DOI: 10.1016/j.engappai.2007.07.002

A Simulated Annealing Approach to Speaker Segmentation in Audio Databases

José M. Leiva-Murillo, Sancho Salcedo-Sanz* [b] ,
Ascensión Gallardo-Antolín and Antonio Artés-Rodríguez [a]

[a] *Department of Signal Theory and Communications, Universidad Carlos III de Madrid.*

[b] *Department of Signal Theory and Communications, Universidad de Alcalá.*

Abstract

In this paper we present a novel approach to the problem of speaker segmentation, which is an unavoidable previous step to audio indexing. Mutual Information is used for evaluating the accuracy of the segmentation, as a function to be maximized by a Simulated Annealing (SA) algorithm. We introduce a novel mutation operator for the SA, the *Consecutive Bits Mutation* operator, which improves the performance of the SA in this problem. We also use the so called *Compaction Factor*, which allows the SA to operate in a reduced search space. Our algorithm has been tested in the segmentation of real audio databases, and it has been compared to several existing algorithms for speaker segmentation, obtaining very good results in the test problems considered.

Key words: Speaker segmentation, simulated annealing, information theory, audio indexing.

---

* Corresponding author: Sancho Salcedo-Sanz. Department of Signal Theory and Communications, Universidad de Alcalá. 28871 Alcalá de Henares. Madrid. Spain.
Tlph: +34 91 885 6731
Fax: +34 91 885 6699
Email address: `sancho.salcedo@uah.es` (Sancho Salcedo-Sanz*).

1 Introduction

Due to the increasing number of TV channels and broadcasting radio stations, many hours of TV and radio broadcasts are collected every year by public institutions and private companies. In addition to the architectural problems in the design of databases for storaging these data, another crucial problem is information retrieval. In audio data files, information retrieval is normally performed by indexing the audio databases, associating each audio document with a file describing its structure in terms of retrieval keys [1].

In order to perform full indexing, an essential initial step is to determine the segmentation of the database with respect to different signals of the audio file (speech, music or noise, for example). In many cases, such as interviews or dialogs, this process consists of knowing which speaker is speaking at a given time. This is known as *speaker segmentation* of an audio database, and it is the problem this paper focuses on.

In addition to indexing, the segmentation of audio databases can be useful for speech recognition purposes [1], speaker verification [2], low bit-rate audio coding and environment and channel change detection. Also, it can provide interesting additional information such as speaker turn and speaker identities (allowing the automatic indexing and retrieval of all occurrences of a same speaker) [3].

The problem of segmenting an audio record has been tackled before by using different approaches. First, in the literature the so called energy-based methods can be found [4], where it is assumed that sentences uttered by different speakers in a conversation are delimited by pauses. As a consequence, the segmentation relies on the accuracy of an inter-speaker silence detector, which usually works by measuring the energy of each segment and comparing it to a predefined or estimated threshold. This technique suffers from two important drawbacks: first, the accuracy of the segmentation strongly depends on the choice of the energy threshold; secondly, it is not always true that people speak between significant silences.

Other significant algorithms used in the segmentation of audio data records are the so called distance-based methods. This approach consists of measuring the dissimilarity between two adjacent windows of (parameterized) audio data. Then, the system locates a changing mark in the point in which the dissimilarity is maximum. Several dissimilarity measures have been proposed in the literature, like the Generalized Likelihood Ratio (GLR) [5], the Kullback-Leibler distance or the Bayesian Information Criterion (BIC) [6], [7]. Again, the main drawback is the presence of a threshold which has to be tuned for each kind of audio database.

Finally, we mention the model-based segmentation, in which a statistical model (i.e. a Gaussian Mixture model [8], [9]) is trained for a set of predefined acoustic classes (speech, speaker, background noise, music, telephone speech, etc.). For segmentation purposes, each frame (or various frames) of the audio stream is classified by using a Maximum Likelihood criterion and the segment boundaries are located in the temporal point where a change of the acoustic class occurs. The main disadvantages of this method are the need of predefining the number and nature of the acoustic classes, and the necessity of having available labelled data for building the different acoustic models.

In this paper we consider the problem of the segmentation of audio records containing two speakers. The problem consists of automatically marking the periods of time in which every speaker is talking (speaker turns). We propose a novel approach to this task, using a Simulated Annealing (SA) algorithm, [10], [11], which maximizes a measure of Mutual Information (MI) [12] between classes and data. Mutual Information, $I(\mathbf{x}, C)$, is a concept taken from Information Theory (IT) [12], which measures the quantity of "common" information between a sequence of labels C and a vector of data $\mathbf{x}$. Intuitively, signals with a high degree of MI between samples and classes are more easily separable than others that contain a lower level. In this paper we use a new approach to MI [13], which is based on direct approximation of entropy. The samples, $\mathbf{x}$, involved in the calculation of $I(\mathbf{x}, C)$ are the Mel Frequency Cepstrum Coefficients (MFCC) of the audio record, whereas the classes, C, are binary values $\{0, 1\}$ which represent who the speaker currently talking is. A SA algorithm is then used to obtain the sequence of classes C^o which maximizes the MI $I(\mathbf{x}, C)$. Since the problem consists of segmenting audio records containing two speakers, a SA with binary representation is suitable for this purpose. Thus, the SA codifies the sequence C as a binary string (configuration in the SA notation) which represents a possible segmentation of the audio record. After the SA process, the current configuration represents the sequence C^o (segmentation of the audio record). We test two mutation operators for the SA algorithm, the traditional Random Flip Mutation, and a novel mutation operator introduced in this paper: the Consecutive Bits Mutation, which improves the performance of the algorithm in this problem. We describe in detail these two operators in Section 3.

