Speaker Change Detection and Tracking in Real-Time News Broadcasting Analysis

Lie Lu, Hong-Jiang Zhang
Microsoft Research, Asia
Beijing, China, 100080
{llu, hjzhang}@microsoft.com

ABSTRACT

This paper addresses the problem of real time speaker change detection and speaker tracking in broadcasted news video analysis. In such a case, both speaker identities and number of speakers are assumed unknown. A two-step speaker change detection algorithm, including potential change detection and refinement, is proposed. Speaker tracking is performed based on the results of speaker change detection. A Bayesian Fusion method is used to fuse multiple audio features to get a more reliable result. The algorithm has low complexity and runs in real-time with a very limited delay in analysis. Our experiments show that the algorithms produce very satisfactory results.

Keywords

Audio content analysis, speaker tracking, speaker segmentation, speaker change detection

1. INTRODUCTION

Speaker recognition, which includes speaker identification and verification [1][7], are widely researched in recent years. In these existing speaker recognition systems, it is supposed that the input speech belongs to one of the known speakers. However, in many applications, such as in a real-time conversation or news broadcasting, the speech stream is continuous and there is no information about the beginning and ending of the speech segment of one speaker. Therefore, if we need to index speech streams based on speaker or perform video content analysis based on audio track, it is necessary to find speaker change points first in such applications before the speaker can be identified. This procedure is called speaker segmentation, or speaker change detection. Speaker tracking, or segmenting a speech stream by speaker identities is essential in many applications, such as conference and meeting indexing [6][18], audio/video retrieval or browsing [24][25], speaker adaptation for speech recognition [12][21] and video content analysis, and speaker change detection is a preliminary processing for speaker tracking.

Unlike the speaker identification or verification problem defined in most previous studies, we assume that there is no prior knowledge about the number and the identities of speakers in speaker tracking process. If the speakers are registered *a priori*, traditional speaker identification algorithm can be used for speaker segmentation and tracking as it was done in the work on supervised speaker identification in HF radio by Brummer [2]. However, in many cases, such as continuous speech stream from live news broadcasting or a meeting, the *a priori* knowledge of speaker identities and the number of speakers are often not available or difficult to obtain. Even in the well structured news broadcasting, we cannot assume that the anchorpersons are always the same. Therefore, it is desirable to perform unsupervised speaker change detection and tracking algorithm in audio content analysis.

There are several papers that reported work on unsupervised speaker identification or tracking, using different algorithms in different applications. In the work of Sugiyama [3], a simpler case was studied. The number of the speakers to be clustered was assumed to be unknown, and VQ and HMM (Hidden Markov Model) were used in the implementation. The algorithm proposed by Wilcox [4] was also based on HMM segmentation, in which an agglomerative clustering method was used when the prior knowledge of speakers is unknown. Another system [5][8] was proposed to separate controller speech and pilot speech with GMM model; and speech and noise detection were also considered in the same framework. Speaker discrimination from telephone speeches was studied in [6] using HMM segmentation. However, in this system, the number of speakers was limited to two. K. Mori [12] addressed the problem of detection of speaker changes and speaker clustering without prior knowledge available. The speaker grouping information was used in speaker adaptation for speech recognition. Chen [13] also presented an approach to detecting changes in speaker identity, environmental condition, and channel conditions using Bayesian Information Criteria (BIC). A segmentation accuracy of about 80% is reported. Couvreur [16] presented a first approach to build an automatic system for broadcasting news speaker-based segmentation. The system is developed in the framework of the

Multimedia'02, December 1-6, 2002, Juan-les-Pins, France.

THISL project based on a Chop-Recluster method. Sonmez [17] and Bonastre [18] also reported their work on speaker tracking and detection system.

The previous efforts to solve the problem of unsupervised speaker segmentation and tracking consist of clustering audio segments into homogeneous clusters according to speaker identity, background conditions, or channel conditions. Most of the methods are based on VQ, GMM or HMM model. A disadvantage of these models is that iterative operation is unavoidable, which makes these algorithms very time consuming and not suitable for real-time processing.

In this paper, we will present an algorithm on real-time speaker change detection and speaker tracking in news broadcasting. We consider a more general case: both speaker identity and speaker number are assumed to be unknown. Further, real-time processing is the objective of the proposed algorithm.

There are many difficulties in real-time speaker segmentation and speaker tracking:

(1) Because of no prior knowledge of speakers, there is no training data to obtain an accurate speaker model *a priori*. Thus, the speaker data should be obtained from speech stream gradually, and the speaker model should be established incrementally. As the process accumulates more data, the speaker model should become more accurate.

(2) Because of real-time limitation, it is not affordable to use complex audio features and complex clustering methods. For example, when training GMM model, EM algorithms are not feasible since they are usually so time-consuming that it could not be processed in real-time.

(3) Also with the real-time requirement, the speaker identity needs to be estimated with little or no delay after the speaker changes are detected. This is challenging since there is little training data to model the new speaker when a speaker change occurs.

(4) The environment in news broadcasting audio is so complex that channel or environment mismatch remains a challenging problem.

The rest of the paper is organized as follows; Section 2 discusses system framework. Section 3 presents the selected features and distance metrics. Detailed description on segmentation and clustering scheme is presented in Section 4. Section 5 describes the speaker tracking scheme. In Section 6, experiments and the evaluations of the proposed algorithms are given.

