BIC-based Speaker Segmentation Using Divide-and-Conquer Strategies with Application to Speaker Diarization

Shih-Sian Cheng, Hsin-Min Wang, *Member, IEEE* and Hsin-Chia Fu, *Member, IEEE*

***Abstract*—In this paper, we propose three divide-and-conquer approaches for BIC-based speaker segmentation. The approaches detect speaker changes by recursively partitioning a large analysis window into two sub-windows and recursively verifying the merging of two adjacent audio segments using ΔBIC, a widely-adopted distance measure of two audio segments. We compare our approaches to three popular distance-based approaches, namely, Chen and Gopalakrishnan's window-growing-based approach, Siegler *et al.*'s fixed-size sliding window approach, and Delacourt and Wellekens's DISTBIC approach, by performing computational cost analysis and conducting speaker change detection experiments on two broadcast news data sets. The results show that the proposed approaches are more efficient and achieve higher segmentation accuracy than the compared distance-based approaches. In addition, we apply the segmentation approaches discussed in this paper to the speaker diarization task. The experiment results show that a more effective segmentation approach leads to better diarization accuracy.**

***Index Terms*—speaker segmentation, speaker change detection, Bayesian Information Criterion, divide-and-conquer, speaker diarization**

I. Introduction

The goal of speaker (audio) segmentation is to detect speaker (acoustic) change boundaries in an audio stream. In the last decade, researchers in the speech processing community have expended a great deal of effort on this problem because of its application to many speech and audio processing tasks, such as audio classification [1], [2], automatic transcription of audio recordings [3], [4], speaker tracking [5], [6], and speaker diarization [7], [8].

Existing audio segmentation approaches generally fall into two categories, namely, distance-based segmentation [9], [10], [11], [12], [13] and model-decoding-based segmentation [4], [11]. In distance-based segmentation, a distance measure of two audio segments is defined first, and then an acoustic change detection strategy is designed based on the distance measure. In contrast to model-decoding-based segmentation, which detects acoustic changes in a supervised manner, distance-based segmentation has the advantage that acoustic changes can be detected in an unsupervised manner, i.e., *a priori* knowledge about the content of the input audio stream is unnecessary. In this paper, we focus on distance-based segmentation.

When low-level acoustic features like mel-scale frequency cepstral coefficients (MFCCs) are used in distance-based segmentation, the distance measure is usually derived from a statistical modeling framework. More precisely, it is assumed that the feature vectors in each of the two audio segments arise from some probability distribution (e.g., the multivariate Gaussian distribution); then, the distance between the two segments is represented by the dissimilarity between the two distributions. Several distance measures have been proposed, e.g., the Kullback-Leibler distance (KL or KL2) [10], the Generalized Likelihood Ratio (GLR) [9], [14], ΔBIC [11], [15], [16], [13], [17], [18], the Bhattacharyya distance [12], and the XBIC [19]. In addition, some high-level features have been used for audio segmentation; e.g., the spectrum flux and zero-crossing rate (ZCR) [20], [21], and the smoothed zero-crossing rate (SZCR) [22].

Window-growing-based segmentation (WinGrow) [11], [15], [23], [17], fixed-size sliding window segmentation (FixSlid) [10], [12], [24], [25], [26], and DISTBIC [9] are three popular distance-based segmentation approaches.

1) The WinGrow approach was first proposed by Chen and Gopalakrishnan [11]. For the distance measure of two audio segments, they used the Bayesian Information Criterion (BIC) [27], [28] to evaluate the following two hypotheses: 1) the union of the feature vectors of the two segments forms a Gaussian cluster in the feature space, and 2) the feature vectors of each segment form a distinct Gaussian cluster. Then, the difference between the two evaluation scores, ΔBIC, is used as the distance measure. In the acoustic change detection procedure, a small analysis window is put at the beginning of the audio stream initially. If no change point is detected in that analysis window, the search range is increased. The decent segmentation accuracy of this approach is widely recognized; however, as the window size grows, it incurs a heavy computational cost due to numerous ΔBIC computations, especially when the audio stream contains many long homogenous segments. To reduce the computational cost, Tritschler and Gopinath [29] proposed

Manuscript received January 10, 2009; revised May 06, 2009. This work was supported in part by the National Science Council of Taiwan under Grants NSC96-2628-E-001-024-MY3 and NSC98-2631-001-013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mark Hasegawa-Johnson.

S.-S. Cheng is with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C., and also with the Institute of Information Science, Academia Sinica, Taipei, Taiwan, R.O.C. e-mail: (sscheng@iis.sinica.edu.tw).

H.-M. Wang is with the Institute of Information Science, Academia Sinica, Taipei, Taiwan, R.O.C. e-mail: (whm@iis.sinica.edu.tw).

H.-C. Fu is with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C. e-mail: (hcfu@csie.nctu.edu.tw).

some heuristics that ignore the distance computations at locations where acoustic changes are unlikely to occur. Zhou and Hansen [17] used Hotelling's T^2-Statistic, which has the advantage of low computational cost, as the distance measure in the WinGrow detection process, and only used ΔBIC to verify the acoustic change candidates. In addition, [15] and [13] proposed more efficient implementations for the ΔBIC computation that do not affect the segmentation accuracy.

2) In FixSlid, a certain distance measure is used to evaluate the dissimilarity between two adjacent windows that slide along the audio stream to produce a distance curve; then, some heuristic thresholds are used to judge whether the locations of peaks are acoustic changes. To detect the change boundary associated with a short homogeneous segment, the size of the analysis window is usually set at a small value (e.g., two seconds). This is a dilemma because a small analysis window does not contain sufficient feature vectors to obtain a reliable distance statistic. For this approach, the KL2, GLR, and ΔBIC derived from the uni-Gaussian model are popular distance measures because they have the advantage of low computational cost; however, their effectiveness may be limited due to the limited generalization ability of uni-Gaussian. In [25], the authors proposed a bilateral scoring approach for calculating the distance between two segments based on adapted Gaussian mixture models (GMMs). Because of the good generalization ability of GMMs, this approach has been shown to be more effective than WindGrow and XBIC, which are developed on the basis of the uni-Gaussian model; however, it suffers from a higher computational cost due to the requirement for the adaptation of GMMs and calculation of mixture likelihoods in the distance measure.
3) Under the DISTBIC approach, the input audio stream is first segmented by FixSlid; then, the acoustic change candidates are verified sequentially by segment merging using ΔBIC. In practical use of this approach, FixSlid is usually applied to over-segment the audio stream to ensure a low miss detection rate at the cost of a high false alarm rate; then, the segment merging process is applied to reduce false alarms while maintaining the low miss detection rate. This approach is highly efficient. Moreover, it has been reported that this approach achieves decent segmentation accuracy [9]. The sequential segment merging process can be replaced by the hierarchical agglomerative clustering (HAC), which has been widely used in many speaker diarization systems [7], [8], [30]. However, it is not as efficient as DISTBIC because of the essential computational cost of HAC.

Although WinGrow is more efficient than the adapted-GMMs approach while maintaining high segmentation accuracy, the computational cost is still quite considerable when applying it to a large-scale task, e.g., the indexing of a database containing thousands of audio recordings. Therefore, more efficient segmentation approaches are desirable. In this paper, we propose three divide-and-conquer approaches for distance-based speaker segmentation. The first approach (DACDec1) detects speaker changes by recursively partitioning a large analysis window into two sub-windows at the position with the largest positive ΔBIC value obtained by Chen's one-change-point detection algorithm [11], rather than by applying a size-growing analysis window. All the divided points are output as change points. The second approach (DACDec2), which is a variant of DACDec1, recursively partitions a large analysis window into two sub-windows at the position with the largest ΔBIC value, no matter whether it is larger than zero or not. It then recursively verifies whether the divided points with negative ΔBIC values calculated in the division stage are speaker changes based on the new ΔBIC measurements of their left and right neighbor segments. The third approach (DACDec3) is a recursive variant of DISTBIC. It recursively partitions an audio stream at the locations of speaker change candidates obtained by FixSlid, and then recursively verifies those candidates based on the ΔBIC measurements of their left and right neighbor segments. To compare the performance of WinGrow, FixSlid, DISTBIC, the HAC-based approach and the proposed approaches, we conducted speaker change detection experiments on two broadcast news data sets, namely the MATBN corpus [31] and the broadcast news data in the 2003 NIST rich transcription evaluation data (RT03) [32]. For the efficiency comparison, we analyzed their computational costs and reported their respective run times in the experiments. The experiment results show that DACDec1 and DACDec2's recursive (top-down) multiple-change-point detection strategies are more effective and efficient than WinGrow's bottom-up multiple-change-point detection strategy. The results also show that, by providing a more effective and efficient segment merging process, DACDec3 outperforms DISTBIC and the HAC-based approach. Moreover, it achieves similar segmentation accuracy as WinGrow at a much lower computational cost. We applied the segmentation approaches discussed in this paper to the speaker diarization task, where the segmentation result is input to a HAC speaker clustering module. The experiment results on RT03 show that a more effective segmentation approach leads to better diarization accuracy.

The remainder of this paper is organized as follows. To help explain our proposed approaches, we review the ΔBIC distance measure and the WinGrow approach in Section II. We then present the proposed divide-and-conquer approaches for speaker segmentation in Section III. In Section IV, we analyze the computational costs of the baseline approaches and the proposed approaches. Section V details the experiments on speaker segmentation, Section VI details the application of the segmentation approaches to the speaker diarization task, and Section VII contains some concluding remarks.