We have tested our approach in the segmentation of real and synthetic audio records of different lengths. We have compared our results to several existing approaches to speaker segmentation: the DISTBIC algorithm in [3], a clustering algorithm and a hidden Markov model approach, obtaining solid improvements in all test performed.

The rest of the paper is structured as follows: In the next section, we give a brief description of Mel Frequency Cepstrum Coefficients, used in this article for parameterizing the audio signal. Section 3 introduces the SA algorithm we propose, and Section 4 shows the results obtained by our algorithm segmenting real and synthetic audio records, and they are discussed through comparison with the results obtained by other existing approaches to the segmentation of audio records. Section 5 concludes the paper giving some final remarks.

2 Background: MFCC parameterization of speech signals

Speech (in general, audio) signals need to be parameterized prior to the segmentation process. Parameterization consists of the extraction of a set of features from the speech waveform, which have to have two main characteristics: they must provide a reasonable and compact representation of the speech signal (usually, in the time-frequency domain) and they must have adequate discrimination capabilities for distinguishing between sounds.

MFCCs [14] are the most commonly used Fourier-based parameters in Automatic Speech Recognition (ASR), speaker verification and more recently, in speaker change detection [6]. In this paper we have also used the MFCCs for representing the speech signal. In this subsection, we briefly describe the procedure for extracting them, as illustrated in Figure 1.

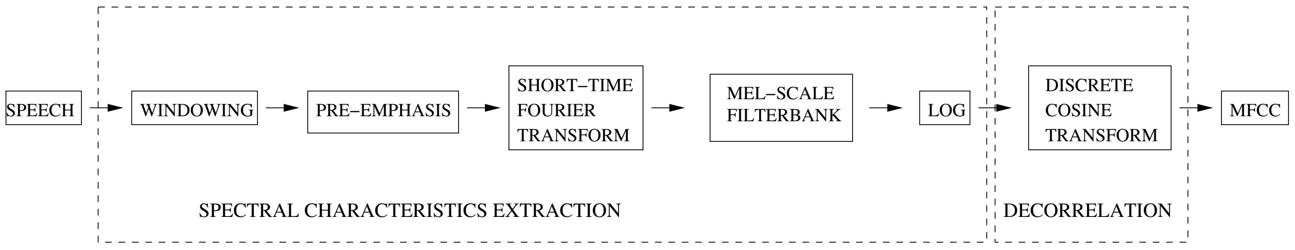
SPEECH
WINDOWING
PRE–EMPHASIS
SHORT–TIME FOURIER TRANSFORM
MEL–SCALE FILTERBANK
LOG
DISCRETE COSINE TRANSFORM
MFCC
SPECTRAL CHARACTERISTICS EXTRACTION
DECORRELATION

Fig. 1. Top-level block diagram of the speech parameterization procedure. The diagram illustrates the steps followed for the extraction of the MFCC coefficients.

Since the speech signal varies for each sound and is not stationary, speech analysis must be performed on short windowed segments. Typically, a speech signal is divided into a number of overlapping temporal windows (called frames) and a speech parametric vector is computed to represent each time frame. The frame period (interval between analysis instants) is usually set to a value between 10 and 20 milliseconds (ms. hereafter), in our case, the frame period was 10 ms. Next, we describe the parameterization procedure for each frame at a given time t:

- As mentioned before, due to its non-stationary nature, the speech signal has to be divided into quasi-stationary segments in which an accurate spectral analysis can be performed. For this purpose, the speech signal is windowed using a Hamming window defined by the following expression:

$$w[n] = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \qquad 0 \leq n \leq N-1 \tag{1}$$

in which the window length is set to 32 ms. ($N = 512$ for a sampling frequency of 16 kHz).

- A pre-emphasis filter is applied to the windowed speech signal in order to compensate from the negative spectral slope of its voiced portions. The impulse response of this filter is:

$$h[n] = \delta(n) - \alpha\, \delta(n-1) \tag{2}$$

in which the value of the pre-emphasis coefficient, α, usually falls between 0.85 and 1.0 ($\alpha = 0.95$, in our case).

- Next, the N-point spectrum of the speech frame is computed, via the fast Fourier transform, as follows:

$$S[k] = X\left(\Omega = k\frac{2\pi}{N}\right) = \sum_{n=0}^{N-1} s[n]e^{-jk\frac{2\pi}{N}n}, \qquad N = 512,\ k = 0, \ldots, 255 \tag{3}$$

where $S(k)$ is the k^{th} component of the short-time Fourier transform of $s[n]$ which is the previously windowed and pre-emphasized speech signal. Due to the symmetry properties of the spectrum of real signals, we only consider the 256-sample positive half of it.

- The power spectrum $|S[k]|^2$ is filtered using a set of M (we have employed $M = 40$) triangular band-pass filters, where the energy corresponding to the output of each filter, $E[i]$, is computed as follows

$$E[i] = \sum_{k=0}^{N-1} |S[k]|^2\ W_i[k], \qquad i = 0, \ldots, M-1 \tag{4}$$

where $W_i[k]$ is the k^{th} frequency component of the i^{th} element of the filter-bank. The central frequencies of the filter-bank are distributed following a mel scale which is linear for frequencies below 1 kHz and approximately logarithm for frequencies above 1 kHz, in order to obtain a better frequency resolution for low frequencies than for high ones. In addition, the filters are half-overlapped in the mel domain. This filtering process simulates the behavior of the human auditory system, which is known to be more discriminative for low frequencies.