2. SYSTEM OVERVIEW

The flow diagram of our proposed real-time unsupervised speaker detection and tracking algorithm is illustrated in Fig. 1. It is mainly composed of four modules: front-end process module, segmentation module, clustering and speaker model updating module and the speaker tracking module. It is assumed that the input audio stream is speech only--non-speech segments of the input audio have been filtered. In our system, non-speech audio segment filtering is performed by the audio segmentation and classification algorithms presented in [15][23].

The input speech stream is first segmented into 3-second sub-segments with 2.5-second overlapping. These 3-second sub-segments are used as the basic units in speaker segmentation and tracking. Each sub-segment is pre-processed by removing silence and unvoiced frames. Then, speaker change detection is performed. If there is no speaker change between two sub-segments, the model of current speaker is updated by incorporating data of the current sub-segment. Otherwise, if there is a speaker change, the algorithm searches the speaker model database to identify the new appeared speaker. If the speaker could not be found in the database, it is registered as a new speaker and is added into the database. If the speaker is found, the speaker model in database will be updated by new speaker data. The speaker model database is setup gradually in this way.

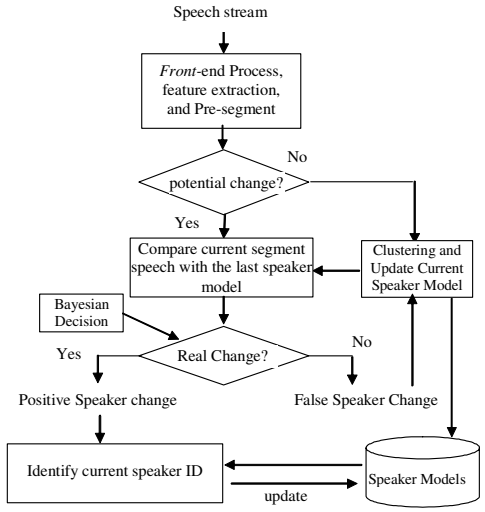

Fig. 1 A brief flow diagram for speaker change detection and speaker tracking

3. FEATURE SELECTION AND DISTANCE MEASURE

In this section, we discuss the problem of feature selection and distance measurement. In the previous research works, many features were proposed for speaker recognition systems. The most widely used features are LPC [3][7], MFCC [20], and LSP[1]. Other features are also used in some research works [22]. Meanwhile, CMS or RASTA processing [7][9][10] is processed on the feature vectors to remove the channel convolutional effects.

Different feature shows different performance in different speaker recognition systems. It is difficult to determine which feature is the best for all purposes. However, different features can complement each other in different contexts. Considering this fact, we will fuse multiple features to improve the performance of our system.

Our approach of feature fusion is based on feature discrimination power analysis. We first measure the distance of different feature set between two speaker models. The distances are then

Bayesian fused to get a more reliable result, as discussed in Section 4.4.

3.1 Feature Selection

Linear Spectral Pairs (*LSP*) and Mel-Frequency Cepstrum Coefficient (MFCC) are used in our speaker tracking system. In general, MFCC and LSP have similar performance on speaker recognition. But they performed differently in some specials cases. In other words, these two features can complement each other to improve the performance of speaker segmentation and tracking algorithm.

Besides these two features, pitch information is also useful. It is a much more direct feature to discriminate man and woman. So it is also considered in our system and used as assistant features.

In news broadcasting, the environment and channel are variable and very complex to estimate. They will affect the performance severely. How to compensate the effect of the channel or environment mismatch remains a difficult issue in speaker recognition research. CMS (Cepstral Mean Subtraction) is used in our algorithm. Actually, CMS alone is not sufficient as proved by many researches.

LSP, MFCC and pitch will be Bayesian Fused to get a more reliable result in our algorithm.

3.2 Distance Measure

Suppose the feature vector extracted from each sub-segment is Gaussian, its probability distribution function (*pdf*) can be represented as follows:

$$p(\xi) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{C}|^{1/2}} \exp\left\{-\frac{1}{2}(\xi-\boldsymbol{\mu})^T \boldsymbol{C}^{-1}(\xi-\boldsymbol{\mu})\right\} \quad (1)$$

where $\boldsymbol{C}$ is the estimated covariance matrix and $\boldsymbol{\mu}$ is the estimated mean vector.

Then the distance between two speech segments can be defined as [1],

$$D = \int_{\xi}[p_i(\xi) - p_j(\xi)]\ln\frac{p_i(\xi)}{P_j(\xi)}d\xi \quad (2)$$

Under the assumption that feature probability distribution functions are n-variable normal populations, (2) can be derived into,

$$\begin{aligned} D &= \frac{1}{2}tr[(\boldsymbol{C}_i - \boldsymbol{C}_j)(\boldsymbol{C}_j^{-1} - \boldsymbol{C}_i^{-1})] \\ &+ \frac{1}{2}tr[(\boldsymbol{C}_j^{-1} + \boldsymbol{C}_i^{-1})(\boldsymbol{u}_i - \boldsymbol{u}_j)(\boldsymbol{u}_i - \boldsymbol{u}_j)^T] \end{aligned} \quad (3)$$

The distance is composed of two parts. The first part is determined by the covariance of two sub-segments, and the second is determined by the covariance and mean. Since the mean is easily biased due to various environment conditions, we will not consider the second part. So, only the first part is used to represent the distance, based on the work [1]. It is also similar to the *CMS* method to compensate the effect of channel mismatch.