II. Window-growing-based Segmentation

We review the ΔBIC distance measure of two audio segments in Section II-A and window-growing-based segmentation (WinGrow) in Section II-B.

A. ΔBIC as the distance measure of two audio segments

1) Model selection and BIC: Given a data set $\mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n\} \subset \mathbb{R}^d$ and a set of candidate models $\mathcal{M} = \{M_1, M_2, \cdots, M_k\}$, the purpose of model selection is to choose the model that best fits the distribution of $\mathcal{Z}$ from $\mathcal{M}$. When using the Bayesian Information Criterion (BIC) for model selection, the BIC value of M_i for $\mathcal{Z}$ is

$$BIC(M_i, \mathcal{Z}) = \log p(\mathcal{Z} \mid \hat{\mathbf{\Theta}}_i) - \frac{1}{2}\lambda \#(M_i) \log n, \quad (1)$$

where $\lambda = 1$, $\hat{\mathbf{\Theta}}_i$ is the maximum likelihood estimate of the parameter set of M_i, and $\#(M_i)$ is the number of parameters of M_i. The model with the largest BIC value will be selected.

2) ΔBIC as the distance measure: Given two audio segments represented by feature vectors, $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{n_x}\} \subset \mathbb{R}^d$ and $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n_y}\} \subset \mathbb{R}^d$, we evaluate the following two hypotheses [11]:

$$\begin{aligned} H_0 &: \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{n_x}, \mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n_y} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma}), \\ H_1 &: \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{n_x} \sim \mathcal{N}(\boldsymbol{\mu}_\mathcal{X}, \mathbf{\Sigma}_\mathcal{X}); \\ & \quad \mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n_y} \sim \mathcal{N}(\boldsymbol{\mu}_\mathcal{Y}, \mathbf{\Sigma}_\mathcal{Y}). \end{aligned} \quad (2)$$

H_0 posits that $\mathcal{X}$ and $\mathcal{Y}$ are derived from the same multivariate Gaussian, while H_1 posits that they are derived from two distinct multivariate Gaussians.

Let $\mathcal{Z} = \mathcal{X} \bigcup \mathcal{Y}$ and $n = n_x + n_y$. Then, the ΔBIC value can be computed as the difference between the BIC values of H_1 and H_0 as follows:

$$\begin{aligned} \Delta BIC_{\{\mathcal{X},\mathcal{Y}\}} &= BIC(H_1, \mathcal{Z}) - BIC(H_0, \mathcal{Z}) \\ &= \log p(\mathcal{X} \mid \hat{\boldsymbol{\mu}}_\mathcal{X}, \hat{\mathbf{\Sigma}}_\mathcal{X}) + \log p(\mathcal{Y} \mid \hat{\boldsymbol{\mu}}_\mathcal{Y}, \hat{\mathbf{\Sigma}}_\mathcal{Y}) \\ &\quad - \log p(\mathcal{Z} \mid \hat{\boldsymbol{\mu}}, \hat{\mathbf{\Sigma}}) - \frac{1}{2}\lambda(d + \frac{1}{2}d(d+1)) \log n \\ &= \frac{n}{2}\log|\hat{\mathbf{\Sigma}}| - \frac{n_x}{2}\log|\hat{\mathbf{\Sigma}}_\mathcal{X}| - \frac{n_y}{2}\log|\hat{\mathbf{\Sigma}}_\mathcal{Y}| \\ &\quad - \frac{1}{2}\lambda(d + \frac{1}{2}d(d+1)) \log n, \end{aligned} \quad (3)$$

where $\hat{\boldsymbol{\mu}}$, $\hat{\boldsymbol{\mu}}_\mathcal{X}$, and $\hat{\boldsymbol{\mu}}_\mathcal{Y}$ are, respectively, the sample mean vectors of $\mathcal{Z}$, $\mathcal{X}$, and $\mathcal{Y}$; $\hat{\mathbf{\Sigma}}$, $\hat{\mathbf{\Sigma}}_\mathcal{X}$, and $\hat{\mathbf{\Sigma}}_\mathcal{Y}$ are, respectively, the sample covariance matrices of $\mathcal{Z}$, $\mathcal{X}$, and $\mathcal{Y}$; and d is the dimension of the feature vector [13]. The larger the value of ΔBIC, the less similar the two segments will be; thus, the larger the distance between the two segments will be. When $\lambda = 0$, the ΔBIC distance between two segments is equivalent to the GLR distance [11], [33].

B. Window-growing-based segmentation

1) One-change-point detection: Let the feature vectors of the input audio stream be $\mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n\}$. In Chen and Gopalakrishnan's one-change-point detection algorithm [11] (denoted as OCD-Chen in this paper), it is assumed that there is at most one change point in $\mathcal{Z}$. Then, the $\Delta BIC_{\{\mathcal{X}_i,\mathcal{Y}_i\}}(i)$ value for $i_{min} < i \leq n - i_{min}$ is computed as

$$\begin{aligned} \Delta BIC_{\{\mathcal{X}_i,\mathcal{Y}_i\}}(i) &= \frac{n}{2}\log|\hat{\mathbf{\Sigma}}| - \frac{i}{2}\log|\hat{\mathbf{\Sigma}}_{\mathcal{X}_i}| - \frac{n-i}{2}\log|\hat{\mathbf{\Sigma}}_{\mathcal{Y}_i}| \\ &\quad - \frac{1}{2}\lambda(d + \frac{1}{2}d(d+1)) \log n, \end{aligned} \quad (4)$$

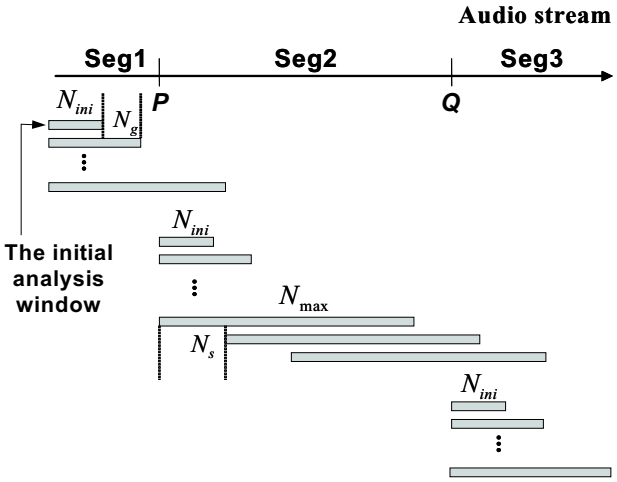


Fig. 1. Diagram of the multiple-change-point detection in window-growing-based segmentation (WinGrow). The audio stream contains three segments, namely Seg1, Seg2, and Seg3; P and Q denote the change points.

where $\hat{\mathbf{\Sigma}}$, $\hat{\mathbf{\Sigma}}_{\mathcal{X}_i}$, and $\hat{\mathbf{\Sigma}}_{\mathcal{Y}_i}$ are, respectively, the sample covariance matrices of $\mathcal{Z}$, $\mathcal{X}_i$=$\{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_i\}$, and $\mathcal{Y}_i$=$\{\mathbf{z}_{i+1}, \mathbf{z}_{i+2}, \cdots, \mathbf{z}_n\}$. If $max_{i_{min}<i\leq n-i_{min}} \Delta BIC_{\{\mathcal{X}_i,\mathcal{Y}_i\}}(i) > 0$, the time index corresponding to the maximum value is output as the change point; otherwise, there is no change point in $\mathcal{Z}$. It is not necessary to compute the ΔBIC value for time indices within the ranges 1 to i_{min} and $n - i_{min} + 1$ to n because in these cases the number of samples in $\mathcal{X}_i$ or $\mathcal{Y}_i$ is insufficient to give a reliable estimate of the parameters. Empirically, it is appropriate to set i_{min} at a value within the range 30 to 50 for practical applications. According to the BIC theory, the penalty factor λ in Eq. (4) is 1; however, in practical segmentation tasks, it is usually adjusted to allow a tradeoff between error types.

2) Multiple-change-point detection: OCD-Chen outputs at most one change point, even though there are multiple change points in the analysis window. To detect multiple change points in an audio stream, as shown in Fig. 1, OCD-Chen can be applied sequentially to a sliding, size-growing analysis window whose initial size is N_{ini} samples. The window repeatedly grows by N_g samples until a change point is detected or its size exceeds a pre-defined upper bound N_{max}. Here, the upper bound ensures the search efficiency [15], [13]. If a change point is detected during the window growing step, the detection process restarts at that change point with an analysis window of N_{ini} samples. When the size of the window grows to N_{max}, it is repeatedly shifted by N_s samples until a change point is detected or the analysis window reaches the end of the audio stream. In this way, the change points in the audio stream can be detected sequentially.