- Finally, the log-energies, $\log(E[i])$, are subsequently de-correlated using a Discrete Cosine Transform yielding to N_c (12, in our case) Mel Frequency Cepstrum Coefficients, $\mathbf{x}$,

$$x[m] = \sqrt{\frac{2}{M}} \sum_{i=0}^{M-1} \log(E[i]) \cos\left(\frac{\pi m}{M}(i + \frac{1}{2})\right), \qquad m = 1, \ldots, N_c \tag{5}$$

$x[m]$ are the input data for the calculation of the simulated annealing's objective function, as we will show in the next section.

3 Speaker segmentation based on simulated annealing

SA has been widely applied to solve combinatorial optimization problems [15], [16], [17]. It is inspired by the physical process of heating a substance and then cooling it slowly, until a strong crystalline structure is obtained. This process is simulated by lowering an initial temperature by slow stages until the system reaches to an equilibrium point, and no more changes occur. Each stage of the process consists of changing the configuration several times, until a thermal equilibrium is reached, and a new stage starts, with a lower temperature. The solution of the problem is the configuration obtained in the last stage. In the standard SA, the changes in the configuration are performed in the following way: A new configuration is built by a random displacement of the current one. If the new configuration is better, then it replaces the current one, and if not, it may replace the current one probabilistically. This probability of replacement is high in the beginning of the algorithm, and decreases in every stage. This procedure allows the system to move toward the best configuration. Although SA is not guaranteed to find the global optima, it is better than other algorithms at escaping from local optima. Its solution can be considered to be "good enough", but it is not guaranteed to be the best.

The most important part in a SA algorithm are: the chosen representation for solutions, the objective function to be minimized during the process and the mutation or configuration change operator. We present these three characteristics in the next subsections.

3.1 Problem representation

Every possible segmentation (configuration in the SA) has been coded by means of a binary string, in which every bit codifies 10 ms. of audio (due to the MFCC frame period, see Section 2). Thus, the labelling of each minute of audio to be segmented is represented by a binary string of length $l = 6000$ bits. This implies that the search space will have a size of 2^{6000}; in such a search space the SA will have problems of convergence, obtaining low quality solutions.

It is possible to overcome this difficulty by analyzing the problem's structure: first of all, note that we have codified a solution with an accuracy of 10 ms. (one second of audio is represented by 100 bits). That is, we would be able to detect changes in speakers with such accuracy by using the representation exposed above. On the other hand, in a standard audio record, a speaker rarely talks less than one second, which means that the optimal solution will have large strings of all 1s (or 0s) for codifying the segmentation of the audio record. For example if a speaker talks for three seconds before changing to other speaker, the optimal solution would be a string of 300 1s (0s) before changing to 0s (1s). Thus, it is possible to reduce the length of the representation by compacting a number CF of bits into one. We call this parameter the Compaction Factor CF. Figure 2 shows this process. Note that the new length of SA configurations will be $l' = \frac{l}{CF}$, after applying the Compaction Factor.

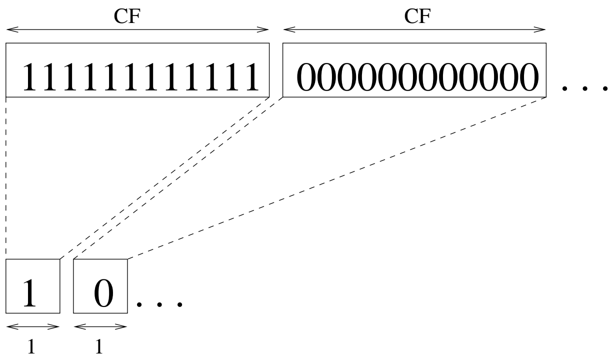

Fig. 2. Example of encoding compression.

In our approach, the SA is run using this new representation l'. We say then that the SA is being run in its compacted form. Note, however, that the calculation of the objective function involves individuals of length l and not l' (because of the audio data length), so every individual in the compacted SA has to be expanded, i.e. every bit expanded to CF equal bits, for the fitness calculation.

This length reduction of individuals in the SA affects, obviously, to the accuracy of the encoding: using the expanded representation we have an accuracy of 10 msecs. for detecting changes of speaker. If we use the compacted form of the SA, the accuracy of segmentation is reduced in a factor of $10 \cdot CF$ milliseconds. Thus, for instance, if an accuracy of one second is enough for detecting speaker changes, we could set $CF = 100$, and the length of individuals in the GA will be reduced as $l' = \frac{l}{100}$. If we want to be more accurate, the compaction factor CF has to be smaller.

3.2 Objective function: Mutual Information

The Mutual Information (MI) between two signals is described by Shannon's Information Theory as the quantity of information the signals carry about each other. Thus, it is a generalization of the concept of entropy. In this case, the uncertainty of each variable is measured with respect to the other one [12]:

$$I(\mathbf{u},\mathbf{v}) = h(\mathbf{u}) - h(\mathbf{u}|\mathbf{v}) \tag{6}$$

where $h(\cdot)$ is the differential entropy of a given multidimensional variable, and is defined as:

$$h(\mathbf{u}) = -\int p(\mathbf{u}) \log p(\mathbf{u}) d\mathbf{u}$$

and $p(\mathbf{u})$ is the probability density function (*pdf*) of $\mathbf{u}$.