The final distance is called divergence shape distance, which is defined by

$$D = \frac{1}{2}tr[(\boldsymbol{C}_i - \boldsymbol{C}_j)(\boldsymbol{C}_j^{-1} - \boldsymbol{C}_i^{-1})] \quad (4)$$

In general, if two speech clips are said by the same speaker, the distance between them would be small; otherwise, the distance would be large. So, here is a simple criterion: if the distance between two speech segments is larger than a threshold, these two segments could be considered as being said by different speakers.

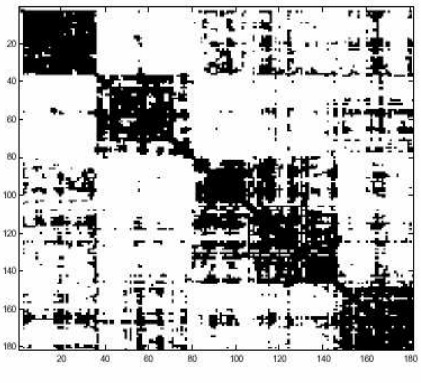

Fig. 2. A *LSP divergence distance* map

To show the effectiveness of the selected feature and distance measure, Fig. 2 illustrates a LSP diverge distance map for a 180-second-long speech segment. The dissimilarity between any two sub-segments is calculated. One threshold is used to transform the distance to a binary value (0, 1) based on the above simple criterion. Value 0 is represented by black pixel, and value 1 is represented by white pixel. It can be clearly seen that the figure is symmetric, and there are four speakers existing in this speech segment. MFCC also shows the similar characteristics.

4. SPEAKER CHANGE DETECTION

As illustrated in Fig.1, the real-time unsupervised speaker change detection algorithm is mainly composed of three modules. They are front-end process module, segmentation module, and the clustering and speaker model updating module. In the front–end processing, features are extracted from each sub-segment. Then, *LSP/MFCC/Pitch* divergence distance between every consecutive two sub-segments is examined. A potential speaker change point is detected if the distance is above a threshold. Otherwise, the model of current speaker is updated by incorporating the data of current sub-segment. If a potential speaker change boundary is detected, Bayesian Criteria are used to confirm if it is really a speaker change boundary.

4.1 Front-End Processing

The input audio stream is first down-sampled into a uniform Format: 8KHZ, 16bits, mono channel, whatever the input format is. The speech stream is then pre-emphasized and divided into

sub-segments by 3-second window with 2.5-second overlapping. That is, the basic processing unit is 3-second and the temporal resolution of the segmentation is 0.5 second. The sub-segment is further divided into non-overlapping 25ms-long frames. The most important features extracted from the frame are *LSP, MFCC*, and pitch. Other features extracted include short-time energy and zero-crossing rate; they are used to discriminate silence frames and unvoiced frames, which should be excluded in estimating speaker model.

4.2 Detect Potential Speaker Change Point

At this step, a simple speaker model is estimated for each sub-segment. The speaker model composes of three features: LSP, MFCC and pitch. Supposing each feature satisfies Gaussian distribution, the speaker model for *i*-th sub-segment can be represented as $\{N_F(C_i, u_i)\}$, where *F* represents different feature. In the rest of paper, we will omit the subscript *F* for simplicity and express speaker model as $N(u, C)$, since each feature can be processed in the same way. Divergence shape distance is used to measure the dissimilarity between each two neighboring sub-segments at each time slot as shown in Fig. 3 (a). At this step, only LSP divergence distance is used to detect potential speaker change point.

Thus, a potential speaker change is found between *i*th and *(i+1)*th sub-segment, if the following conditions are satisfied:

$$\begin{aligned} &D(i,i+1) > D(i+1,i+2), \\ &D(i,i+1) > D(i-1,i), \\ &D(i,i+1) > Th_i \end{aligned} \tag{5}$$

where *D(i, j)* is the distance between *i-th* sub-segment and *j-th* sub-segment, Th_i is a threshold.

The first two conditions guarantee that a local peak exists, and the last condition can prevent very low peaks from being detected. Reasonable results can be achieved by using this simple criterion. But the threshold is difficult to set in advance. The threshold setting is affected by many factors, such as insufficient estimate data and various environmental conditions. For example, the distance between speech sub-segments will increase if the speech is in a noisy environment. Accordingly, the threshold should increase in a noisy environment. To obtain optimal result, an automatic threshold setting method has been proposed as the following:

$$Th_i = \alpha \cdot \frac{1}{N} \sum_{n=0}^{N} D(i-n-1, \quad i-n) \tag{6}$$

where *N* is the number of previous distances used for predicting threshold, and α is a coefficient as amplifier. That is, threshold is automatically set according to the previous *N* successive distances. We set $\alpha = 1.2$ in our algorithm. The threshold determined in this way works well in various conditions, but false detections still exist. This is because there is no sufficient data to estimate the speaker model accurately from only one short speech sub-segment, such that the estimated speaker model would be biased.

To solve this problem, we use as much data as possible to update the speaker model. A more accurate refinement method has also been proposed to refine the above results.

