 5 and 6 friends a group for 10 initial model meeting domain, there is some DER increase, but still in general, the large-number friends group's (data percentage more than 50%) DER is better than the small-number friends group's DER.

Table 4.8: Data covering percentage from the speech file using different friends numbers.

Test file	No. of initial models	No. of friends	Data percentage	DER
meeting	10	2	23%	0.2459
meeting	10	3	38%	0.2626
meeting	10	4	62%	0.0063
meeting	10	5	64%	0.0420
meeting	10	6	86%	0.1608
meeting	10	7	95%	0.0050
meeting	16	2	39%	0.2848
meeting	16	3	63%	0.1300
meeting	16	4	92%	0.0048
broadcast	40	2	58%	0.4362
broadcast	40	3	89%	0.2484

If the number of initial models is the number returned from the number of models when running out of the friends group, a general good purity score and DER is achieved. This also proves that the percentage of the data used for training the initial models can effect the result. But note that if the number of friends is very small, then the returned initial model numbers is sometimes much larger than the real number of speakers, the speakers found will be far away from the real speaker numbers. For example, in table 4.5, when using 2 friends a group, though the purity score is good enough, the number of speakers found is 13 which is much larger than the real number 4, due to the reason that the number of initial models are too large which is 37 in this case. Therefore, when using the FE, both the number of friends and the initial models should be taken into consideration to get good results both in the purity score and in the number of speakers found.
An interesting thing from the result which is worth to be pointed out is: when using 4 friends per group, we get very good performance for all the data. The number of friends together with the number of initial models are a bit tricky to adjust.

Chapter 5 Conclusion

In this project we have been dealing with the speaker segmentation problem using a proposed non-uniform cluster initialization algorithm: Friends and Enemies. A problem statement was created, which concerned whether or not the FE algorithm can improve the system performance comparing to the uniform segmentation for both the meeting and broadcast news domains. And it also concerned testing different parameters in the FE algorithm.

The speaker segmentation in this report shares the most commonly used baseline system for agglomerative clustering. FE algorithm is applied to make the initial models in a more reasonable way comparing to the uniform initialization. It works in two steps: First it finds likely speaker change points in the data. Secondly, it groups the friend segments (testing different numbers of friends) together, and creates the clusters for the initial models (testing different desired numbers of initial models). These clusters are also called enemy groups.

From the results, we can conclude that FE algorithm works much better in the meeting domain than the uniform segmentation, that is because FE initialization constrains only one speaker in every initial model and therefore reduces the clustering errors at the initialization stage which is hard to correct for the final result if using uniform segmentation. But FE algorithm has its limitation as, it does not work better on the broadcast domain data than the uniform segmentation. That is because the speakers in the broadcast domain are very often either talking a few times with a long time speech or speaking only one time with a short speech, which make it difficult to find proper friends for the segment.

In FE algorithm, the data percentage of the whole speech data used to train the initial models should be large, then a general better result could be obtained.

The parameters in the FE algorithm, like the number of friend and the number of initial models, are sensitive to adjust, further work could be done to improve the robustness of the algorithm.

To solve the problems of FE algorithm in the broadcast news domain, when a single large segment do not have any real friend segments, an approach to limit the maximum size of a segment after changing points detection could possible improve the FE effectiveness in the broadcast news domain. Another approach could be not to use a fixed number of friends in every cluster, but instead, using a cluster data size criteria, where to stop adding friends to the cluster when the cluster size exceeds some given amount of data.

Bibliography

[1] N. L. A. Dempster and D. Rubin.
Maximum likelihood from incomplete data via the em algorithm.
J. Royal Stat. Soc., ASSP28:1–38, 1977.

[2] J. Ajera and C. Wooters.
A robust speaker clustring algorithm.
Proceedings of the IEEE automatic speech recognition understanding work-shop (ASRU).

[3] B. P. C. Wooters, J. Fung and X. Anguera.
Toward robust speaker segmentation: The icsi-sri fall 2004 diarization system.
Proc. Fall 2004 Rich Transcription Work-shop (RT-04), November 2004.

[4] S. S. Chen and P. Gopalakrishnan.
Speaker, environment and channel change detection and clustering via the bayesian infromation criteria.
IBM T. J. Watson Reasearch Center, 1998.

[5] L. B. J. F. B. D. Moraru, S. Meignier and I. Magrin-Chagnolleau.
The elisa consortium approaches in speaker segmentation during the nist 2002 speaker recognition evaluation.
Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.

[6] Davis and Mermelstein.
Comparison of parametric representations for monosyllable word recognition in continuously spoken sentences.
IEEE Transactions on Acoustic, Speech and Signal Processing, 1980.

[7] L. B. et al.
A maximization technique occurring in the statistical analysis of probabilistic functions of markov chain.
Ann. Math Stat, 41:164–171, 1970.