In a learning problem, the variables involved are the multidimensional data $\mathbf{x} \in \mathbb{R}^d$ and a discrete and finite set of classes $C \in \{c_1, c_2, \ldots, c_K\}$, that are the patterns to be learned. Thus, Equation 6 may be re-expressed as:

$$I(\mathbf{x},C) = h(\mathbf{x}) - h(\mathbf{x}|C) = h(\mathbf{x}) - \sum_k p(c_k) h(\mathbf{x}|c_k) \tag{7}$$

Unfortunately, the problem of estimating the entropy is difficult except for few, analytically defined *pdf*s. However, for the simpler case of one-dimensional signals, some successful approximations have been made. In order to make use of such estimations, the cost function based on MI will be modified as:

$$I(\mathbf{x},C) \approx \sum_i I(x_i, C) \tag{8}$$

that is equivalent to assume $h(\mathbf{x}) \approx \sum_i h(x_i)$ and $h(\mathbf{x}|c_k) \approx \sum_i h(x_i|c_k)$ where i indexes each dimension in vector $\mathbf{x}$. This is not a significant change, since in some applications the statistical dependence between the components x_i is very low. In the case of speech parameterization, MFCC are considered to have a low correlation and statistical dependence among them.

The problem of estimating the *pdf* of each one-dimensional component can be avoided by directly computing the entropies from statistics of the data. Two cumulant-based polynomial expansions have been traditionally used for this task: the Gram-Charlier series and the Edgeworth series [18]. These approximations assume an expression for the entropy as:

$$h(x) = h(x_{gauss}) - J(x)$$

where x_{gauss} follows a gaussian distribution with the same variance as x and $J(x)$ is the so-called *negentropy* of the random variable x, and must be computed by means of the statistical moments with different orders over the set of realizations. This quantity is always positive since a gaussian random variable is, among all the possible distributions with the same variance, the one with the highest entropy. Both Edgeworth and Gram-Charlier expansions have a drawback: they are very sensitive to outliers and, in general, to any outstanding samples, coming from the tails of the distributions. Alternative estimations of the negentropy have been successfully used in previous works in Independent Component Analysis (ICA). The approximation of $J(x)$ given by the expression:

$$h(x) = h(x_{gauss}) - k_1 \left(E\{xg(x)\}\right)^2 + k_2 \left(E\{g(x)\} - 1/\sqrt{2}\right)^2 \tag{9}$$

where k_1 and k_2 are constants defined as $k_1 = \dfrac{36}{8\sqrt{3}-9}$ and $k_2 = \dfrac{24}{16\sqrt{3}-27}$, respectively [19]. This expression has been proven to be more robust and stable against outliers, providing values for the negentropy quite close to the actual ones. In this case, we have use as non-polinomial function $g(x) = \exp\left(\dfrac{-x^2}{2}\right)$, as proposed in [19].

The problem's objective function is then given by Equation 8 based on the entropy estimation in Equation 9.

3.3 Mutation operator

In this paper we consider two mutation operators for the SA. First, a clasical Random Flip Mutation (RFM) operator is applied. The RFM operator obtains a configuration in the neighborhood of the current one by means of randomly selecting and flipping N_f bits.

The second mutation operator used consists of changing S_f sets of K_f consecutive bits with randomly selected values 1 or 0. We call this mutation operator Consecutive Bits Mutation (CBM) operator. As was mentioned before, in a standard audio record, a speaker usually talks more than one second, which means that the optimal solution will have large strings of all 1s (or 0s) for codifying the segmentation of the audio record. Thus, it seems that the mutation operator should have the ability of produce chains of bits set to 1 or 0. The CBM operator mutates sets of consecutive bits, trying to achive these chains of bits. Figure 3 shows an example of how CBM performs in an instance with $S_f = 2$ and $K_f = 5$. Note that the CBM will produce binary strings with more equal consecutive bits than the standard RFM.

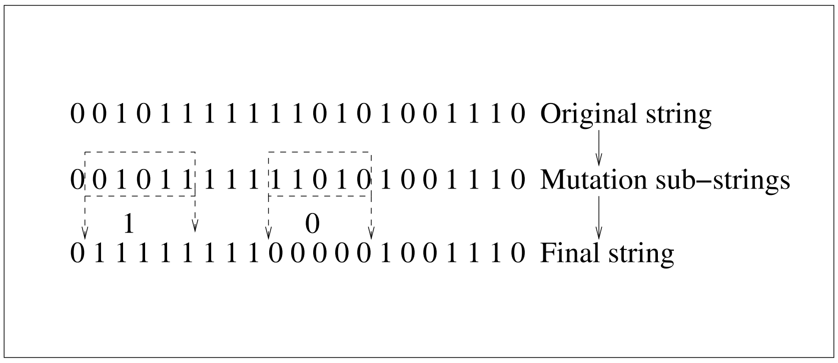
0 0 1 0 1 1 1 1 1 1 1 0 1 0 1 0 0 1 1 1 0 Original string
0 0 1 0 1 1 1 1 1 1 1 0 1 0 1 0 0 1 1 1 0 Mutation sub−strings
1
0
0 1 1 1 1 1 1 1 1 0 0 0 0 0 1 0 0 1 1 1 0 Final string

Fig. 3. Example of how the CBM mutation operator works ($S_f = 2$ and $K_f = 5$).