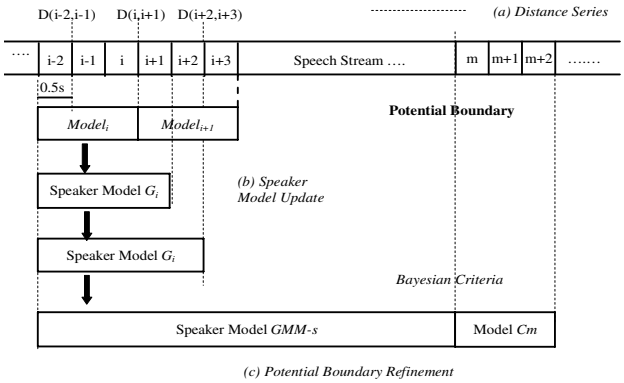

Fig. 3 An step by step illustration of speaker change detection algorithm

4.3 Incremental Speaker Model Updating

In order to get as much data as possible to estimate speaker model more accurately, we utilize the results of potential speaker change detection. If no potential speaker change point is detected, it means the current sub-segment is from the same speaker as the previous one. Thus, we update the current speaker model using this available new data, as illustrated in Fig. 3 (b).

GMM (Gaussian Mixture Model)-32 is used to model a speaker. The model is established progressively as more and more data become available. At the beginning, there is no sufficient speaker data to accurately estimate a GMM-32 model; thus, the GMM-1 is used. With the increase of the speaker data, the model will grow up to GMM-32 gradually.

A standard EM algorithm can be used to estimate the Gaussian Mixture Models. However, the EM algorithm employs recursive process, that it is usually time consuming, thus does not meet the real-time requirement. Furthermore, the EM algorithm requires all feature data to be saved in the memory or disk. Therefore, we have introduced an alternative clustering method which is less time consuming. Although its accuracy is not as high as EM algorithms, it works well in our applications.

Suppose that the current speaker model G_i is obtained from the previous *(M-1)* sub-segments and there is no potential speaker change point between *(M-1)*th and *M*th speech sub-segment, it means these two segments belong to the same speaker. Thus, we update the current speaker model G_i using the feature data of the *M*th sub-segment.

If the current model G_i is represented by $N(u, C)$, in which the number of feature vectors used is *N*; and the model obtained from *M*th sub-speech segment is $N(u_m, C_m)$, in which the number of feature vectors is N_m. It could be easily derived that the current speaker model could be updated by the following method

$$\mu' = \frac{N}{N+N_m}\mu + \frac{N_m}{N+N_m}\mu_m \tag{7}$$

$$C' = \frac{N}{N+N_m} C + \frac{N_m}{N+N_m} C_m + \frac{N \cdot N_m}{(N+N_m)^2} (\mu - \mu_m)(\mu - \mu_m)^T \quad (8)$$

$$N' = N + N_m \quad (9)$$

The third part of (8) is determined by the means which are easily biased by environment conditions. Thus, in practice, we ignore the mean part of (8) to compensate the effect of various environment conditions or transmission channel. Then (8) is simplified as

$$C' = \frac{N}{N+N_m} C + \frac{N_m}{N+N_m} C_m \quad (10)$$

The above procedure is looped till the dissimilarity between the speaker models before and after updating is small enough or a potential speaker change point is met. The dissimilarity is also measured by the divergence shape distance. When the dissimilarity is sufficiently small, it is assumed that the current Gaussian model is estimated accurately; i.e, it is not necessary to continue updating G_i. The next Gaussian model, G_{i+1}, is initiated and updated with the new data using the same method.

For one speaker, it will have several Gaussian Models estimated by the above method. Combining these Gaussian Models would form a quasi-Gaussian Mixture Model. The weight of each Gaussian model is set by their corresponding number of training data.

Suppose the speaker model is GMM-s, in which each Gaussian model G_i is $N(u_i, C_i)$ and the number of feature vectors used to estimate the G_i is N_i, $(i = 1,\ldots s)$, then the weight w_i of the ith Gaussian Model G_i is computed as $w_i = N_i / N$, where $N = \sum_{i=1}^{S} N_i$ is the total number of feature vectors.

By using the above method, speaker model will grow from GMM-2, GMM-3, …, up to GMM32. When the GMM32 is reached, the updating of the speaker model is stopped. This method (quasi-GMM) uses segmental clustering and it is a little bit different from the original GMM. It is less accurate than Gaussian Mixture Model obtained using EM algorithms, but it is computationally simple and meets our real-time processing requirements, while it is able to achieve reasonable accuracy, which could be seen from our experiments results presented later in the paper.

4.4 Speaker Change Boundary Refinement

Often there are false positives in potential speaker change points obtained with the algorithms described above. To remove false positives, a refinement algorithm is applied. The algorithm is based on the dissimilarity between the current sub-segment and the last speaker model obtained from the sub-segments before the current potential boundary. In this step, Bayesian Decision is used for potential boundary refinement, as shown in Fig. 3 (c).

Suppose at the potential speaker boundary, the model of the last speaker is GMM-s, in which each Gaussian model is $N(u_i, C_i)$ $(i = 1,\ldots s)$ and the model of current segment is $N(u_m, C_m)$; then the distance between them is roughly estimated as the weighted sum of the distance of $N(u_m, C_m)$ and each $N(u_i, C_i)$.

$$D = \sum_{i=1}^{S} w_i \cdot D(\boldsymbol{C}_i, \boldsymbol{C}_m) \quad (11)$$

LSP distance, MFCC distance, and Pitch distance are calculated respectively according to the above method. Then, they are input to a Bayesian decision engine, where these features are fused. Fig. 4 shows the Bayesian process to fuse multiple features.