[8] http://www.nist.gov/speech/tests/rt/rt2005/spring/rt05s-meeting-eval-plan V1.pdf.

[9] J. H. H. John R. Deller, Jr. and J. G. Proakis.
Discrete-Time Processing of Speech Signals.
Macmillan Publishing Co., 1993.
ISBN 0-7803-5386-2.

[10] F. B. M. Ben, M. Betser and G. Gravier.
Speaker diarization using bottom-up clustering based on a parameter-derived distance betwwen adapted gmms.
Proc. Int. Conf. Spoken Language Processing, 2004.

[11] A. Martin and M. Przybocki.
Speaker recognition in a multi-speaker environment.

in Proc. Eur. Conf. Speech Commun. Technol., 2:787–790, September 2001.

[12] G. McLachlan.
Mixture models.
New York: Marcel Dekker, 1988.

[13] X. A. Miró.
Robust speaker diarization for meetings.
PhD thesis, Universitat Politècnica de Catalunya, October 2006.

[14] S. A. S. Morten Højfeldt Rasmussen and M. T. Svendsen.
Signal and speaker segmentation - using hmms.
Technical report, Department of Communication Technology, Aalborg University, 2004.

[15] M. J. F. G. R. Sinha, S. E. Tranter and P. C. Woodland.
The cambridge university march 2005 speaker diarization system.
Proc. Eur. Conf. Speech Commun. Technol., pages 2437–2440, September 2005.

[16] L. R. Rabiner.
A tutorial on hidden markov models and selected applications in speech recognition.
Processings of the IEEE, 77(2):257–286, Febrary 1989.

[17] D. A. Reynolds and R. C. Rose.
Robust text-independent speaker identification using gaussian mixture speaker models.
IEEE Transactions on speech and audio processing, 3(1), 1995.

[18] D. A. Reynolds and P. Torres-Carrasquillo.
The mit lincoln laboratory rt-04f diarization systems: Applications to broadcast audio and telephone conversations.
Proc. Fall 2004 Rich Transcription Work-shop (RT-04), November 2004.

[19] C. F. J. F. B. S. Meignier, D. Moraru and L. Besacier.
Step-by-step and integrated approaches in broadcast news speaker diarization.
Comput. Speech Lang, (20):303–330, September 2005.

[20] T. H. D. K. G. M. J. O. D. O. D. P. V. V. Steve Young, Gunnar Evermann and P. Woodland.
The HTK Book (for HTK version 3.2.1).
Cambridge University Engineering Department, 2002.

[21] A. T. P. W. T.Hain, S.E.Johnson and S. Young.
Segment generation and clustering in the htk broadcast news transcription system.
DARPA Broadcast News Transcription System and Understanding Workshop, January 1998.

[22] S. E. Tranter and D. A. Reynolds.
An overview of automatic speaker diarization systems.
IEEE Transactions on audio, speech, and language processing, 14(5), September 2006.

[23] A. Vandecatseye and J.-P. Martens.
A fast, accurate and stream based speaker segmentation and clustering algorithm.
Eurospeech, January 2003.

[24] B. P. X. Anguera, C. Wooters and M. Aguiló.

Robust speaker segmentation for meetings: The icsi-sri spring 2005 diarization system.
Proc. Machine Learning for Multimodal Interaction Workshop (MLMI), Edinburgh, U.K., 2005.

[25] C. W. X. Anguera and J. Hernando.
Automatic cluster complexity and quantity selection: Towards robust speaker diarization.
Speaker Odyssey, June 2006.

[26] C. W. Xavier Anguera and J. Hernando.
Friends and enemies: A novel initialization for speaker diarization.
Interspeech, 2006.

[27] P. N. Y. Moh and J. C. Junqua.
Toward domain independent clustering.
Proc. IEEE Int. Conf. Acoust., Speech, Signal Process, 2:85–88, 2003.

Appendix

Program Conversion

A

The baseline system for speaker segmentation was originally made to work on Linux Operative Systems (OS). To be able to continue work with it, a conversion to Windows and DOS was necessary. As the system was coded in multiple program languages, different changes had to be made.

A.1 Perl and HTK programs

Firstly all of the HTK executables had to be recompiled and built for DOS. Secondly the Pearl script code that controls the program flow and calls the HTK executables had to change the call syntax. This is because all the HTK calls is done through the command console of the OS and there are some minor but significant differences between Linux and DOS.

A.2 C++ problems

A program called read_file.exe is used as a memory copying tool and is coded in C++. Major problems occurred with this program as it was originally coded following the C99 standard and used functionality that is not supported by the Microsoft Visual DOS compiler that only supports C90. To circumvent this problem a free Windows C++ compiler and builder was downloaded from the internet called Dev-C++. This compiler is based on the GNU compiler for Linux. Still, some memory call operations had to be modified to satisfy DOS.