III. Divide-and-Conquer-based Segmentation

In this section, we present three implementations of the divide-and-conquer paradigm for detecting multiple change points in an analysis window. Note that the proposed approaches are based on the same assumption as that of WinGrow, i.e., the feature vectors of audio segments from

Algorithm 1 $CP \leftarrow$DACDec1(W)

Require: W: the analysis window
Ensure: CP: the set of change points detected in W
Begin
1) detect whether there is a change point in W by OCD-Chen;
2) //Check termination
if (there is no change point in W or the size of W is smaller than N_{min})
$CP \leftarrow \phi$; //empty set
goto End; //return
3) //Divide
let $\hat{t}$ be the change point detected in 1);
divide W into two sub-windows, W_1 and W_2, at $\hat{t}$;
4) //Solve sub-instances
$CP_{W_1} \leftarrow DACDec1(W_1)$; $CP_{W_2} \leftarrow DACDec1(W_2)$;
5) //Combine
$CP \leftarrow \hat{t} \cup CP_{W_1} \cup CP_{W_2}$;
End

different speakers are derived from different Gaussian distributions.

A. The DACDec1 approach

We use the example in Fig. 2 to explain the concept of divide-and-conquer-based segmentation. It is assumed that the audio stream in Fig. 2 (a) consists of three homogeneous segments derived from different speakers. Initially, OCD-Chen is applied in an analysis window that covers the entire audio stream. After the change point C_2 has been detected with the ΔBIC curve in Fig. 2 (b), the audio stream is divided into two analysis windows. Then, OCD-Chen is recursively applied in these two windows to search for the remaining change points so that C_1 can be detected. This approach, called DACDec1, allows us to detect the change points by a divide-and-conquer (DAC) strategy. As described in Algorithm 1, DACDec1 terminates (returns) if no change point is detected by OCD-Chen in the analysis window or the size of the analysis window is smaller than a pre-defined value, denoted as N_{min} samples. In the *Divide* stage, the analysis window is partitioned into two sub-windows at the change point detected by OCD-Chen. Then, the sub-windows are input to DACDec1 in the *Solve sub-instances* stage. Finally, the *Combine* stage outputs all the change points detected in step 1) and step 4) (i.e., the *Solve sub-instances* stage).

1) Discussion: In general, when the data samples are derived from more than one Gaussian distribution, two Gaussians (the H_1 hypothesis) fit the distribution of the data better than one Gaussian (the H_0 hypothesis) if the samples belonging to the same Guassian are used together to estimate the parameters. For example, Fig. 3 schematically illustrates a case where the three audio segments are derived from three different speakers and their feature vectors distribute as three Gaussian clusters. This case explains why the ΔBIC values at C_1 and C_2 in Fig. 2 (b) are positive. From the above perspective, if the homogeneous segments in the analysis window of DACDec1 are always derived from different speakers during the recursive process, we can be confident that, at each change point, the H_1 hypothesis will fit the data better than the H_0 hypothesis; thus, the ΔBIC value will be positive.

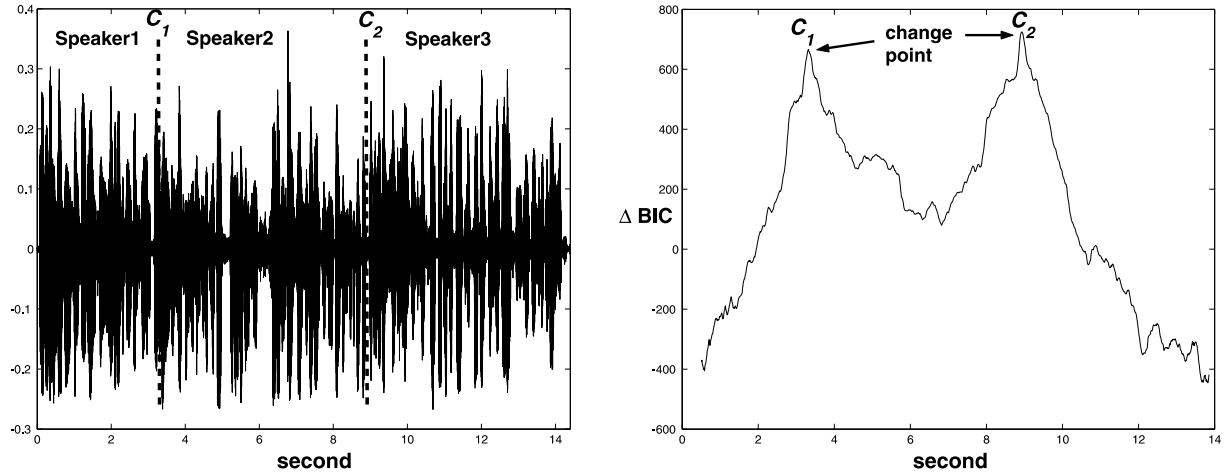
However, if two or more segments in the analysis window are derived from the same speaker, the performance of DACDec1 may decline dramatically. For example, in Fig. 4 (a), the first and third segments are derived from the same speaker (Speaker1), while the second segment is derived from another speaker (Speaker2). When applying OCD-Chen to the audio stream in Fig. 4 (a) with the same λ value of BIC used in the example in Fig. 2, we obtain the ΔBIC curve in Fig. 4 (b). The curve still has two peaks at the change points C_1 and C_2 because the H_1 hypothesis models the distribution of the data samples better at change points than it does at non-change points. We use Figs. 4 (c) and (d) to explain this perspective. Fig. 4 (c) diagrammatically illustrates the two hypotheses at C_2, where all the data samples of Speaker2 (the circles) are used with those of Speaker1 (the stars) to estimate one Gaussian in H_1. In contrast, at the non-change point R in Fig. 4 (b), as shown in Fig. 4 (d), the data samples of Speaker2 are divided into two parts, each of which is combined with the data samples of Speaker1 (one with the stars and the other with the diamonds) to estimate a distinct Gaussian in H_1. Clearly, the H_1 hypothesis in Fig. 4 (c) fits the data better than that in Fig. 4 (d).

In this example, we have peaks at C_1 and C_2. However, their ΔBIC values are negative, and no change point will be output by OCD-Chen because, as illustrated in Fig. 4 (c), H_1 over-fits the data samples of Speaker1 and obtains a smaller BIC value than that of H_0. We may adjust the value of λ so that, at C_2, the ΔBIC value will be positive (i.e., the hypothesis test favors H_1). However, this may result in false alarms when the recursive process continues to detect change points in a homogeneous segment. In other words, it is difficult to determine a reliable λ value for an audio stream like the example in Fig. 4 (a). Moreover, it is infeasible to adjust the value of λ for each specific audio stream in practical applications.

B. The DACDec2 approach

To overcome the performance limitation caused by unreliable ΔBIC measurements of the over-fitting cases in DACDec1, we developed an alternative implementation of the divide-and-conquer paradigm, called DACDec2. In this approach (Algorithm 2), the ΔBIC value is not used to check the termination in the *Check termination* stage because it may be unreliable, as illustrated in Figs. 2 and 4. The recursive process terminates (returns) when the size of the analysis window is smaller than N_{min} samples. In the *Divide* stage, the analysis window is partitioned into two sub-windows at the time index $\hat{t}$ that has the largest ΔBIC value located by OCD-Chen. Then, the sub-windows are input to DACDec2 in the *Solve sub-instances* stage. In the *Combine* stage, $\hat{t}$ is labeled as a change point if the ΔBIC value at $\hat{t}$ calculated in the $Divide$ stage is positive; otherwise, it needs to be verified using its two neighboring segments $\mathcal{X}$ and $\mathcal{Y}$. In the verification process, $\hat{t}$ is only labeled as a change point if $\Delta BIC_{\{\mathcal{X},\mathcal{Y}\}}(\hat{t}) > 0$.

Fig. 5 illustrates a recursive tree that simulates the recursive process of DACDec2 on the audio stream in Fig. 4 (a). We



(a) (b)

Fig. 2. (a) An audio stream comprised of three speech segments, each derived from a distinct speaker. C_1 and C_2 are the change points. (b) The ΔBIC curve obtained by applying OCD-Chen to the audio stream in (a).

Algorithm 2 $CP \leftarrow$DACDec2(W)

Require: W: the analysis window
Ensure: CP: the set of change points detected in W
Begin
1) //Check termination
if (the size of W is smaller than N_{min})
$CP \leftarrow \phi$; //empty set
goto End; //return
2) //Divide
apply OCD-Chen to W and let $\hat{t}$ be the time index with the largest ΔBIC value;
divide W into two sub-windows, W_1 and W_2, at $\hat{t}$;
3) //Solve sub-instances
$CP_{W_1} \leftarrow DACDec2(W_1)$; $CP_{W_2} \leftarrow DACDec2(W_2)$;
4) //Combine
if ($\Delta BIC_{\{W_1,W_2\}}(\hat{t})$ calculated in 2) is positive)
$CP \leftarrow \hat{t} \cup CP_{W_1} \cup CP_{W_2}$;
else
let $\mathcal{X}$ be the segment on the left of $\hat{t}$ in W_1 and $\mathcal{Y}$ be the segment on the right of $\hat{t}$ in W_2;
if ($\Delta BIC_{\{\mathcal{X},\mathcal{Y}\}}(\hat{t}) > 0$) //$\hat{t}$ is a change point
$CP \leftarrow \hat{t} \cup CP_{W_1} \cup CP_{W_2}$;
else //$\hat{t}$ is not a change point
merge $\mathcal{X}$ and $\mathcal{Y}$;
$CP \leftarrow CP_{W_1} \cup CP_{W_2}$;
End

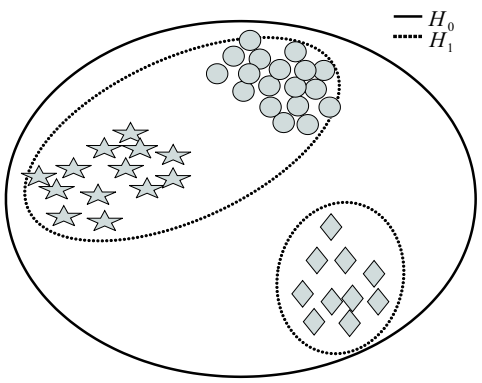


Fig. 3. An illustration that data samples distribute as three Gaussian clusters. For this case, generally, two Gaussians (H_1) fit the distribution of the data better than one Gaussian (H_0) if the samples belonging to the same Gaussian cluster are used together to estimate the parameters.

assume that there are no miss and false alarm errors in the detection process. In the figure, each tree node corresponds to a *divide-point* (i.e., $\hat{t}$) in the analysis window; the number inside the node indicates the order of the division, while the number below the node indicates the order in which the divide-point is verified in the *Combine* stage. In Fig. 4 (b), Node 1 (C_2) has a negative ΔBIC value in the *Divide* stage; however, it will be labeled as a change point by the verification process with segments {c, d, e, f} and {g, h, i} in the *Combine* stage. Node 2 (C_1) has a positive ΔBIC value in the *Divide* stage; thus, it is labeled as a change point and verification is not necessary. Segments {a} and {b} will be used for verifying Node 3; segments {c, d} and {e, f} will be used for verifying Node 4, and so on.