3.4 The complete algorithm

The complete SA algorithm for the segmentation of audio data records containing two speakers performs in the following way:

```
$k = 0$;
$T = T_0$;
Initialize the current configuration $C$ at random;
$(C \rightarrow C_{expanded})$: **evaluate**$(I(\mathbf{x}, C_{expanded}))$;
**repeat**
   **for** $j = 0$ **to** $\xi$
      $C^{mut} = \textbf{mutate}(C)$;
      $(C^{mut} \rightarrow C^{mut}_{expanded})$: **evaluate**$(I(\mathbf{x}, C^{mut}_{expanded}))$;
         **if**$((I(\mathbf{x}, C^{mut}_{expanded}) > I(\mathbf{x}, C_{expanded}))$ OR $(random(0,1) < e^{(\frac{-a}{T})}))$ then
         $C = C^{mut}$;
         **endif**
   **endfor**
$T = f_T(T_0, k)$;
$k = k + 1$;
**until**$(T < T_{min})$;
```

where k counts the number of iterations performed; T keeps the current temperature; T_0 is the initial temperature; T_{min} is the minimum temperature to be reached; C stands for the current configuration (compacted form), C^{mut}

stands for the new configuration after mutation operator is applied (compacted form), $C_{expanded}$ for the current configuration in its expanded form and $C^{mut}_{expanded}$ stands for the new configuration after mutation in its expanded form. $I(\cdot,\cdot)$ represents the mutual information we use as objective function (see Section 3.2); $\mathbf{x}$ represents the vector of data containing the MFCC of the audio record to be segmented; ξ is the number of changes performed with a given temperature T; f_T is the freezer function; and a is a previously fixed constant. The parameter a and the initial temperature T_0 are calculated in order to the initial acceptance probability to be 0.8, which is the value commonly used. We do not allow that the probability of acceptance to be lower than 0.005 until the last 50 iterations of the algorithm.

The freezer function is defined as

$$f_T = \frac{T_0}{1+k}. \tag{10}$$

The minimum temperature T_{min} is calculated on the basis of the desired number of iterations as:

$$T_{min} = f_T(T_0, numIt). \tag{11}$$

4 Computational experiments and results

In this section, first we briefly describe the speech databases used; secondly, we describe the existing algorithms we use for comparison. Finally, we report and discuss the results obtained by our algorithm comparing its performance with the previous approaches mentioned.

4.1 Test problems

Two different types of speech data have been used to test the performance of our algorithm, artificially created audio records and real audio records from TV interviews:

- Several conversations with a total duration of approximately 12.5 minutes were artificially created by concatenating sentences from the Resource Management RM1 Database [20]. This database consists of speech recorded at 16 kHz in clean conditions, and it has been widely used by the speech technology community for Automatic Continuous Speech Recognition (CSR) assessment. The original pauses between sentences were shortened to an average duration of approximately 185 ms. for a better simulation of real conversations.

The conversations created contain a total of 215 speaker turns and the duration of each segment varies from 1.17 seconds to 6.5 seconds with an average length of 3.35 seconds. A CF of 30 (which corresponds to a segmentation resolution of 300 ms.) was used in all the instances.

- Several TV news broadcasts (corresponding to interviews) with a total duration of 23.35 minutes were extracted from the 1997 HUB English Evaluation Speech Database, distributed by NIST [21]. The original aim of this database was to foster research on the problem of accurately transcribing broadcast news speech, in which the first step is the segmentation of the speech data into homogeneous segments (same speaker, same acoustic environment). The selected data contains spontaneous speech recorded at 16 kHz and at different acoustic conditions (clean and in telephone environment). NIST provides hand-segmentations of these data that we have used as a reference.

  The conversations extracted from this database contain 88 segment boundaries, which correspond to an average segment length of approximately 14.15 seconds, with a maximum length of 52.08 seconds and a minimum of 0.75 seconds. The average duration of the pauses between speech segments is about 200 ms. This length distribution is typical for interviews, in which the shortest segments usually corresponds to the questions from journalists. A CF of 30 was also used in this case.

4.2 Assessment methods

We consider two type of errors measures related to speaker turn detection. False alarms (FA) occur when a speaker turn is detected although it does not exist. The empirical false alarm rate (FAR) is defined as:

$$\mathrm{FAR} = 100 \cdot \frac{\text{number of FA}}{\text{number of actual speaker turns} + \text{number of FA}}\ \% \tag{12}$$

Missed detections (MD) occur when the algorithm does not detect an existing speaker turn. The empirical missed detection rate (MDR) is calculated as follows:

$$\mathrm{MDR} = 100 \cdot \frac{\text{number of MD}}{\text{number of actual speaker turns}}\ \% \tag{13}$$

For measuring FAR and MDR, it is necessary to take into account that the position of the speaker turns are not exactly defined, due to the presence of inter-speaker silences or non-speech sounds [22]. Therefore, it is considered that a changing point is correctly located if it belongs to a time interval $[t_o - \Delta t, t_o + \Delta t]$ in which t_o is the reference mark and Δt is the tolerance (600 ms., in our case).

4.3 Algorithms for comparison

4.3.1 DISTBIC

DISTBIC algorithm [3] is a distance-based method based on the Bayesian Information Criterion (BIC), first proposed in [27]. BIC uses a likelihood ratio, in which it is decided whether two fragments belong to the same source or to two different ones. The log-likelihood ratio associated with the frame i is defined as:

$$R(i) = \log \frac{L(H_0)}{L(H_1)L(H_2)} \tag{14}$$

being H_0 the hypothesis of that there is not a change of source in i. $L(H_0)$ is its corresponding likelihood when a Gaussian distribution is assumed. H_1 assumes all frames with index $\leq i$ to belong to speaker 1 and so H_2 does with index $> i$ and speaker 2.