Appendix B

Description of Sourcecode

The aim of the computer programs developed for this project are to utilize the friends and enemy theory in computing a HTKTool compatible label file that contains information on the initial grouping for the speaker segmentation. All programs have been made in LabVIEW for Windows XP operative system (OS). The program files (called vi's) require LabVIEW installed to be executed, and are all located in a library file called "friendsAndEnemies.llb". The following document describes in short the major vi's used in the different steps of friends and enemies, these all have a file name ending on "_main.vi". The rest of the vi's are either sub-vi's or used for testing. The programs make usage of the HTKTools: HCOPY, HINIT, HREST, HPARSE and HVITE, which all need to be compiled and built for Windows OS.

ComputeBIC_main.vi

Description: This program computes the ΔBIC values for a MFCC speech data file and saves them.
Input:

- Sampling Frequency: The sampling frequency of the MFCC file. Not necessary the original sound files.

- File End: If the total number of samples of the MFCC file is known, then it can be written here. If not, a guessed number which is bigger than the total number of samples will be used.

- Load Path of Speech File: File path of the MFCC file.

- Path for HTKTools directory: Directory path to the HTKTool files.

- par: The initial parameters of the HMM's. See HTK documentation [20].

- Changing Point Parameters: Set the time boundaries (windowSize) for the a, b and c models and the time step (scrollsize) between the BIC computations.

Output:

- Save Path of BIC Save File: File path of the file where the BIC values are saved.

FindChangingPoints_main.vi

Description: This code looks through a *BIC* values save file and computes a label file based on the minimum segment duration.
Input:

- Sampling Frequency: The sampling frequency of the MFCC file. Not necessary the original sound files.
- Changing Point Parameters: Set the minimum segment duration (minSegDuration) between two changing points.
- Load Path for *BIC* Save File: File path to the file containing the *BIC* values.

Output:

- Save Path for Changing Point Label File: File path of the label file where segments between the found changing points are defined.

friendsAndEnemies_main.vi

Description: The primary Friends And Enemies computational tool. Some indicators show data progress and possible errors. Others show the memory storage arrays.
Input:

- Load Path of Speech File: File path of the MFCC file.
- Path for HTKTools directory: Directory path to the HTKTool files.
- Load Path for Changing Point Label File: File path to the file containing the changing points.
- No. of Friends Besides Initial Group: Every initial cluster will contain this number plus one segments.
- No. of Groups Besides S0: Gives the number of initial clusters besides the "rest" cluster S0.
- Load Storage File?: If false, the program will start out by computing the probability values of every segment against a global model of the whole MFCC file. It will then save these values in a file with name and path defined by the controller "Save/Load Path for Global Model Contra Segment Storage File". if true, the program will instead load the probability values from the file. The problem will continue normal program flow in either case except if the file with the probability values does not exist naturally.
- Save/Load Path for Global Model Contra Segment Storage File: See previous item.
- par: The initial parameters of the HMM's. See HTK documentation.
- Minimum duration: Parameter used in the definitions of the word net files made by HPARSE. See HTK documentation [20].

Output:

- Save Path for Initial Segment Label File: Path to the file where the initial clusters are defined and ready to be used in the speaker segmentation operations.

Appendix

C Results of Occurence Matrices

Table C.1: Test results when using uniform segmentation for meeting data with 10 initial models

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S3   0.0980 0.0000 0.0000 0.0000 | 0.9998
  e S5   0.1918 0.0005 0.2659 0.0001 | 0.5802
  s S7   0.0000 0.2091 0.0000 0.0000 | 0.9998
  u S6   0.0002 0.0050 0.0004 0.2225 | 0.9753
  l S2   0.0023 0.0041 0.0000 0.0000 | 0.6343
         PURITY = 0.799593449287717
         DER = 0.302470451304077
         Number of speakers found = 5
```

Table C.2: Test results when using uniform segmentation for meeting data with 16 initial models

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S5   0.2796 0.0002 0.0010 0.0000 | 0.9957
  e S2   0.0000 0.2180 0.0000 0.1161 | 0.6524
  s S7   0.0002 0.0005 0.2653 0.0000 | 0.9973
  u S8   0.0126 0.0000 0.0000 0.0000 | 1.0000
  l S9   0.0000 0.0000 0.0000 0.0334 | 0.9995
  t S12  0.0000 0.0000 0.0000 0.0268 | 1.0000
    S15  0.0000 0.0000 0.0000 0.0463 | 0.9996
         PURITY = 0.881870789003459
         DER = 0.237160046556614
         Number of speakers found = 7
```

Table C.3: Test results when using uniform segmentation for broadcast news data with 40 initial models