1) Advantages of DACDec1 and DACDec2: We use Fig. 6 to explain the potential advantages of using DACDec1 and DACDec2 for speaker change detection. In the figure, there are two change points, C_1 and C_2. For WinGrow, if there is a false alarm error at F near C_1, the detection process restarts at F, but the false alarm error may lead to miss errors in the subsequent detection process. For example, if the three segments are derived from different speakers, like the case in Fig. 2 (a), it is very likely that C_2 will be detected and C_1 will be missed because the analysis window does not contain sufficient data from Seg1. On the other hand, if Seg1 and Seg3 are spoken by the same speaker and Seg2 is from another source, C_1 may be missed for the same reason mentioned above. If C_1 is missed, we may suffer from the unreliable ΔBIC measurement issue as the example in Fig. 4 when OCD-Chen continues to detect C_2; thus, C_2 may also be missed.

In DACDec1, the *Divide* stage partitions the audio stream into two sub-streams at the point with the largest positive

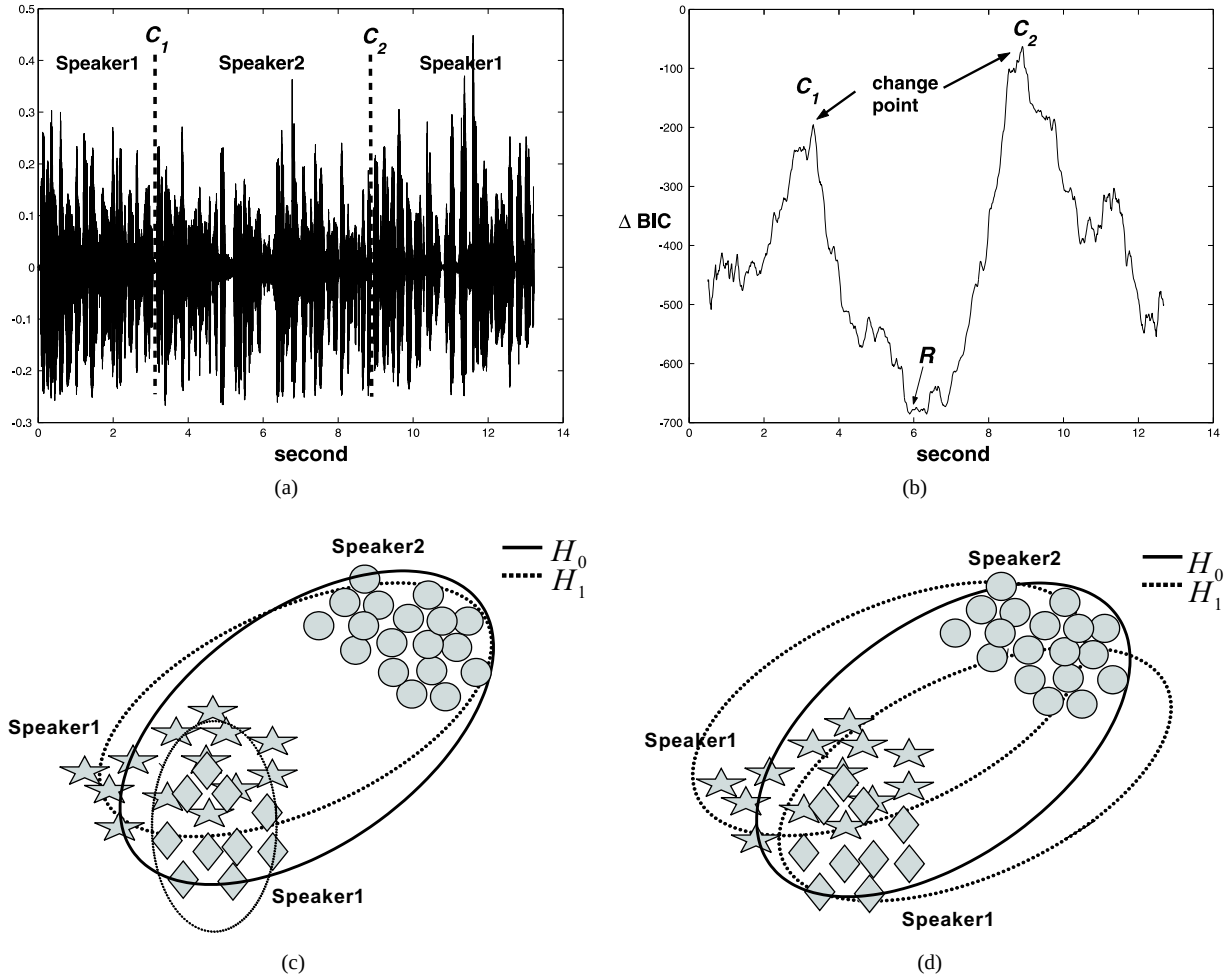


Fig. 4. (a) An audio stream comprised of three speech segments; the first and third segments are derived from the same speaker (Speaker1), while the second is derived from another speaker (Speaker2). (b) The ΔBIC curve obtained by applying OCD-Chen to the audio stream in (a). (c) The diagram of the hypothesis test at the change point C_2 in (b). (d) The diagram of the hypothesis test at the non-change point R in (b).

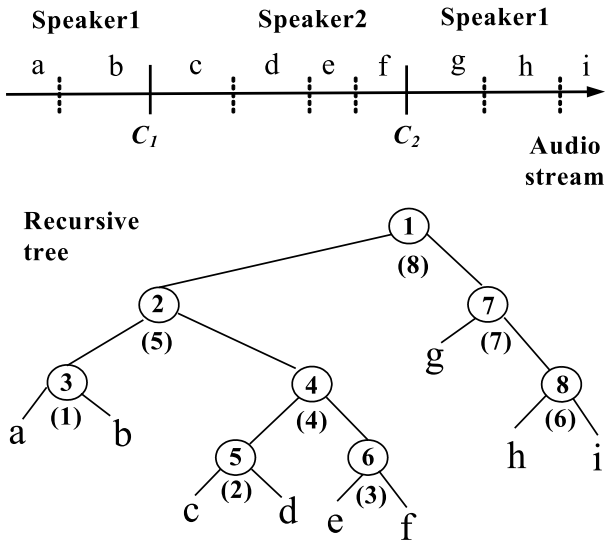


Fig. 5. A recursive tree that simulates the recursive process of DACDec2 on the audio stream in Fig. 4 (a).

ΔBIC value. As shown in Figs. 4 (c) and (d), change points usually have larger ΔBIC values than non-change points; thus, false alarm errors may only occur after the true change points have been detected, and they will not lead to miss errors in the subsequent detection process. For example, in Fig. 6, the false alarm errors in Seg1 only occur after C_1 has been detected, and so on.

In DACDec2, false alarm errors may not cause a true change point with a positive ΔBIC value calculated in the *Divide* stage to be missed; however, the false alarm points' neighboring true change points with negative ΔBIC values calculated in the *Divide* stage may be missed. For instance, for the example in Fig. 4, DACDec2 first divides the audio stream at C_2 and generates the recursive tree shown in Fig. 5. C_2 needs to be verified in the *Combine* stage because its ΔBIC value calculated in the *Divide* stage is negative, as shown in Fig. 4 (b). However, if there is a false alarm point near C_2, it may be missed. We explain this phenomenon as follows. In Fig. 5, the boundary between segments {e} and {f} (i.e., Node 6) is verified before C_2. If it is detected as a change point, {f} and {g, h, i} are used to verify C_2. However, the number of data samples in segment {f} may be insufficient to

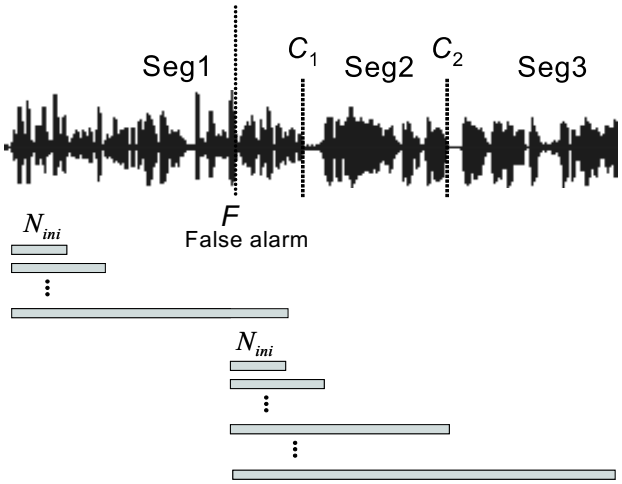


Fig. 6. An example of the WinGrow detection process. The audio stream contains two change points C_1 and C_2, and the detection process generates a false alarm error at F.