BIC criterion takes also into account the complexity of the solution. The cost function is given by:

$$\Delta BIC(i, m) = -R(i) + \lambda P(m) \tag{15}$$

where $P(m)$ is the penalizing term when m parameters are used, being λ a threshold parameter. Samples with the higher ΔBIC are the most likely to correspond to a change.

DISTBIC is based on an sliding-windowing that applies BIC to frames all along the sequence. After measuring the $\Delta BIC(i, m)$, DISTBIC carries out two later steps of refinement and validation that improve the performance obtained if just the BIC criterion were applied.

There are two main parameters (apart from the threshold parameter λ) DISTBIC depends on. The first one is the size of the sliding window from which the Gaussian models are built, i.e. the number of samples with index $\leq i$ the hypothesis H_1 is built according with. This size is usually maintained fixed, with a typical value of 1 second [3]. The second parameter is the shift of the window, which determines the resolution of the method.

4.3.2 Clustering-based segmentation

Speaker segmentation of audio data files can be carried out by using an hierarchical agglomerative clustering algorithm as proposed in [25]. This technique consists of dividing the audio data into a certain number of segments (clusters) and iteratively merging pairs of clusters according to a predetermined metric. For two-speaker segmentation, this procedure finishes when two clusters (each one containing the part of speech uttered by each speaker) are obtained.

In the initialization stage of the clustering algorithm, the data are divided into segments of equal length. The initial size of the clusters determines the resolution of the segmentation procedure and, in this sense, it plays a similar role that the CF factor in our SA approach. Thus, to allow a more fair comparison to our SA method, the clusters have an initial size of CF= 30 (corresponding to 300 ms.) in both databases.

The distance between two clusters is based on the log-likelihood ratio defined in Equation 14. To computate the corresponding likelihoods, each cluster is modelled by tied mixtures of multivariate Gaussian distributions in the cepstral space, which are trained following the procedure described in [25]. We have carried out different experiments varying the number of mixtures and we find that, for our databases, the best results are obtained using 32 mixtures.

4.3.3 HMM-based segmentation

Hidden Markov models can also be used for speaker segmentation as it has been shown in [25] and more recently in [22] and [24].

In this case, speakers are considered as different acoustic classes. Each of these classes is statistically represented by a mixture of multivariate gaussian densities which are trained before the segmentation. Then, the audio data is classified using a Maximum Likelihood criterion with a Viterbi decoder [26] that yields a set of boundaries between classes corresponding to the hypothesized speaker turns.

To build the corresponding models, some labelled data is needed. In real applications, it is difficult to obtain these audio data, so we have adopted an unsupervised strategy in the training stage as proposed in [25]. In particular, we have used the segmentation provided by the agglomerative clustering method described in the previous subsection for the initialization of the speaker models which are adequately retrained using the well known Baum-Welch algorithm [26].

To design the HMM-based segmenter we have used the HMM topology shown in Figure 4. A similar approach has been proposed in [9] for speech and music segmentation and recently adapted for speaker segmentation purposes in [24].

As it can be seen in Figure 4, the system consists of two fully connected HMM sub-networks, each one corresponding to each speaker. Each sub-network is composed of several internal states associated with the same mixture gaussian distribution. It is important to note that the number of concatenated states imposes a minimum segment duration and determines the resolution of the algorithm. For a better comparison with clustering and SA approaches, we have enforced the same constraint of minimum duration of 300 ms.

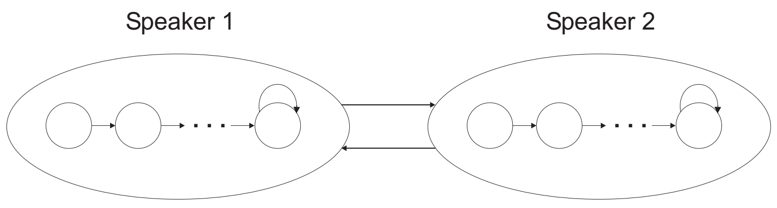

Fig. 4. HMM topology for the HMM-based speaker segmentation system.

As information about prior probabilities of speakers is not available, we have assumed that both speakers are equally likely. Transition probabilities between speakers have been empirically selected in order to favor remaining in the current state (speaker 1 or speaker 2).

Preliminary experiments showed that using 32 mixtures components per internal state provides a good segmentation accuracy, so we have used this value in the experiments described in next subsection.

4.4 Results

The SA algorithm defined in Section 3 is used in the simulations, with the following parameters: the number of iterations of the SA, *numIt*, was fixed to 300 with $\xi = 50$. The parameters of the CMB operator were empirically fixed to be $S_f = 3$ and $K_f = 5$, whereas the value of N_f in the RFM operator was fixed to 20 (see Section 3.3). A Compaction Factor $CF = 30$ has been used in both databases considered. We compare the results obtained using the SA with those obtained by the DISTBIC algorithm, the clustering and HMM approaches.

Figure 5 shows the average values of FAR versus MDR obtained by the SA with RFM and CBM operators, clustering and HMM over all the conversations in the RM1 database (one point each in the figure). We compare these results with the Detection Trade-off (DET) curves obtained with the DISTBIC algorithm. The DET curve is calculated by means of varying the DISTBIC threshold. It shows the number of MD in function of the number of FA. Note that the results obtained by the DISTBIC algorithm strongly depend on the election of the threshold. In this figure we also indicate the Equal Error Rate (EER) point, which is defined as the point in which the value of FAR is equal to the value of MDR for the DISTBIC DET curve. Note that the point obtained with the SA using the CBM operator is below the EER point of DISTBIC, and also below the points obtained by the clustering and HMM approaches. The SA with the RFM does not provide good results in these problems.