	R S1	e S2	f S3	e S4	r S5	e S6	n S7	c S8	e S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	PIDX
R	S1	0.0027	0.0206	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8787
e	S2	0.0121	0.0016	0.0136	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4998
s	S3	0.0098	0.0153	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.6099
u	S4	0.0000	0.0273	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
l	S5	0.0000	0.0004	0.0000	0.0434	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9913
t	S6	0.0000	0.0000	0.0000	0.0003	0.0173	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9851
	S7	0.0000	0.0000	0.0000	0.0137	0.0001	0.0088	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.6077
	S8	0.0000	0.0026	0.0000	0.0073	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7369
	S13	0.0000	0.1277	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0004	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9958
	S9	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0625	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9949
	S10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0061	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5996
	S14	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0292	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9979
	S16	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0133	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S17	0.0000	0.0011	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0096	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8963
	S18	0.0000	0.0238	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0067	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7805
	S19	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0456	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9965
	S20	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0601	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S23	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0002	0.0000	0.0199	0.0000	0.0000	0.0000	0.0000	0.0000	0.9910
	S24	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0240	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S25	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.0255	0.0000	0.0000	0.0000	0.0000	0.9872
	S28	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0635	0.0000	0.0000	0.0000	0.0000	0.0000	0.9997
	S27	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0188	0.0000	0.0000	0.0000	0.0000	0.9984
	S29	0.0000	0.0046	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0035	0.0000	0.0000	0.0010	0.0000	0.0000	0.5077
	S30	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0770	0.0000	0.0000	1.0000
	S31	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.0135	0.0000	0.9767
	S33	0.0000	0.0177	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.0000	0.0000	0.9810
	S34	0.0000	0.0363	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S35	0.0155	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9872
	S36	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S37	0.0000	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0509	0.0007	0.9806
	S40	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0240	1.0000

PURITY = 0.939720518749372
DER = 0.289564692872223
Number of speakers found: 31

Table C.4: Test results when using FE for meeting data with 10 initial models and 2 friends a group

	R S1	e S2	f S3	e S4	PIDX
R	S7	0.2923	0.0011	0.0084	0.0002	0.9678
e	S2	0.0000	0.1493	0.0001	0.2040	0.5774
s	S1	0.0000	0.0005	0.2578	0.0001	0.9980
u	S6	0.0001	0.0000	0.0000	0.0183	0.9964
l	S4	0.0000	0.0634	0.0000	0.0000	1.0000
t	S10	0.0000	0.0044	0.0000	0.0000	1.0000

PURITY = 0.840314093211587
DER = 0.245864821888166
Number of speakers found = 6

Table C.5: Test results when using FE for meeting data with 10 initial models and 3 friends a group

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S9   0.2534 0.0002 0.0001 0.0000 | 0.9986
  e S4   0.0000 0.2179 0.0000 0.1872 | 0.5378
  s S1   0.0023 0.0006 0.2661 0.0001 | 0.9887
  u S7   0.0000 0.0000 0.0000 0.0354 | 0.9991
  l S8   0.0366 0.0000 0.0000 0.0000 | 1.0000
         PURITY = 0.809314601399977
         DER = 0.262602252422091
         Number of speakers found = 5
```

Table C.6: Test results when using FE for meeting data with 10 initial models and 4 friends a group

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S9   0.2900 0.0004 0.0010 0.0002 | 0.9947
  e S6   0.0000 0.2160 0.0000 0.0000 | 1.0000
  s S1   0.0023 0.0006 0.2653 0.0001 | 0.9889
  u S8   0.0000 0.0018 0.0000 0.2224 | 0.9920
         PURITY = 0.993688628055278
         DER = 0.006311371944722
         Number of speakers found = 4
```

Table C.7: Test results when using FE for meeting data with 10 initial models and 5 friends a group

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S5   0.2709 0.0002 0.0020 0.0002 | 0.9909
  e S6   0.0000 0.2054 0.0000 0.0046 | 0.9781
  s S4   0.0000 0.0120 0.0000 0.0000 | 1.0000
  u S1   0.0001 0.0004 0.2639 0.0000 | 0.9979
  l S10  0.0000 0.0007 0.0000 0.2177 | 0.9967
  t S3   0.0213 0.0000 0.0004 0.0000 | 0.9819
         PURITY = 0.991262438320683
         DER = 0.042015704660579
         Number of speakers found = 6
```

Table C.8: Test results when using FE for meeting data with 10 initial models and 6 friends a group

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S7   0.2923 0.0002 0.0027 0.0000 | 0.9902
  e S4   0.0000 0.1236 0.0000 0.0628 | 0.6632
  s S1   0.0000 0.0006 0.2636 0.0001 | 0.9971
  u S10  0.0000 0.0006 0.0000 0.1597 | 0.9959
  l S2   0.0000 0.0937 0.0000 0.0000 | 1.0000
         PURITY = 0.932919132473238
         DER = 0.160800642612416
         Number of speakers found = 5
```

Table C.9: Test results when using FE for meeting data with 10 initial models and 7 friends a group