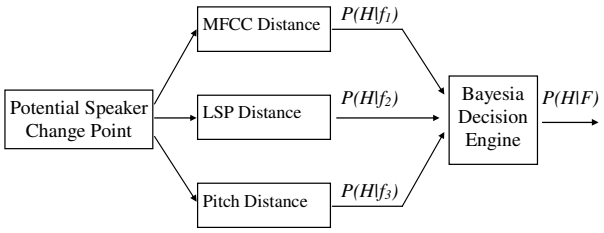

Fig. 4 A Bayesian fusion model

In Fig. 4, each feature gives the probability of the hypotheses. Then the three probabilities are fused by a Bayesian decision engine to get a final decision.

Assuming that each feature f_i $(i = 1\ldots N)$ is independent of each other (although this assumption is not always true in real world), based on Bayesian inference [26][27], the fusion can be given:

$$P(H \mid F) = P(H)^{1-N} \prod_{i=1}^{N} P(H \mid f_i) \quad (12)$$

Thus, a basic hypothesis test can be used to refine the potential speaker change point:

H_0: *It is a true speaker change point*

H_1: *It is not a true speaker change point*

The optimum test to decide between these two hypotheses is a likelihood ratio test given by

$$\lambda = \frac{P(H_0 \mid F)}{P(H_1 \mid F)} \begin{cases} \geq \lambda_0 & accept\ H_0 \\ < \lambda_0 & accept\ H_1 \end{cases} \quad (13)$$

If a potential candidate is not a real speaker change boundary, the current speaker data is still used to update the speaker model following the method described above.

5. SPEAKER TRACKING

When a speaker change boundary is detected, the next step is to identify the new appeared speaker. The new speaker may or may not have registered in the speaker model database, which means this step is like speaker identification and rejection process.

The current sub-segment will be compared with all the speaker models existed in a database to find which model is most similar to the current sub-segment. The most similar model is the most possible speaker of the current sub-segment.

The dissimilarity between the verifying speaker model and current sub-segment is set as weighted distance sum of K nearest Gaussian Model in the speaker model, but not the weighted sum of all model components as (11):

$$D^{'} = \underset{i}{KNN}\{D(\boldsymbol{C}_i, \boldsymbol{C}_m)\} \tag{14}$$

This is because that the speaker data in a speaker model may be from a various environment or channel, weighted sum of all of them is not useful and will reduce performance. But in the speaker change boundary refinement, the speaker model is estimated from the last speaker change point. We can assume that there is no environment/channel change in this duration.

Then, for each speaker model, we will have a hypothesis test:

H_{i0}: *The current sub-segment is speaker-i*

H_{i1}: *The current sub-segment is not speaker-i*

The likelihood ratio for speaker-*i* can be given by

$$\lambda_i = \frac{P(H_{i0} \mid F_i)}{P(H_{i1} \mid F_i)} \tag{15}$$

Then the most possible speaker models could be found according to the biggest likelihood ratio. If the largest likelihood ratio is larger than a threshold, the speaker of current segment is identified and the speaker mode is updated with the new data; otherwise, current segment is considered from a new speaker which will be registered into the database. In this way, we can determine the identity of the current speaker.

Suppose up to now there are *K* speakers registered in the speaker model database, the concrete expression to identify the speaker of current segment is as follows;

$$ID = \begin{cases} \arg\max_i \lambda_i & \text{if } \max \lambda_i \geq \lambda_0 \\ K+1 & \text{if } \max \lambda_i < \lambda_0 \end{cases} \tag{16}$$

where $1 \leq i \leq K$, and $K+1$ represents a new speaker. Suppose that each speaker appears with equal probability, λ_0 can be set as 1. The threshold could be also set experimentally according to application context.

In fact, it is difficult to identify the speaker of the current sub-segment immediately after the speaker change point is detected. This is because the data to describe the current speaker model is only 3 second, which is not sufficient to model an accurate speaker model. In our implementation, we experimented three ways of making identification decision, corresponding to three different amounts of data used in making the decision:

(1) Perform identification immediate after a speaker change point is detected. That is, only the data of current sub-segment is used in making identification;

(2) Perform identification until the next potential speaker change point is detected. That is, the data of current and previous sub-segments are used in making identification. However, the potential change point may not be a true boundary, thus, may add noise to the decision;

(3) Perform identification until the next speaker change point is confirmed. That is, the information used in refinement is also used in making the decision compared to case 2.

There is the least data available in Case (1) and the most data available in Case (3). The experiments presented in the next section prove that Case (3) achieve the highest performance. However, Case (3) also introduces the longest delay in decision making thus may not be desirable in some applications with strict real-time requirement.

6. EXPERIMENTS

In this section, we present the evaluation results of the proposed real-time speaker change detection and speaker tracking algorithms.

6.1 Database Information

The evaluations of the proposed speaker change detection and speaker tracking algorithms have been performed on the Hub-4 1997 English Broadcast News Speech Database. The database composes of about 97 hours news broadcasting, which are from different radios, such as CNN, ABC, CRI and C-SPAN. In our application, we only used the news broadcasting from *ABC World News Tonight* and *CNN Headline News*, which is about 10 hours in total. Half of our data is selected randomly for training and the other half is for testing. In testing database, each speech file is about 30 minute long, and there are about 30 speakers and about 60-80 speaker changes in each file.