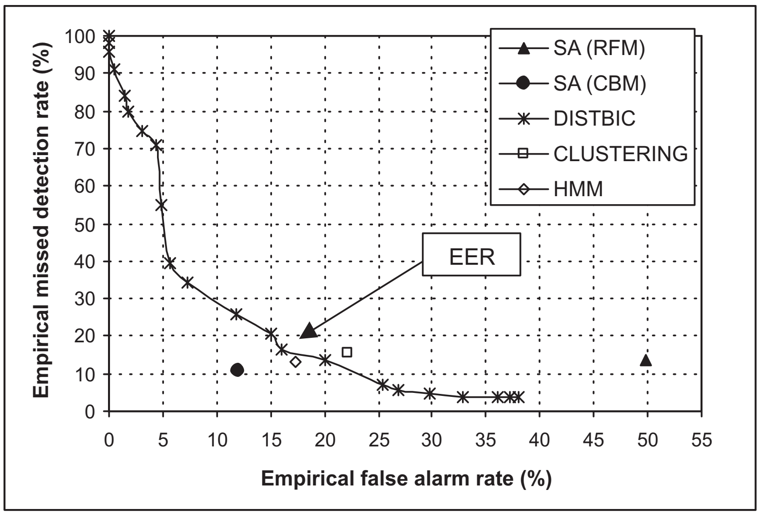

Fig. 5. DET curve obtained varying the DISTBIC threshold parameter, and FA-MD rates obtained using the SA approach with RFM and CBM, clustering and HMM algorithms for the RM1 database.

Table 1 shows the false alarm rate (FAR) and missed detection rate (MDR) for the RM1 database, for the different algorithms considered. The results of DISTBIC correspond to the EER point, note that the average values in the table corresponds to the FAR and MDR over all the conversations in the RM1 database. It is easy to note that the RFM mutation operator in the SA produces a larger number of false alarms compared with the SA with CBM. However, the SA algorithm working with the CBM obtains better results in

Table 1
Change detection error rates (FAR and MDR) obtained with our approach with the CBM mutation operator, compared with the RFM mutation operator, DISTBIC, clustering and HMM methods for the RM1 Evaluation Database.

Segmentation method	FAR	MDR
SA (CBM)	11.89 %	10.70 %
SA (RFM)	49.88 %	13.49 %
DISTBIC (EER point)	16.02 %	16.28 %
CLUSTERING	22.10 %	15.35 %
HMM	17.31 %	12.95 %

terms of FAR and MDR than all the other algorithms.

Results for HUB 97 database are shown in Figure 6 and Table 2. The same experiments as for the RM1 database have been carried out. It is easy to see that the SA with the CBM operator outperforms the compared approaches. The differences between our SA with CBM and the points obtained by the other algorithms (EER point for the DISTBIC), are even larger in this database.

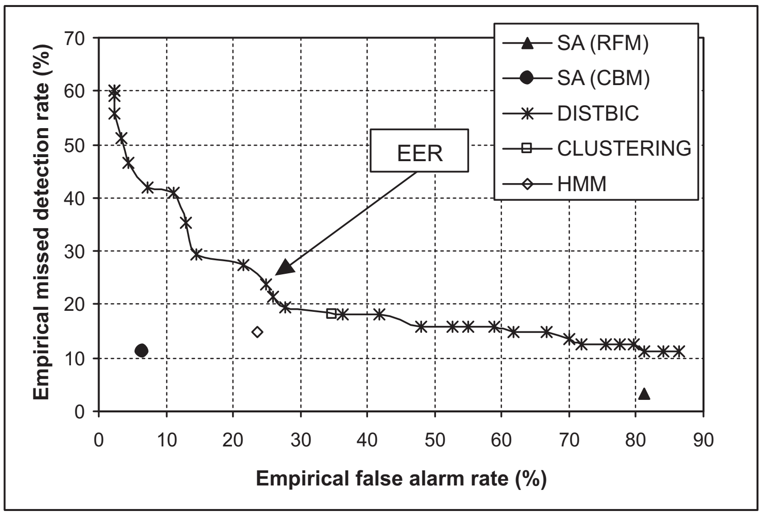

Fig. 6. DET curve obtained varying the DISTBIC threshold parameter, and FA-MD rates obtained using the SA approach with RFM and CBM, clustering and HMM algorithms for the HUB 97 database.

5 Conclusions

In this paper we have presented a Simulated Annealing (SA) approach to the segmentation of audio databases. The SA algorithm looks for the best

Table 2
Change detection error rates (FAR and MDR) obtained with our approach with the CBM mutation operator, compared with the RFM mutation operator, DISTBIC, clustering and HMM methods for the NIST HUB 97 Evaluation Database.

Segmentation method	FAR	MDR
SA (CBM)	6.38 %	11.36 %
SA (RFM)	81.12 %	3.41 %
DISTBIC (EER point)	24.79 %	23.86 %
CLUSTERING	34.81 %	18.18 %
HMM	23.48 %	14.77 %

segmentation, i.e. the binary string which maximizes a measure of Mutual Information with the samples of the audio record. We introduce several modifications in the algorithm like the so called Consecutive Bit Mutation (CBM) operator, which improves SA performance in this problem, and the so called Compaction Factor (CF), which allows the SA to operate in a reduced search space. The performance of the presented algorithm has been tested in several conversations from two different audio databases, and has been compared with several state of the art approaches to segmentation of audio data files, obtaining very good results in all the experiments carried out.