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S3   0.2922 0.0002 0.0027 0.0000 | 0.9899
  e S6   0.0000 0.2169 0.0000 0.0001 | 0.9994
  s S9   0.0001 0.0008 0.2635 0.0001 | 0.9960
  u S4   0.0000 0.0008 0.0001 0.2224 | 0.9961
         PURITY = 0.994967295618105
         DER = 0.005032704381895
         Number of speakers found = 4
```

Table C.10: Test results when using FE for meeting data with 16 initial models and 2 friends a group

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S12  0.2837 0.0010 0.0072 0.0002 | 0.9711
  e S2   0.0000 0.1040 0.0002 0.1727 | 0.6236
  s S1   0.0000 0.0007 0.2588 0.0001 | 0.9970
  u S6   0.0001 0.0000 0.0000 0.0182 | 0.9964
  l S13  0.0000 0.0775 0.0000 0.0111 | 0.8743
  t S16  0.0000 0.0001 0.0000 0.0203 | 0.9968
    S10  0.0000 0.0043 0.0000 0.0000 | 1.0000
    S8   0.0000 0.0312 0.0000 0.0000 | 1.0000
    S3   0.0085 0.0000 0.0000 0.0000 | 1.0000
         PURITY = 0.875280733102736
         DER = 0.284782216684972
         Number of speakers found = 9
```

Table C.11: Test results when using FE for meeting data with 16 initial models and 3 friends a group

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S5   0.2699 0.0002 0.0021 0.0000 | 0.9914
  e S4   0.0000 0.2180 0.0000 0.0929 | 0.7010
  s S1   0.0000 0.0002 0.2641 0.0001 | 0.9986
  u S7   0.0000 0.0002 0.0000 0.1180 | 0.9972
  l S8   0.0127 0.0000 0.0000 0.0000 | 1.0000
  t S3   0.0096 0.0000 0.0000 0.0000 | 1.0000
    S12  0.0000 0.0000 0.0000 0.0116 | 1.0000
         PURITY = 0.903985180570810
         DER = 0.129997868887395
         Number of speakers found = 7
```

Table C.12: Test results when using FE for meeting data with 16 initial models and 4 friends a group

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S9   0.2923 0.0004 0.0019 0.0002 | 0.9917
  e S6   0.0000 0.2166 0.0000 0.0000 | 0.9998
  s S5   0.0000 0.0001 0.2640 0.0001 | 0.9993
  u S8   0.0000 0.0018 0.0004 0.2224 | 0.9904
         PURITY = 0.995180406878576
         DER = 0.004819593121424
         Number of speakers found = 4
```

Table C.13: Test results when using FE for meeting data with 2 friends a group and 37 initial models

```
         R      e      f      e
         S1     S2     S3     S4     | PIDX
  R S7   0.2736 0.0002 0.0010 0.0002 | 0.9949
  e S36  0.0000 0.0129 0.0002 0.1877 | 0.9347
  s S2   0.0000 0.0204 0.0000 0.0000 | 1.0000
  u S4   0.0000 0.1255 0.0000 0.0001 | 0.9992
  l S23  0.0008 0.0010 0.2651 0.0000 | 0.9930
  t S6   0.0000 0.0000 0.0000 0.0089 | 0.9945
    S27  0.0000 0.0514 0.0000 0.0000 | 1.0000
    S37  0.0000 0.0074 0.0000 0.0000 | 1.0000
    S3   0.0099 0.0000 0.0000 0.0000 | 1.0000
    S35  0.0000 0.0000 0.0000 0.0084 | 1.0000
    S33  0.0000 0.0000 0.0000 0.0120 | 1.0000
    S21  0.0080 0.0000 0.0000 0.0000 | 1.0000
    S29  0.0000 0.0000 0.0000 0.0052 | 0.9969
         PURITY = 0.983459287552663
         DER = 0.148161505549089
         Number of speakers found = 13
```

Table C.14: Test results when using FE for meeting data with 3 friends a group and 24 initial models

```
        R      e      f      e
        S1     S2     S3     S4     | PIDX
R S9    0.2557 0.0002 0.0014 0.0002 | 0.9929
e S4    0.0000 0.1652 0.0004 0.1329 | 0.5534
s S3    0.0001 0.0005 0.2644 0.0001 | 0.9978
u S7    0.0000 0.0003 0.0001 0.0894 | 0.9958
l S6    0.0000 0.0526 0.0000 0.0000 | 1.0000
t S8    0.0366 0.0000 0.0000 0.0000 | 1.0000
        PURITY = 0.863871084080589
        DER = 0.314716152194226
        Number of speakers found = 6
```

Table C.15: Test results when using FE for meeting data with 4 friends a group and 18 initial models

```
        R      e      f      e
        S1     S2     S3     S4     | PIDX
R S9    0.2923 0.0004 0.0019 0.0002 | 0.9917
e S6    0.0000 0.2166 0.0000 0.0000 | 0.9998
s S5    0.0000 0.0001 0.2640 0.0001 | 0.9993
u S8    0.0000 0.0018 0.0004 0.2224 | 0.9904
        PURITY = 0.995180406878576
        DER = 0.004819593121424
        Number of speakers found = 4
```