Though this database is originally designed for Spoken Document Retrieval, it is more suitable for our intended application: speaker tracking for news broadcasting. The ground truth is obtained from its accompanying transcripts.

As we mentioned in previous sections, we use 3 second sub-segments as our basic identification unit. This unit size has been determined from the statistics of our experiments. This size is critical since if it is too short, there will be insufficient data to estimate an accurate speaker model; otherwise, if it is too long, two speakers' speech may intervene resulting in inaccurate speaker model estimation, and it will also introduce more delay.

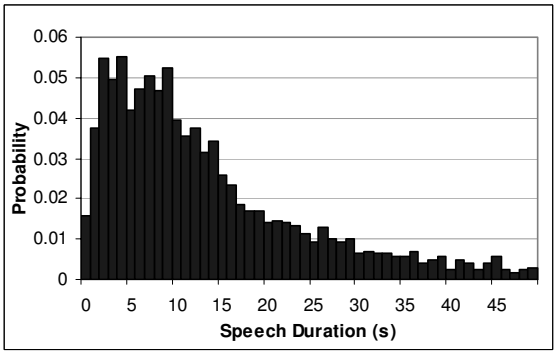

Fig. 5 The histogram for the length of speaker segment

Fig. 5 shows a histogram for the length of speaker segment in the training database. It is seen that there are about 5% speaker segments are less than or equal to 2 seconds, and 10% less than 3 seconds. We tested the performance with the sub-segment being 2 seconds or 3 seconds, it was observed that the performance decreased dramatically when sub-segment is 2 second. Hence, we selected 3 seconds as a sub-segment unit size. That is to say that for those speaker segments which are less than 3 second, the segmentation and tracking results are not reliable.

6.2. Speaker Segmentation

Fig. 6 shows an example of speaker change detection on 176-second long speech. The speech segment includes four speaker change boundaries, which are 17s, 52s, 86s, 154s respectively. Fig. 6 shows the initial LSP distance between each two speech sub-segments, the adaptive threshold, and the potential speaker change boundaries. It can be seen that the potential boundaries are much more than the real boundaries. Bayesian Decision is performed on these potential speaker change points to remove the false ones.

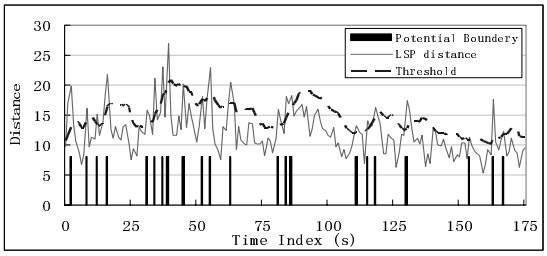

Fig. 6 An example of speaker change detection algorithm

Recall and Precision is used to evaluate the performance of proposed speaker segmentation algorithm. As mentioned before, since the smallest unit used to cluster a speaker model is 3s, our algorithm is not very good at detecting the speaker change point if the speaker segment is very short. The recall for short segments (<3s) is only 37.6%. For the long segments (>3s), the performance is much higher. The following reported results are all based on the long segments. Fig.7 shows the Recall-False curve achieved by the proposed speaker segmentation algorithm, where False = 1 – Precision. It is seen that when the false alarm is 15%, the recall is above 85%. Fig. 7 also shows the tradeoff between recall and false alarm, and provides a reference to select different recall and false alarm for different applications.

Because there is no ground truth on speech/non-speech information in the test database, an audio type classifier was used to segment them. Unfortunately, there still exist some misclassifications. The detection results are strongly affected by the short non-speech segments which were misclassified as speech segments, especially those sounds intervening in speech, such as a burse of wind, laugh or applause. Many (larger than 50%) false alarms are caused for this reason.

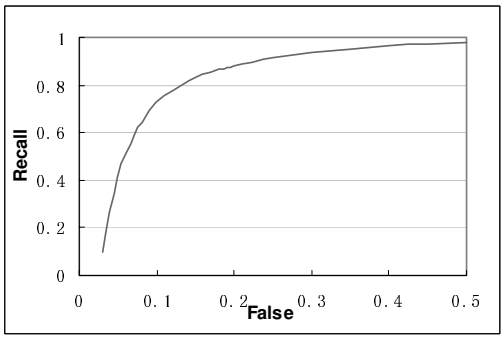

Fig. 7 The False-Recall curve for speaker segmentation

Since the algorithm is based on a resolution of 0.5 second and a context of 3-6 seconds, the detected speaker change point may shift from the true one. Another experiment was implemented to obtain the statistics of the shift information. Fig. 8 illustrates the shift histogram for those detected speaker change boundary.