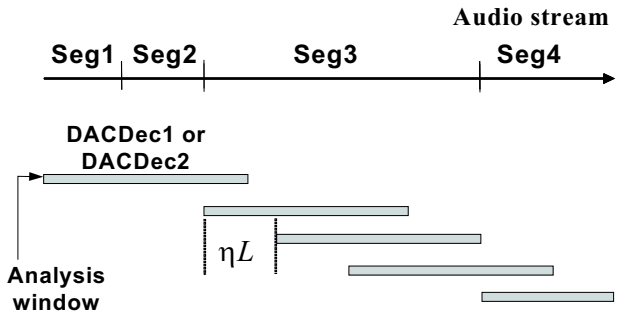


Fig. 7. Diagram of the detection process of SeqDACDec1 and SeqDACDec2. If a change point is detected in the fixed-size analysis window by DACDec1 or DACDec2, the window is moved to the change point with the largest time index. Otherwise, it is moved forward by ηL samples, where L denotes the window size, and $\eta > 0$.

obtain a reliable ΔBIC measurement, thus C_2 may be missed. On the other hand, if DACDec2 first divides the audio stream at C_1 and outputs a false alarm point at F near C_1, like the case in Fig. 6, C_1 may be missed for the same reason. Even so, missing C_1 may not cause DACDec2 to miss C_2 because, in the *Divide* stage, C_2 will be determined whether it is a change point using complete, pure Seg2 and Seg3; therefore, it is very likely that the ΔBIC value will be positive. In contrast, WinGrow may not be able to use complete, pure Seg2 and Seg3 for speaker change detection if C_1 is missed.

2) Sequential segmentation by DACDec1 and DACDec2: For a long audio stream, such as a one-hour broadcast news program, the segmentation task becomes computationally intractable when DACDec1 or DACDec2 are used to detect change points. Moreover, if the initial analysis window contains too many segments, it may be difficult for OCD-Chen to find an appropriate λ value to obtain robust ΔBIC measurements for the various hypothesis tests in the recursive process. Therefore, in practical applications, we apply DACDec1 and DACDec2 in a large analysis window of fixed-size (e.g., 20 seconds) that moves from the beginning to the end of the audio stream to detect the speaker changes sequentially. The proposed sequential segmentation algorithms, SeqDACDec1 and SeqDACDec2, are shown in Fig. 7. In SeqDACDec1 (or SeqDACDec2), if a change point is detected in the fixed-size analysis window by DACDec1 (or DACDec2), the window is moved to the change point with the largest time index. Otherwise, it is moved forward by ηL samples, where L denotes the window size, and $\eta > 0$. Note that a small η will allow a missed change point to be checked again by DACDec1 (or DACDec2) in the subsequent fixed-size analysis window. Like WinGrow, SeqDACDec1 and SeqDACDec2 are suitable for on-line applications.

C. The DACDec3 approach

The third implementation (DACDec3) of the divide-and-conquer paradigm is detailed in Algorithm 3. In DACDec3, we use FixSlid instead of OCD-Chen to detect the divide-points of an input audio stream. As shown in Fig. 8 (a), given the distance curve obtained by FixSlid using the GLR [9] distance measure, the time index t associated with the peak that has the largest GLR value within the interval [$t-pRange$, $t+pRange$] is considered a divide-point. In this example, all the peaks except S are divide-points. Let DP_{set} be the set of divide-points obtained by FixSlid. As described in Algorithm 3, DACDec3 returns if a divide-point is not found in DP_{set}. In the *Divide* stage, the analysis window is partitioned at the time index of the divide-point with the largest GLR value. Then, in the *Combine* stage, each divide-point is evaluated to determine whether it is a change point based on the ΔBIC measurement of its two neighboring segments $\mathcal{X}$ and $\mathcal{Y}$.

The major difference between DACDec2 and DACDec3 is as follows. DACDec2 detects change points by OCD-Chen in the *Divide* stage. Then, only the divide-points with negative ΔBIC values calculated in the *Divide* stage are verified by segment merging based on the ΔBIC values of their neighboring segments in the *Combine* stage. In contrast, DACDec3 detects change points by verifying all the input divide-points indicated by FixSlid using segment merging. For example, if we apply FixSlid to the audio stream in Fig. 5 and obtain the distance curve in Fig. 8 (a), the recursive tree for DACDec3 will be the same as that in Fig. 5. In this case, DACDec2 finds the change point C_1 using OCD-Chen in the *Divide* stage, while DACDec3 finds it in the *Combine* stage using segment merging; however, both DACDec2 and DACDec3 find the change point C_2 in the *Combine* stage.

There is a close link between DISTBIC [9] and DACDec3. In DISTBIC, a distance curve is first generated by FixSlid, and then the "significant" local maximums on the distance curve are evaluated to determine whether they are change points by a sequential, left-to-right (in time order) segment merging process. As shown in Fig. 9, a local maximum is significant if $|d(\max) - d(\min_r)| > \alpha\sigma$ and $|d(\max) - d(\min_l)| > \alpha\sigma$, where σ is the standard deviation of the distance values on the distance curve, α is a positive real number, and $\min_r$ and $\min_l$ are, respectively, the locations associated with the right minimum and left minimum around the local maximum.

Algorithm 3 $CP \leftarrow$DACDec3(W, DP_{set}, GLR_{set})

Require: W: the analysis window
$DP_{set} = \{DP_1, \ldots, DP_N\}$: the divide-points in W obtained by FixSlid using the GLR distance measure
$GLR_{set} = \{GLR_1, \ldots, GLR_N\}$: GLR_i denotes the GLR value at DP_i for $i = 1, 2, \cdots, N$
Ensure: CP: the set of change points detected in W
Begin
1) //Check termination
if (DP_{set} is empty)
$CP \leftarrow \phi$; //empty set
goto End; //return
2) //Divide
search in DP_{set} and let DP_k be the divide-point whose GLR value is the largest in GLR_{set};
let $\hat{t}$ be the time index of DP_k; divide W into two sub-windows, W_1 and W_2, at $\hat{t}$;
divide DP_{set} into two sub-sets, $DP_{set1} = \{DP_1, \ldots, DP_{k-1}\}$ and $DP_{set2} = \{DP_{k+1}, \ldots, DP_N\}$;
divide GLR_{set} into two sub-sets, $GLR_{set1} = \{GLR_1, \ldots, GLR_{k-1}\}$ and $GLR_{set2} = \{GLR_{k+1}, \ldots, GLR_N\}$;
3) //Solve sub-instances
$CP_{W_1} \leftarrow DACDec3(W_1, DP_{set1}, GLR_{set1})$; $CP_{W_2} \leftarrow DACDec3(W_2, DP_{set2}, GLR_{set2})$;
4) //Combine
let $\mathcal{X}$ be the segment on the left of $\hat{t}$ in W_1 and $\mathcal{Y}$ be the segment on the right of $\hat{t}$ in W_2;
if ($\Delta BIC_{\{\mathcal{X},\mathcal{Y}\}}(\hat{t}) > 0$) //$\hat{t}$ is a change point
$CP \leftarrow \hat{t} \cup CP_{W_1} \cup CP_{W_2}$;
else //$\hat{t}$ is not a change point
merge $\mathcal{X}$ and $\mathcal{Y}$;
$CP \leftarrow CP_{W_1} \cup CP_{W_2}$;
End

If DISTBIC takes the divide-points of DACDec3 as change point candidates to be verified (we denote this approach as DISTBIC_pR), it is identical to applying DACDec3 in that the recursive division is performed in a right-to-left manner, whereas the recursive segment merging is performed in a left-to-right manner. As shown in Fig. 8 (b), Node 8 is verified by segments {a} and {b} first, then Node 7 is verified by segments {a, b} and {c}, and so on.

DACDec3 should be more effective than DISTBIC_pR because it evaluates the divide-points with smaller GLR values to determine whether they are change points before those with larger GLR values. In contrast, in DISTBIC_pR, the divide-points are simply verified sequentially without considering the GLR information. The advantage of DACDec3 can be seen by comparing the recursive tree of DACDec3 in Fig. 5 to that of DISTBIC_pR in Fig. 8 (b). In DACDec3, C_1 may be verified with segments {a, b} and {c, d, e, f}, which are complete homogeneous segments of Speaker1 and Speaker2, respectively; whereas, in DISTBIC_pR, C_1 can only be verified with segments {a, b} and {c} or segments {b} and {c}, where only a small portion of Speaker2's data is used.