Table C.16: Test results when using FE for meeting data with 5 friends a group and 14 initial models

```
        R      e      f      e
        S1     S2     S3     S4     | PIDX
R S5    0.2922 0.0002 0.0021 0.0001 | 0.9920
e S4    0.0000 0.2184 0.0004 0.1235 | 0.6380
s S3    0.0001 0.0001 0.2638 0.0001 | 0.9990
u S8    0.0000 0.0001 0.0000 0.0989 | 0.9988
        PURITY = 0.873346338584613
        DER = 0.225602858969525
        Number of speakers found = 4
```

Table C.17: Test results when using FE for meeting data with 6 friends a group and 12 initial models

```
          R       e       f       e
          S1      S2      S3      S4      | PIDX
  R S7    0.2922  0.0005  0.0029  0.0002  | 0.9879
  e S4    0.0000  0.0147  0.0003  0.2224  | 0.9369
  s S6    0.0000  0.0912  0.0000  0.0000  | 1.0000
  u S5    0.0001  0.0008  0.2631  0.0001  | 0.9965
  l S2    0.0000  0.1116  0.0000  0.0000  | 1.0000
          PURITY = 0.980524909427714
          DER = 0.110670316880051
          Number of speakers found = 5
```

Table C.18: Test results when using FE for broadcast data with 2 friends a group and 40 initial models

```
        R       e       f       e       r       e       n       c       e
        S1      S2      S3      S4      S5      S6      S7      S8      S9      S10     S11     S12     S13     S14     S15     S16     S17     S18     S19     | PIDX
R S38   0.0139  0.0002  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.9837
e S3    0.0000  0.0219  0.0138  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.6144
s S30   0.0044  0.0052  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.5389
u S7    0.0007  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0040  0.0001  0.0000  0.0000  | 0.8347
l S19   0.0055  0.0110  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.6653
t S31   0.0000  0.1732  0.0000  0.0000  0.0000  0.0000  0.0676  0.0000  0.0001  0.0001  0.0503  0.0001  0.0003  0.0000  0.0000  0.0005  0.0000  0.0338  0.0000  | 0.5312
  S12   0.0000  0.0020  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 1.0000
  S14   0.0000  0.0105  0.0000  0.0006  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.9499
  S32   0.0000  0.0004  0.0000  0.0641  0.0003  0.0002  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0029  0.0000  | 0.9442
  S37   0.0000  0.0000  0.0000  0.0000  0.0171  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 1.0000
  S5    0.0000  0.0028  0.0000  0.0000  0.0000  0.0086  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.7518
  S6    0.0000  0.0029  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 1.0000
  S16   0.0000  0.0058  0.0000  0.0000  0.0000  0.0000  0.0008  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.8775
  S35   0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0002  0.0093  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0133  0.0000  0.0000  | 0.5825
  S10   0.0000  0.0008  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0168  0.0000  0.0000  0.0000  0.0000  | 0.9527
  S9    0.0000  0.0001  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0295  0.0000  0.0000  0.0601  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.6702
  S39   0.0000  0.0035  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0002  0.0001  0.0020  0.0000  0.0000  0.0000  0.0000  | 0.6073
  S11   0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0228  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 1.0000
  S4    0.0000  0.0249  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0022  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.9204
  S1    0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0282  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 1.0000
  S34   0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0730  0.0003  0.0001  0.0000  0.0000  0.0000  0.0000  | 0.9948
  S18   0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0159  0.0000  0.0000  0.0000  0.0000  0.0000  | 1.0000
  S33   0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0092  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.9957
  S13   0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0096  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 1.0000
  S15   0.0000  0.0005  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0741  0.0000  0.0142  0.0247  | 0.6533
  S36   0.0000  0.0136  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 1.0000
  S17   0.0247  0.0001  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  | 0.9967
        PURITY = 0.730159847190108
        DER = 0.436181763345732
        Number of speakers found: 27
```

Table C.19: Test results when using FE for broadcast data with 3 friends a group and 40 initial models