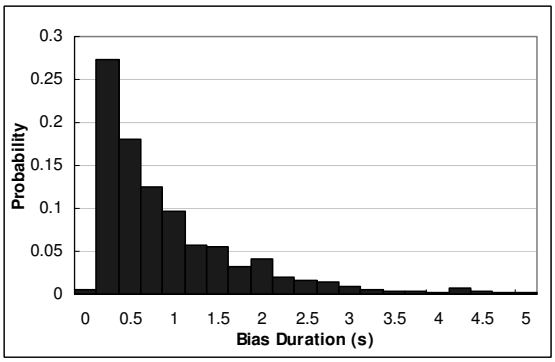

Fig. 8 the histogram of shift duration from true speaker change boundary

From Fig. 8, it can be seen near 70% detected boundaries are in less than 1 second away from the true boundary, and 86.6% are in less than 2 second away. Only 7.6% is out of 3 second neighborhood. This also proves that our algorithm has good resolution in speaker boundary detection.

It is very easily to increase the resolution from 0.5s to 0.1s, or even higher if we increase the overlapping of the shifting window. However, the computation complexity will increase linearly.

6.3. Speaker Tracking

In our approach, the tracking problem is considered as a retrieval problem. For a special speaker, he/she has a ground truth speech set and a detected speech set, thus recall and precision can be used to evaluate the tracking results. In our implementation, the average recall and the average precision are used to evaluate the tracking performance. Here, average recall and precision are calculated from a weighed sum of recall and precision of each speaker in the tested speech file.

Suppose there are N speakers in a test speech segment, and the recall and precision of *i-th* speaker is R_i and P_i, respectively, then

$$R_{avr} = \sum_{i=1}^{N} \alpha_i R_i$$
$$P_{avr} = \sum_{i=1}^{N} \alpha_i P_i \quad (17)$$

where α_i is weight for *i-th* speaker, and it can be calculated from:

$$\alpha_i = L_i \Big/ \sum_{i=1}^{N} L_i \quad (18)$$

where L_i is the speech length of *i-th* speaker.

As mentioned above, we tested three positions as to when to track the speaker. The first one is to perform speaker recognition instantly once a speaker change is found (*Instant*). The second one is to perform speaker recognition only after the next potential change is found (*Change*). The third one is to perform speaker recognition when the next real speaker boundary is detected (*Bound*). Fig. 9 illustrates the average false *vs.* average recall curve in these three cases, where $F_{avr} = 1 - P_{avr}$.

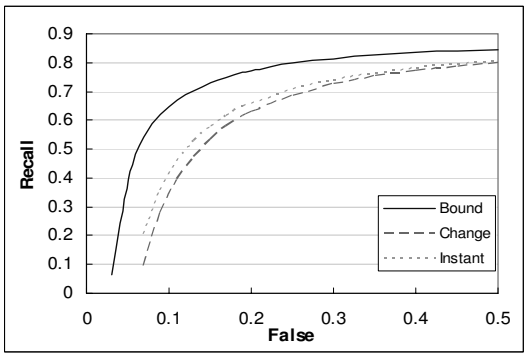

Fig. 9 the average false-recall curve for speaker tracking in three cases: (1) Instant: once at a real speaker change boundary. (2) Change: at the next potential change. (3) Bound: at the next real speaker change boundary

From Fig. 9, it can be seen that the third case has the most data to model the appearing speaker for speaker tracking, so its performance is the best among these three cases. The performances of *Change* and *Instance* are similar. When the false alarm is 0.1, the recall of *Bound* is 20% higher than the other two. And when the false alarm is 0.2, the recall of *Bound* is 12% higher.

In our experiments, there are about 30 speakers in each testing speech file. The higher the speaker number, the more confusion there will be in speaker identification. The performance of our algorithm can be increased if the speaker number decreases.

6.4 Computation Complexity

We have also tested the time complexity of our algorithm. With Pentium III 864MHz PC running Windows XP, the segmentation and tracking process can be completed in about 15% of the time-length of an audio clip. The LSP/MFCC correlation analysis is the most time-consuming part in our algorithm. After using an optimized function to compute correlation analysis, the time performance has been increased dramatically. Our scheme can totally meet the real-time processing requirements in multimedia application.

7. DISCUSSION AND CONCLUSION

In this paper, we have presented a novel approach on real-time unsupervised speaker segmentation and speaker tracking system. A two-step speaker change detection algorithm is proposed, which includes potential speaker change detection and refinement. Speaker tracking is based on the results of speaker change. A Bayesian Fusion method is used to fuse multiple features, which include MFCC, LSP, and pitch, to obtain a more reliable result. The algorithm achieves satisfactory accuracy. Although this system is for news broadcasting, the same algorithms could be used in other audio applications.

There are still rooms to improve the proposed approach. In particular, our future research will be focused in addressing the following issues.

First, in order to process in real-time, the available data to train speaker model is always limited. How to estimate an accurate model from small training data is still a challenge. Also, in the news broadcasting, the environment and context are so complex that the segmentation result is often affected. In the experiments, we have found that if there is a laugh burst between speeches, it is easily detected as speaker change boundary. Therefore, another future focus is on addressing this issue. Furthermore, Environment and channel variations also affect speaker tracking results. It is also found that the same speaker in various environments sometimes is detected as different ones. This indicates that our compensation for the mismatch effect of environment or channel is still insufficient.

8. REFERENCES

[1] J. P. Campbell, JR. "Speaker Recognition: A Tutorial". *Proceedings of the IEEE*, vl.85, No.9, pp.1437-1462, 1997.