In addition to the recursive (sequential) segment merging process of DACDec3 (DISTBIC_pR), one can use the hierarchical agglomerative clustering (HAC) to merge the segments obtained with DACDec3's divide-points [7], [8], [30]. We denote this segmentation approach as FixSlidHAC_pR. Compared to DACDec3 (or DISTBIC_pR), which performs segment merging locally, FixSlidHAC_pR performs segment clustering globally. When performing HAC, each segment is considered as a cluster initially; then, in each merging step, the two clusters with the smallest distance are merged into a new cluster. The globality feature of FixSlidHAC_pR is particularly beneficial to the speaker diarization task because the segment merging process groups the segments into clusters such that each cluster contains segments of the same speaker. However, this feature might not be as beneficial to the speaker segmentation task because the goal is to merge adjacent segments into longer segments. For example, in Fig. 5, the goal of the segment merging process in the speaker segmentation task is to merge segments {a} and {b} into one larger segment and to merge segments {g}, {h} and {i} into another larger segment, rather than to merge these five segments into one cluster. In FixSlidHAC_pR, if segment {h} is incorrectly merged with a segment of a different speaker, say {d}, the error will propagate in the following clustering process. DACDec3 might not suffer the same fate because its locality constraint enforces that segment {h} is first checked with its neighboring segment, {g} or {i}. Therefore, we think DACDec3's segment merging process meets the goal of speaker segmentation better than that of FixSlidHAC_pR. Moreover, it is clear that the computational cost of FixSlidHAC_pR is much larger than that of DACDec3 due to the essential computational cost of the HAC-based global clustering process.

Like DACDec1 and DACDec2, DACDec3 can also be applied sequentially in a fixed-size analysis window for on-line applications.

IV. Computational cost analysis

WinGrow, DACDec1, and DACDec2 detect acoustic changes by applying the OCD-Chen process to the analysis window. From Eq. (4), it is clear that the computational cost of ΔBIC is mainly from the cost of calculating covariance matrices, which is proportional to the number of data samples. Let the time cost of calculating ΔBIC with m samples be $m\tau$, where τ represents the time unit; then, $m^2\tau$ denotes the time cost of applying OCD-Chen to an analysis window of m samples[1].

[1]As mentioned in Section II-B1, the ΔBIC value is not computed for samples at the beginning and the end of the analysis window. However, to simplify the analysis, we assume that the ΔBIC value is computed for each sample of the window.

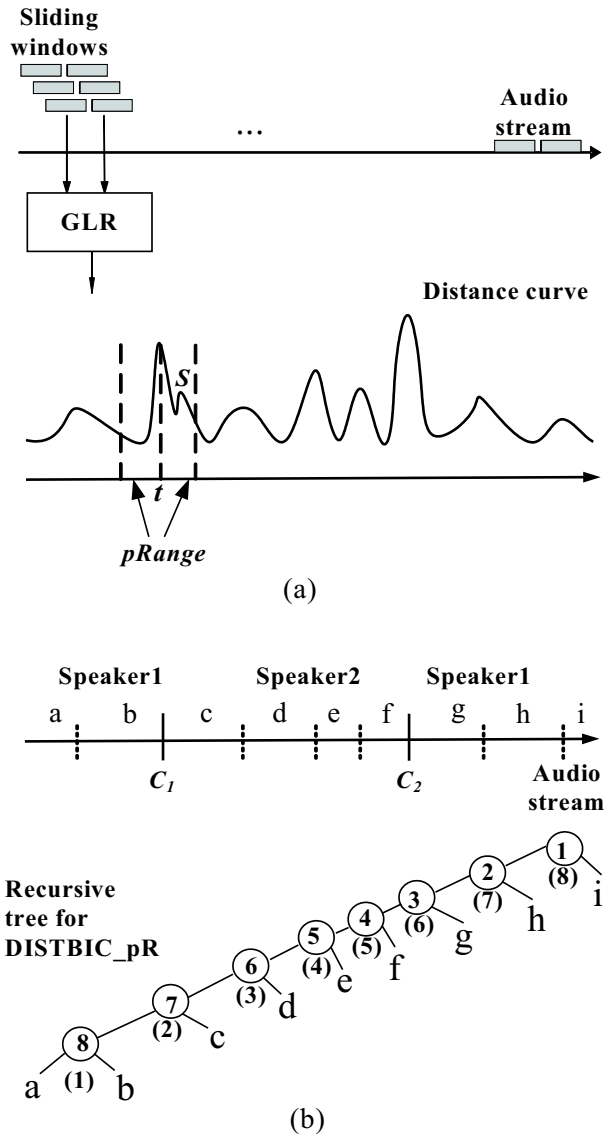


Fig. 8. (a) The distance curve obtained by FixSlid using the GLR distance measure, where the time index t associated with the peak that has the largest GLR value within the interval $[t - pRange,\ t + pRange]$ is considered a divide-point in DACDec3. On this curve, all the peaks, except S, are divide-points. (b) The recursive tree representation of DISTBIC_pR based on the divide-points in (a) for the audio stream in Fig. 5.

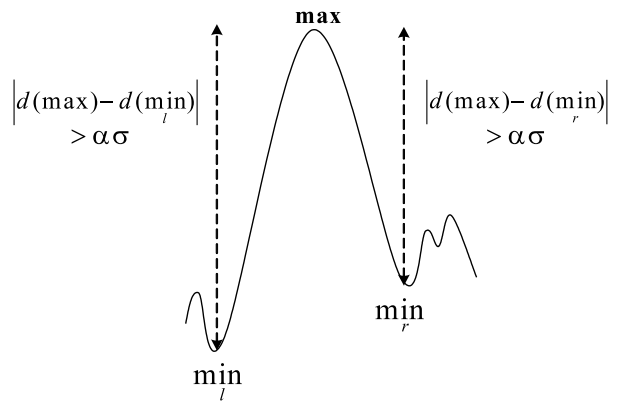


Fig. 9. A significant local maximum on the distance curve.

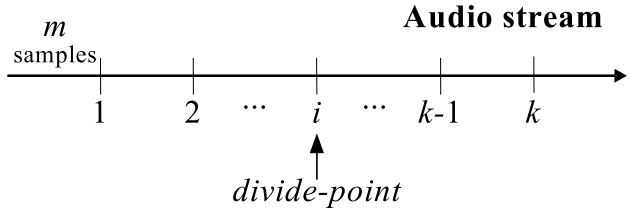


Fig. 10. An audio stream comprised of $k+1$ homogeneous segments, each containing m samples. The stream is divided at the i-th change point.

To simplify the analysis, we assume that each homogeneous segment in the input audio stream (i.e., the initial analysis window for DACDec1, DACDec2, and DACDec3) contains m samples. Moreover, we assume the detection process is perfect, i.e., miss and false alarm errors never occur.

1) For DACDec1: Let $T_1(k)$ denote the time cost of applying DACDec1 to an audio stream of k change points (i.e., $k+1$ homogeneous segments). When the audio stream is divided at the i-th change point, as shown in Fig. 10, we obtain the following recursive expression of $T_1(k)$:

$$T_1(k) = T_1(i-1) + T_1(k-i) + (k+1)^2 m^2 \tau, \quad (5)$$

where $(k+1)^2 m^2 \tau$ is the time cost of finding the divide-point by OCD-Chen; $T_1(i-1)$ and $T_1(k-i)$ are the time costs of applying DACDec1 in the left sub-stream and the right sub-stream, respectively. We have $T_1(0) = m^2\tau$, since it represents the time cost of applying OCD-Chen to a m-sample homogeneous segment.

We assume that the division occurs at each change point with equal probability; therefore, the average time cost of DACDec1 is

$$T_1(k) = \frac{1}{k}\sum_{i=1}^{k}(T_1(i-1) + T_1(k-i)) + (k+1)^2 m^2 \tau. \quad (6)$$

After the algebraic manipulation detailed in Appendix A, we obtain

$$\begin{aligned} T_1(k) &\approx (3(k+1)^2 - 2(k+1)\ln(k+1))m^2\tau \\ &= O(k^2 m^2 \tau). \end{aligned} \quad (7)$$

2) For DACDec2: Compared to DACDec1, DACDec2 incurs an additional time cost in the *Combine* stage as it has to determine whether the divide-point with a negative ΔBIC value calculated in the *Divide* stage is a change point. The cost is $2m\tau$ because each of the divide-point's two neighboring segments contains m samples. To simplify the analysis, we assume that each divide-point must be verified, even though its ΔBIC value calculated in the *Divide* stage is positive. Hence, the average time cost of DACDec2 is

$$T_2(k) = \frac{1}{k}\sum_{i=1}^{k}(T_2(i-1) + T_2(k-i)) + (k+1)^2 m^2 \tau + 2m\tau. \quad (8)$$

Unlike DACDec1, DACDec2 recursively partitions each homogeneous segment of m samples until the analysis window is smaller than the pre-defined minimum value N_{min}. Therefore, $T_2(0)$ is equivalent to the time cost of applying DACDec2 to an m-sample stream in which each sample can be a divide-point. The cost of finding a divide-point in an m-sample stream in the *Divide* stage is $m^2\tau$. In the *Combine* stage, the cost of verifying the divide-point is at most $m\tau$ because the two segments used for verification are sub-segments of the m-sample segment. Therefore, the upper bound of $T_2(0)$ is

$$T'(m) = \frac{1}{m}\sum_{i=1}^{m}(T'(i-1) + T'(m-i)) + m^2\tau + m\tau, \quad (9)$$

where $T'(0) = 0$. After the algebraic manipulation detailed in Appendix B, we obtain

$$T_2(0) \leq T'(m) \approx (3m + 4 - 4\ln(m+1))(m+1)\tau. \quad (10)$$

Then, we can solve the recursive equation in Eq. (8) with $T_2(0)$ in Eq. (10). After the algebraic manipulation detailed in Appendix C, we obtain

$$\begin{aligned} T_2(k) &\leq (3k + 5 - 2\ln(k+1))(k+1)m^2\tau \\ &\quad +(9 + 2\ln k - 2\ln(k+1) - 4\ln(m+1))(k+1)m\tau \\ &\quad +(4 - 4\ln(m+1))(k+1)\tau \\ &= O(k^2m^2\tau). \end{aligned} \quad (11)$$