	R e f e r e n c e																			
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	PIDX
R S37	0.0027	0.0013	0.0027	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4093
e S22	0.0000	0.0216	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
s S40	0.0020	0.0000	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8222
u S4	0.0000	0.0201	0.0020	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0010	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8728
l S26	0.0135	0.0036	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7896
t S9	0.0024	0.0033	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5773
S39	0.0059	0.0097	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.6216
S34	0.0000	0.0110	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S20	0.0000	0.0068	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0004	0.0000	0.0000	0.0000	0.9431
S32	0.0000	0.0122	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S24	0.0000	0.1345	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9990
S14	0.0000	0.0099	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0002	0.0011	0.0000	0.0000	0.0000	0.0000	0.0000	0.8704
S6	0.0000	0.0001	0.0000	0.0647	0.0003	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9898
S17	0.0000	0.0022	0.0000	0.0000	0.0170	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.8787
S11	0.0000	0.0100	0.0000	0.0000	0.0000	0.0086	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5373
S15	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0681	0.0004	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9930
S31	0.0129	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0089	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5898
S8	0.0000	0.0005	0.0000	0.0000	0.0000	0.0000	0.0005	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0781	0.0002	0.0000	0.0000	0.9854
S10	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9976
S2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0228	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S36	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0515	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9967
S18	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0601	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0255	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S38	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0853	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.9968
S3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0150	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0049	0.0000	0.0000	0.0000	0.0000	0.0000	0.9959
S19	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0139	0.0000	0.0000	0.0000	1.0000
S30	0.0000	0.0165	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9886
S25	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0133	0.0000	0.0000	1.0000
S33	0.0000	0.0158	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S7	0.0098	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S21	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0509	0.0004	0.9900
S35	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0243	1.0000

PURITY = 0.953734794410375

DER = 0.248356288328139

Number of speakers found: 34

Table C.20: Test results when using FE for broadcast data with 2 friends a group and 66 initial models

	R	e	f	e	r	e	n	c	e												
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19		PIDX
R	S38	0.0112	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9991
e	S59	0.0000	0.0236	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
s	S64	0.0000	0.0006	0.0068	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9187
u	S65	0.0051	0.0000	0.0070	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5748
l	S49	0.0009	0.0010	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5217
t	S44	0.0123	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S19	0.0002	0.0054	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9709
	S52	0.0000	0.0135	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S45	0.0000	0.1301	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.0000	0.0000	0.0000	0.9958
	S12	0.0000	0.0013	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0018	0.0000	0.0000	0.0000	0.0000	0.5723
	S15	0.0000	0.0105	0.0000	0.0006	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9499
	S48	0.0000	0.0000	0.0000	0.0599	0.0003	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9917
	S37	0.0000	0.0000	0.0000	0.0000	0.0171	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0086	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S50	0.0000	0.0001	0.0000	0.0043	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0015	0.0000	0.0000	0.0000	0.0000	0.0000	0.7228
	S3	0.0000	0.0033	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S6	0.0000	0.0025	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S10	0.0000	0.0056	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9982
	S62	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0685	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9953
	S34	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9934
	S16	0.0000	0.0044	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0062	0.0000	0.0000	0.0000	0.0000	0.0000	0.5849
	S9	0.0000	0.0011	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0094	0.0000	0.0000	0.0000	0.0000	0.8977
	S8	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0601	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.6700
	S39	0.0000	0.0024	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0028	0.0000	0.0000	0.0000	0.0000	0.5360
	S11	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0228	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S41	0.0000	0.0109	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.9811
	S4	0.0000	0.0195	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0007	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9643
	S56	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0518	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9967
	S1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0250	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S54	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0795	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9997
	S57	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.0085	0.0000	0.0000	0.0000	0.0000	0.0000	0.9691
	S32	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0093	0.0000	0.0000	0.0000	0.0000	0.0000	0.9957
	S17	0.0000	0.0024	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0047	0.0000	0.0000	0.0000	0.0000	0.6629
	S13	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0064	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S14	0.0000	0.0005	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0741	0.0000	0.0000	0.0000	0.9929
	S7	0.0000	0.0004	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0040	0.0002	0.0000	0.0000	0.8884
	S42	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0133	0.0000	0.0000	0.0000	1.0000
	S66	0.0000	0.0093	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S47	0.0000	0.0104	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S35	0.0000	0.0052	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S51	0.0000	0.0153	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S61	0.0195	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9964
	S31	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0509	0.0006	0.9862
	S36	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0018	1.0000
	S53	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0222	1.0000

PURITY = 0.9450085452900037
DER = 0.286810093495526
Number of speakers found: 45

Table C.21: Test results when using FE for broadcast data with 3 friends a group and 44 initial models