[2] J. N. L. Brummer. "Speaker Recognition over HF Radio after Automatic Speaker Segmentation". *Proc. of the IEEE South African Symposium on Communications and Signal Processing, COMSIG-94*. pp. 171-176, 1994

[3] M. Sugiyama, J. Murakami, and H. Watanabe. "Speech Segmentation and Clustering Based on Speaker Features". *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*. 1993

[4] L. Wilcox, F. Chen, D. Kumber, and V. Balasubramanian. "Segmentation of Speech Using Speaker Identification". *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1994

[5] M. H. Siu, G. Yu, and H. Gish. "An Unsupervised, Sequential Learning Algorithm for the Segmentation of Speech Waveform with Multiple Speakers". *Proc. of IEEE*

International Conference on Acoustics, Speech, and Signal Processing, pp 189-192, 1992.

[6] A. Cohen and V. Lapidus, "Unsupervised Speaker Segmentation in Telephone Conversations". *Proc. of Nineteenth Convention of Electrical and Electronics Engineers,* Israel, pp.102-105, 1996.

[7] H. Gish and M. Schmidt. "Text-Independent Speaker identification". *Proc. of IEEE Signal Processing Magazine,* pp.18-32, Oct. 1994

[8] H. Gish, M. H. Siu and R. Rohlicek. "Segregation of Speakers for Speech Recognition and Speaker Identification". *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp873-876, 1991.

[9] H. Hermansky and N. Morgan. "RASTA Processing of Speech". *Proc. of IEEE Trans. on Speech and Audio Processing*, Vol 2, No.4, pp. 578-589, Oct.1994.

[10] R. J. Mammone, X. Y. Zhang and R. P. Ramachandran. "Robust Speaker Recognition: A Feature-Based Approach". *Proc. of IEEE Signal Processing Magazine,* pp.58-71, Sept. 1996.

[11] H. A. Murthy, F. Beaufays, L. P. Heck and M. Weintraub. "Robust Text-Independent Speaker Identification over Telephone Channels". *Proc. of IEEE Trans. on Speech and Audio Processing*, Vol. 7, No.5, pp. 554-568, Sept.1999.

[12] K. Mori and S Nakagawa. "Speaker change Detection and Speaker Clustering Using VQ Distortion for Broadcast news Speech Recognition". *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing,* 2002.

[13] S. Chen and P. S. Gopalakrishnan. "Speaker, Environment and Channel Change Detection and Clustering via the Bayesian Information Criterion". *Proc. of DARPA Broadcast News Transcription and Understanding Workshop*, 1998.

[14] G. Schwarz. "Estimation the Dimension of a Model". *The Annals of Statistics*, Vol.6, pp. 461-464, 1978

[15] L. Lu, H. Jiang and H. J. Zhang, "A Robust Audio Classification and Segmentation Method". *Proc. of 9th ACM Multimedia*, pp203-211, 2001.

[16] L. Couvreur and J. M. Boite. "Speaker Tracking in Broadcast Audio Material in the Framework of the THISL Project". *Proc. of the ESCA ETRW workshop Accessing Information in Spoken Audio*, pp. 84-89. 1999.

[17] K. Sonmez, L. Heck and M. Weintraub. "Speaker Tracking and Detection with Multiple Speakers". *Proc. Eurospeech'1999*, vol. 5, pp. 2219-2222, Budapest, 1999.

[18] J. F. Bonastre, P. Delacourt, C. Fredouille, T. Merlin and C. Wellekens. "A Speaker Tracking System Based on Speaker Turn Detection for NIST Evaluation". *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp.1177-1180, 2000

[19] C. Fredouille, J. F Bonastre and T. Merlin. "Segmental normalization for Robust Speaker Verification". *Workshop on robust method for speech recognition in adverse conditions*, pp. 103-106, 1999.

[20] D. A. Reynolds, T. F. Quatieri and R.B. Dunn. "Speaker Verification Using Adapted Gaussian Mixture Models". *Digital Signal Processing,10*, pp.19-41, 2000.

[21] M. Padmanabhan, L. R. Bahl, D. Nahamoo and M. A. Picheny. "Speaker Clustering and Transformation for Speaker Adaptation in Speech Recognition Systems". *IEEE Transaction on Speech and Audio Processing*, Vol. 6, No. 1, pp.71-77, Jan. 1998.

[22] B. L. Berg and A. A. Beex. "Investigating Speaker Features from Very Short Speech Records". *Proc. of IEEE Int'l Symposium on Circuits and Systems, ISCAS'99*, Vol. III, pp. 102-105, 1999.

[23] L. Lu, S. Z. Li and H.-J. Zhang, "Content-Based Audio Segmentation Using Support Vector Machines". *Proc. of ICME01*, pp956-959, 2001

[24] D. Roy and C. Malamud. "Speaker Identification based Text to Audio Alignment for an Audio Retrieval System". *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing,* pp.1099-1102, 1997

[25] D. G. Kimber, L. D. Wilcox, F. R. Chen, and T. P. Moran. "Speaker Segmentation for Browsing Recorded Audio". *ACM CHI'95 Mosaic of Creativity*, pp212-213, 1995

[26] L. Wang and K. L. Chan. "Bayesian Fusion: an Approach for Image Retrieval using Multiple Features". *Proc. of International Conference on Image and Vision Computing '00*, Hamilton, New Zealand, 2000.

[27] M. A. Abidi and R. C. Gonzalez. "Data fusion in robotics and machine intelligence". *Boston Academic Press*. 1992