3) For FixSlid, DACDec3, and DISTBIC_pR: Suppose FixSlid uses GLR (ΔBIC) as the distance measure and the analysis window consists of ω samples. Then, the time cost of FixSlid is

$$\begin{aligned} T_3(k) &= (2\omega\tau)(k+1)m \\ &= O(km\tau). \end{aligned} \quad (12)$$

DACDec3 and DISTBIC_pR incur a higher time cost than FixSlid when verifying divide-points in the segment merging process. Suppose the audio stream is equally divided into $\frac{(k+1)m}{\beta}$ sub-segments of β samples, where $\beta > 0$. Since we assume that the segmentation derived by DACDec3 and DISTBIC_pR is perfect, each of the change points is also a divide-point and the time cost of segment merging verification is less than $2m\tau$ for each divide-point. Therefore, the time cost of DACDec3 and DISTBIC_pR is less than

$$\begin{aligned} T_4(k) &= (2\omega\tau)(k+1)m + (\frac{(k+1)m}{\beta} - 1)2m\tau \\ &= O(km^2\tau). \end{aligned} \quad (13)$$

4) For WinGrow: We analyze the case where the maximum window size N_{max} is large enough to ensure that the search process always restarts at a true change point[2]. In this case, the analysis window W initialized with a small number of N_{ini} samples grows repeatedly by N_g samples until it contains more than m samples, so that there is at least one change point in W. Suppose W needs to grow to γm samples to detect the change point, where $\gamma > 0$; then, the time cost of sequentially detecting k change points will be

$$T'_1(k) = k[N_{ini}^2 + \sum_{i=1}^{(\gamma m - N_{ini})/N_g}(N_{ini} + iN_g)^2]\tau. \quad (14)$$

After the k-th change point has been detected, the detection process continues to search in the last homogeneous segment; the time cost is

$$C_s = [N_{ini}^2 + \sum_{i=1}^{(m - N_{ini})/N_g}(N_{ini} + iN_g)^2]\tau. \quad (15)$$

In practical applications, both N_{ini} and N_g are set at small values. To simplify the analysis, we assume $N_{ini} \approx N_g$. Then, the time cost of WinGrow is

$$\begin{aligned} T_5(k) &= T'_1(k) + C_s \\ &\approx k[\sum_{i=1}^{(\gamma m - N_g)/N_g}(iN_g)^2]\tau + [\sum_{i=1}^{(m - N_g)/N_g}(iN_g)^2]\tau \\ &= (\frac{\gamma^3 m^3}{3N_g} - \frac{\gamma^2 m^2}{2} + \frac{\gamma m N_g}{6})k\tau \\ &\quad +(\frac{m^3}{3N_g} - \frac{m^2}{2} + \frac{mN_g}{6})\tau \\ &= O(km^3\tau). \end{aligned} \quad (16)$$

5) Discussion: From Eqs. (7), (11), (12), (13), and (16), it is obvious that FixSlid, DACDec3, and DISTBIC_pR are more efficient than DACDec1, DACDec2, and WinGrow.

DACDec1 and DACDec2 are more efficient than WinGrow when the input audio stream is composed of long homogeneous segments. For example, if the frame rate is 100 frames per second (i.e., there are 100 feature vectors for a one-second audio stream), it is appropriate to set the value of N_{ini} and N_g at 100. Moreover, the value of γ can be set at 1.5 generally. Then, for a 30-second audio stream (which consists of 3000 feature vectors) containing only one change point (i.e., $k = 1$ and $m = 1500$), the speedups of DACDec1 and DACDec2 over WinGrow are 2.55 and 1.78, respectively. When there is no change point in the 30-second stream, the speedups of DACDec1 and DACDec2 over Wingrow are 10.51 and 3.51, respectively. In contrast, when the audio stream is composed of short homogeneous segments, WinGrow is more efficient than DACDec1 and DACDec2. For example, for a 30-second stream containing five change points (i.e., $k = 5$ and $m = 500$), the speedups of DACDec1 and DACDec2 over WinGrow are 0.42 and 0.37, respectively.

V. Experiments on speaker segmentation

We conducted experiments on a synthetic data set using SeqDACDec1 and SeqDACDec2 to verify the unreliable

[2]Without this assumption, the time cost analysis for WinGrow might be intractable. However, this assumption is appropriate for many kinds of real-world data. For example, in our experiments on the broadcast news data described in Sec. V, it is appropriate to set N_{max} at 20 seconds, which is longer than most of the homogeneous segments in the data set.

ΔBIC measurement issue in DACDec1, and on two real-world broadcast news data sets to evaluate the performances of the baseline and proposed segmentation approaches.

For feature extraction, we used a 32-ms Hamming window shifted with a step of 10-ms to extract 24 mel-frequency cepstral coefficients as the acoustic features [11]. There were 100 24-dimensional feature vectors in a one-second audio stream.

For the performance evaluation, we used the Receiver Operating Characteristic (ROC) curve to show the various miss detection (MD) rates and false alarm (FA) rates yielded by adjusting the threshold parameters. A true change point t was counted as a miss detection if there was no hypothesized change point within $[t-\xi, t+\xi]$ (a 2ξ-second window centered on t); and a hypothesized change point $\hat{t}$ was counted as a false alarm if there was no true change point within $[\hat{t}-\xi, \hat{t}+\xi]$. The miss detection rate (MDR) and false alarm rate (FAR) are defined as

$$\text{MDR}=100\% \times \frac{\text{number of MD}}{\text{number of true change points}},$$

$$\text{FAR}=100\% \times \frac{\text{number of FA}}{\text{number of hypothesized change points}}.$$

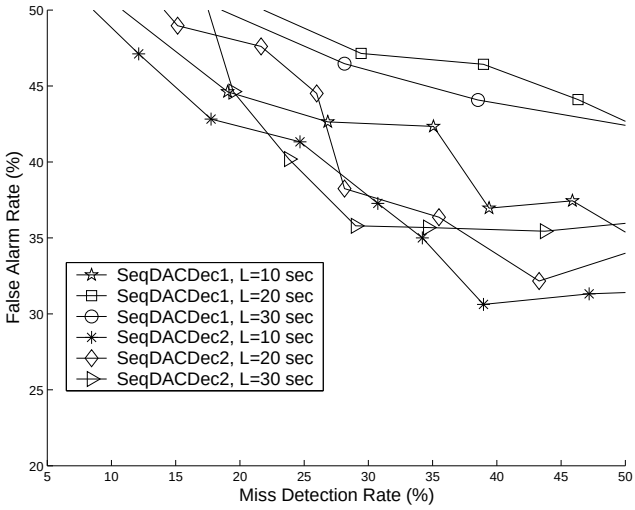


Fig. 11. ROC curves obtained by running SeqDACDec1 and SeqDACDec2 on the synthetic data using 10-second, 20-second, and 30-second analysis windows. L denotes the size of the analysis window.

A. Experiments on the synthetic data

1) Data set description: We used the training data of six speakers from the 2001 NIST speaker recognition evaluation corpus [34] to create three artificial audio streams of conversational speech as the synthetic data set. The speech from speaker#5077 and speaker#5232 was divided into three-second utterances and interlaced to form an audio stream of conversational speech of two speakers. In the same way, the speech from speaker#5326 and speaker#5333 was used to form the second audio stream; and the speech from speakers#5446 and speaker#5269 was used to form the third audio stream. There were 231 speaker change points in total in the three audio streams.

2) Experiment results: Fig. 11 shows the ROC curves obtained by running SeqDACDec1 and SeqDACDec2 on the synthetic data with different analysis window sizes. η was set at 0.25, N_{min} in DACDec1 and DACDec2 was set at one second (i.e., 100 samples), and the penalty factor λ in ΔBIC was set at 0.7 initially and increased to 1.7 in 0.05 increments. The ΔBIC distance was evaluated every 0.1 seconds in both approaches; that is, the resolution for change point detection was 0.1 seconds. The tolerance ξ for counting the number of miss detection or false alarm was set at 0.5 seconds. From the figure, we observe that SeqDACDec2 outperforms SeqDACDec1 for every window size. Moreover, SeqDACDec2 yields similar performances at different window sizes, whereas the performance of SeqDACDec1 declines significantly when the window size is increased from 10 seconds to 20 or 30 seconds. In other words, SeqDACDec2 is more robust to the size of the analysis window than SeqDACDec1. The experiment results conform to the discussion in Sec. III; that is, DACDec1 might not work as well as DACDec2 if the condition that the homogeneous segments in the analysis window are derived from different acoustic sources is not met.