	R S1	e S2	f S3	e S4	r S5	e S6	n S7	c S8	e S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	PIDX
R S37	0.0027	0.0013	0.0027	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4093
e S22	0.0000	0.0216	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
s S40	0.0020	0.0000	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8222
u S4	0.0000	0.0201	0.0020	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0010	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8728
l S26	0.0135	0.0036	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7896
t S9	0.0024	0.0033	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5773
S39	0.0059	0.0097	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.6216
S34	0.0000	0.0110	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S20	0.0000	0.0068	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0004	0.0000	0.0000	0.0000	0.9431
S32	0.0000	0.0127	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S24	0.0000	0.1347	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9981
S14	0.0000	0.0097	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0002	0.0011	0.0000	0.0000	0.0000	0.0000	0.0000	0.8761
S6	0.0000	0.0001	0.0000	0.0647	0.0003	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9898
S17	0.0000	0.0034	0.0000	0.0000	0.0170	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.8272
S11	0.0000	0.0037	0.0000	0.0000	0.0000	0.0086	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7019
S42	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0678	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9956
S31	0.0114	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5551
S8	0.0000	0.0005	0.0000	0.0000	0.0000	0.0000	0.0007	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0781	0.0002	0.0000	0.0000	0.9824
S10	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9976
S2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0228	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S36	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0515	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9967
S18	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0601	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0255	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S38	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0805	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.9984
S43	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0050	0.0037	0.0000	0.0000	0.0000	0.0000	0.0000	0.5685
S3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0115	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S12	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0049	0.0000	0.0000	0.0000	0.0000	0.9959
S19	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0139	0.0000	0.0000	0.0000	0.0000	1.0000
S30	0.0000	0.0120	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S25	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0133	0.0000	0.0000	1.0000
S41	0.0000	0.0145	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S33	0.0000	0.0104	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S44	0.0014	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S7	0.0098	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S21	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0509	0.0004	0.9900
S35	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0243	1.0000

PURITY = 0.953744847692772
DER = 0.248185382527395
Number of speakers found: 37

Table C.22: Test results when using FE for broadcast data with 4 friends a group and 33 initial models

	R S1	e S2	f S3	e S4	r S5	e S6	n S7	c S8	e S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	PIDX
R S32	0.0025	0.0001	0.0000	0.0000	0.0000	0.0000	0.0685	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0007	0.9510
e S8	0.0072	0.0213	0.0000	0.0000	0.0000	0.0013	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7147
s S26	0.0000	0.0003	0.0118	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9735
u S4	0.0000	0.0216	0.0020	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0007	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8893
l S17	0.0081	0.0014	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0058	0.0000	0.0000	0.0000	0.0000	0.0000	0.5260
t S22	0.0067	0.0093	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5829
S10	0.0000	0.0068	0.0000	0.0000	0.0171	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7160
S25	0.0000	0.1592	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000	0.0005	0.0000	0.0000	0.0000	0.0000	0.9955
S13	0.0000	0.0010	0.0000	0.0647	0.0003	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9759
S5	0.0000	0.0045	0.0000	0.0000	0.0000	0.0073	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.6205
S27	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0091	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9934
S23	0.0000	0.0043	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S11	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0133	0.0000	0.0000	0.6879
S2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0228	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S28	0.0000	0.0115	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0022	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8382
S29	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0496	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9966
S21	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0601	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0255	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S33	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0855	0.0000	0.0000	0.0000	0.0000	0.0000	0.9996
S6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0197	0.0000	0.0000	0.0000	0.0000	0.0000	0.9964
S3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0188	0.0000	0.0000	0.0000	0.0000	0.9995
S18	0.0000	0.0005	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0781	0.0002	0.0000	0.0208	0.7846
S24	0.0000	0.0095	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S30	0.0000	0.0281	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
S15	0.0247	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9984
S20	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0509	0.0032	0.9393

PURITY = 0.916668342213733
DER = 0.222549512415804
Number of speakers found: 26

Table C.23: Test results when using FE for broadcast data with 5 friends a group and 26 initial models

	R	e	f	e	r	e	n	c	e											
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	PIDX
R	S24	0.0027	0.0068	0.0138	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5931
e	S7	0.0029	0.0196	0.0000	0.0000	0.0000	0.0086	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.6303
s	S16	0.0128	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9992
u	S26	0.0061	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
l	S14	0.0000	0.0134	0.0000	0.0000	0.0171	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5612
t	S11	0.0000	0.0165	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S20	0.0000	0.1499	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9985
	S12	0.0000	0.0010	0.0000	0.0647	0.0003	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9778
	S18	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0686	0.0004	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9936
	S17	0.0000	0.0018	0.0000	0.0000	0.0000	0.0000	0.0001	0.0090	0.0000	0.0000	0.0000	0.0602	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.8471
	S8	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0295	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0133	0.0000	0.0000	0.6880
	S2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0228	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S25	0.0000	0.0447	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.9909
	S3	0.0000	0.0234	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0021	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9172
	S22	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0225	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9996
	S21	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0278	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9975
	S1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0255	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
	S19	0.0000	0.0012	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0214	0.0058	0.0000	0.0000	0.0000	0.0000	0.0000	0.7524
	S4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0002	0.0124	0.0036	0.0000	0.0000	0.0000	0.0000	0.7671
	S9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0073	0.0152	0.0000	0.0000	0.0000	0.0000	0.6744
	S23	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0639	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9995
	S6	0.0000	0.0009	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0785	0.0002	0.0000	0.0241	0.7574
	S10	0.0247	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9980
	S5	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0509	0.0006	0.9864

PURITY = 0.892610837438424
DER = 0.300060319694380
Number of speakers found: 24