References

[1] T. Zhang and C.-C. Jay Kuo, "Audio content analysis for online audiovisual data segmentation and classification," *IEEE Trans. on Speech and Audio Processing* **9** 4 (2001) 441–457.

[2] K. Chen, "Towards better making a decision in speaker verification," *Pattern Recognition* **36** (2003) 329–346.

[3] P. Delacourt and C. J. Wellekens, "DISTBIC: a speaker-based segmentation for audio data indexing," *Speech Commun.* **32** (2000) 111–126.

[4] M. Nishida and Y. Ariki, "Speaker indexing for news articles debates and drama in broadcasted TV programs", In *Proc. of the Speech Recognition Workshop*, 1997, pp. 67-72.

[5] A. G. Adami, S. Kajarekar and H. Hermansky, "A new speaker change detection method for two-speaker segmentation", In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'02)*, 4, Orlando, USA, 2002, pp. 3908-3911.

[6] S. S. Chen and P. S. Gopalakrishnan, "Speaker, environment and channel change detection and clustering via the bayesian information criterion", In *Proc. of the DARPA Broadcast News Trancription and Understanding Workshop*, Landsdowne, VA, 1998.

[7] J. Ferreiros-López and D. P. W. Ellis, "Using acoustic condition clustering to improve acoustic change detection on broadcast news", In *Proc. Int. Conf. on Spoken Language Processing (ICSLP'98)*, Beijing, China, 2000, pp. 568-571.

[8] R. Bakis, S. Chen, P. Gopalakrishnan, R. Gopinath, S. Maes and L. Polymenakos, "Transcription of broadcast news-system robustness issues and adaptation techniques", In *Proc. IEEE Int. Conf. on Accoustics Speech and Signal Processing (ICASSP'97)*, 2, Munich, Germany, 1997, pp 711-714.

[9] J. Ajmera, I. McCowan and H. Bourlard, "Speech/music segmentation using entropy and dynamism features in a HMM classification framework," *Speech Commun.* **40** (2003) 351-363.

[10] S. Kirpatrick, C. D. Gerlatt and M. P. Vecchi, "Optimization by simulated annealing," *Science* 220 (1983) 671-680.

[11] S. Kirpatrick, "Optimization by simulated annealing–quantitative studies," *J. Stat. Phys.* **34** (1984) 975–986.

[12] T.M. Cover and J.A. Thomas, *Elements of information theory*, John Wiley & Sons, 1991.

[13] J. M. Leiva-Murillo, R. Santiago-Mozos, S. Salcedo-Sanz and A. Artés-Rodríguez, "Symbol decision via genetic optimization of mutual information," In *Proc. 6th Baiona Workshop on Signal Processing in Communication*, Baiona, Spain, 2003, pp. 51-56.

[14] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences", *IEEE Trans. on Acoustics, Speech and Signal Processing* **28** 4 (1980) 357–366.

[15] J. González, I. Rojas, H. Pomares, M. Salmerón and J. J. Merelo, "Web newspaper layout optimization using simulated annealing," *IEEE Trans. Systems, Man and Cybernetics*, **32** 5 (2002) 686–691.

[16] G. Wang and N. Ansari, "Optimal broadcast scheduling in packet radio networks using mean field annealing," *IEEE J. Select. Areas Commun.* **15** 2 (1997) 250–259.

[17] S. Salcedo-Sanz, R. Santiago-Mozos and C. Bousoño-Calzón, "A hybrid Hopfield network-simulated annealing approach for frequency assignment in satellite communications systems," *IEEE Trans. Systems, Man and Cybernetics* **34** 2 (2004) 1108–1116.

[18] S. Haykin, *Neural Networks*, Prentice Hall, 1999.

[19] A. Hyvarinen, "New approximations of differential entropy for independent component Analysis," *Advances in Neural Information Processing System* **10** (1998) pp. 273-279.

[20] NIST, The Resource Management corpus (RM1), 1998.

[21] NIST, 1997 HUB4 English Evaluation Speech and Transcripts, *produced by the Linguistic Data Consortium, Catalog No. LDC2002S11*, http://morph.ldc.upenn.edu/Catalog/LDC2002S11, 1997.

[22] T. Kemp, M. Schmidt, M. Westphal and A. Waibel, "Strategies for automatic segmentation of audio data," In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'00)*, 3, Istanbul, Turkey, 2000, pp. 1423-1426.

[23] P. Chiu, A. Girgensohn, W. Polak, "A genetic segmentation algorithm for image data streams and video," In *Proc. of GECCO*, 2000, pp. 666-673.

[24] G. Lathoud and I. A. McCowan, "Location based speaker segmentation," In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1, 2003, pp. 176-179.

[25] L. Wilcox and F. Chen and D. Kimber and V. Balasubramanian, "Segmentation of speech using speaker identification," In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1, 1994, pp. 161-164.

[26] L. R. Rabiner, "A tutorial in hidden Markov models and selected applications in speech recognition", *Proceedings of the IEEE* **77** 2 (1989) 257–286.

[27] G. Schwarz, "Estimating the dimension of a model," *Annals of Statistics* **6** (1978) 461–464.