B. Experiments on broadcast news data

1) Data set description: We evaluated FixSlid, FixSlidHAC_pR, DISTBIC_pR, DISTBIC, WinGrow, and the proposed methods on two broadcast news data sets. The broadcast news data in the 2003 NIST rich transcription evaluation data [32], which is comprised of six 30-minute audio streams recorded from channels ABC, NBC, CNN, PRI, VOA, and MNB, was used as the evaluation set (denoted as RT03). Three one-hour broadcast news programs (PTSND-20011203, PTSND-20011204, and PTSND-20011205) selected from the MATBN corpus [31] were used as the development set (denoted as MATBN3hr). To be consistent with RT03, each file in MATBN3hr was divided into two 30-minute audio streams in the experiments. According to the manual transcriptions, there were 1261 and 444 speaker change points in MATBN3hr and RT03, respectively. Note that, in the evaluation, we ignored the hypothesized change points that locate in the non-speech regions labeled in the transcription when evaluating the segmentation errors because the detection of acoustic changes within the non-speech regions was outside the scope of this study.

Fig. 12 shows the empirical cumulative distributions of the size of homogeneous segments in the two data sets. As shown in the figure, the average length of the segments in RT03 is longer than that in MATBN3hr.

2) Parameter setting and system description: For FixSlid, we used the GLR distance as the distance measure of two adjacent windows. In the experiments, the window size was fixed at two seconds; and the value of α used to evaluate the "significant" local maximum, as shown in Fig. 9, was set at 0.4 initially, and increased to 2 in 0.05 increments to obtain the ROC curve. For DACDec3, DISTBIC_pR, and FixSlidHAC_pR, the parameter $pRange$ was tuned with the

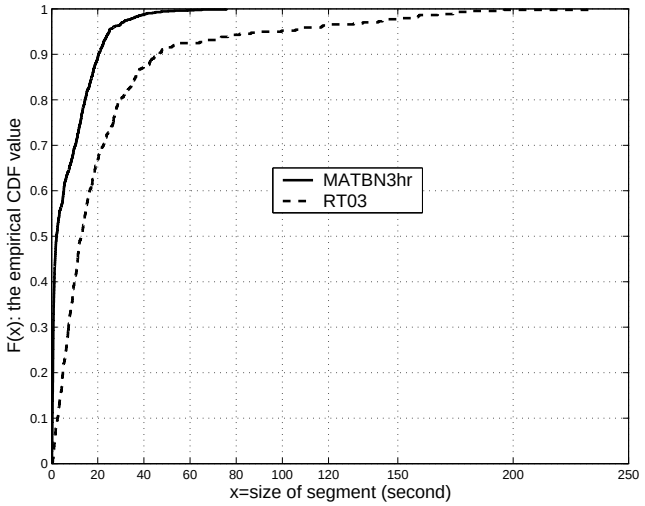


Fig. 12. The empirical cumulative distributions of the size of homogeneous segments in MATBN3hr and RT03.

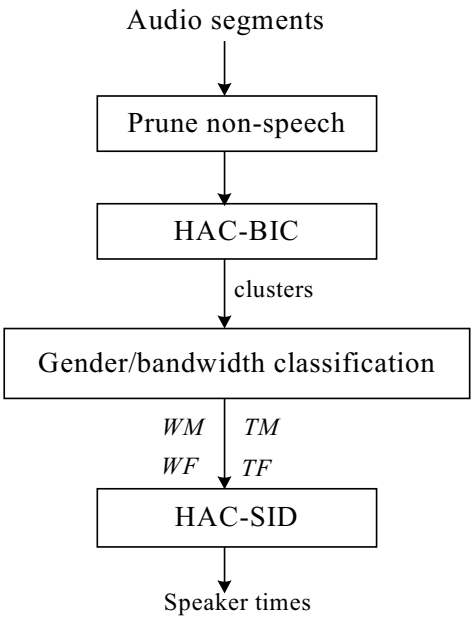


Fig. 13. A multi-stage HAC that consists of BIC clustering (HAC-BIC), gender/bandwidth classification and SID clustering (HAC-SID).

development set. For WinGrow, the values of N_g and N_s were set at one second and N_{max}/4 seconds, respectively; and the values of N_{ini} and N_{max} were tuned with the development set. For SeqDACDec1 and SeqDACDec2, η was fixed at 0.25; and L and N_{min} in DACDec1 and DACDec2 were tuned with the development set. For each BIC-based segmentation approach, various λ values for ΔBIC were applied in order to obtain the ROC curve. Like the above experiments on the synthetic data, the resolution for change point detection was 0.1 seconds for all the approaches. However, the tolerance ξ for counting the number of miss detection or false alarm was set at one second rather than 0.5 seconds. Basically, we made this change because of the limited precision of human reference annotation.

For FixSlidHAC_pR, we first applied FixSlid with the threshold parameter $pRange$ to segment the input audio stream, then we pruned non-speech regions within the audio segments and grouped the segments using HAC with multiple stages, which have been applied in state-of-the-art speaker diarization systems [8], [7], [35], [30]. As shown in Fig. 13, we applied HAC with ΔBIC as the inter-cluster distance measure (HAC-BIC) for initial clustering; the clustering process was stopped if the smallest ΔBIC value among all the cluster pairs was larger than zero. Then, we classified the resultant clusters into four classes, namely, male speech with studio/wide-bandwidth condition (*WM*), male speech with telephone/narrow-bandwidth condition (*TM*), female speech with studio condition (*WF*), and female speech with telephone condition (*TF*). After the gender/bandwidth classification, we applied HAC with the cross log-likelihood ratio derived from GMMs as the inter-cluster distance measure (HAC-SID) to the four classes, individually [7]. The cross log-likelihood ratio is defined as

$$CLR_{GMM}(\pi_i, \pi_j) = \frac{1}{n_i} \log \frac{p(\pi_i|M_j)}{p(\pi_i|B)} + \frac{1}{n_j} \log \frac{p(\pi_j|M_i)}{p(\pi_j|B)}, \quad (17)$$

where M_i and M_j are, respectively, the GMMs for clusters π_i and π_j, which are MAP-adapted from the universal background model (UBM [36]) B. Here, only Gaussian mean vectors were adapted, and the relevant factor for controlling the adaptation rate was experimentally set at 16. CLR_{GMM} reveals the similarity between π_i and π_j. Therefore, when applying this measure in HAC, the two clusters with the largest CLR_{GMM} value are merged; and the clustering process is terminated when it is smaller than a pre-defined stopping threshold. We used 15 MFCCs and energy plus their delta coefficients, which were normalized by feature warping, as the speech feature for HAC-SID [7]. We used the 1998 DARPA/NIST HUB-4 broadcast news evaluation test data to train the UBMs for *WM* and *WF*, and the NIST 2000 speaker recognition evaluation corpus for *TM* and *TF*; each of the UBMs contained 128 mixture Gaussians.

3) Experiment results: We first evaluated all the segmentation approaches on MATBN3hr. Fig. 14 (a) shows the ROC curves obtained by DACDec3 and DISTBIC_pR with different $pRange$ values. From the figure, we observe that DACDec3 outperforms DISTBIC_pR; and the best setting of $pRange$ for DACDec3 and DISTBIC_pR are 0.5 seconds and one second, respectively. Table I shows the results of FixSlidHAC_pR, where for each $pRange$ case, various settings for λ and the stopping threshold in HAC-SID were evaluated to obtain the lowest equal error rate (EER). From the table, we observe that FixSlidHAC_pR achieves the lowest EER with $pRange = 1.5$ seconds. Both DACDec3 and DISTBIC_pR achieve a lower EER compared to FixSlidHAC_pR; this shows that DACDec3's recursive and DISTBIC_pR's sequential strategies for segment merging outperform the hierarchical agglomerative approach.

We also evaluated DACDec3 using the "significant" local maximums obtained by FixSlid as the divide-points (denoted

TABLE I
THE EERS OF FIXSLIDHAC_PR, DACDEC3, AND DISTBIC_PR ON MATBN3HR, WHERE M AND F DENOTE THE MISS DETECTION RATE AND THE FALSE ALARM RATE, RESPECTIVELY.

Approach	$pRange$ (in second)	EER (in %)
FixSlidHAC_pR	1	M:26.05, F:24.60
	1.5	M:21.09, F:22.22
	2	M:24.35, F:25.10
DACDec3	0.5	M:17.61, F:17.46
DISTBIC_pR	1	M:19.19, F:18.44

as DACDec3_SP). We ran DACDec3_SP and DISTBIC with $\alpha = 0.4$ and $\alpha = 0.85$. From Fig. 14 (b), it is clear that DACDec3_SP and DISTBIC substantially outperform FixSlid, while DACDec3_SP outperforms DISTBIC. Moreover, DACDec3 with $pRange = 0.5$ seconds (the line marked with diamonds) slightly outperforms DACDec3_SP. In our experience, $pRange$ is easier to tune than α. Therefore, we did not analyze DACDec3_SP and DISTBIC further in the remaining experiments for speaker change detection.

When conducting the experiments, we found that it was appropriate to set N_{ini} at three seconds and N_{max} at 20 seconds for WinGrow. For both SeqDACDec1 and SeqDACDec2, it was appropriate to set N_{min} at two seconds and the window size L at 20 seconds. Fig. 15 (a) shows the ROC curves obtained by SeqDACDec1 with analysis windows of different size. Unlike the results for the synthetic data in Fig. 11, the results with 10-second and 20-second analysis windows are similar. This is because, in the broadcast news data, if a 10-second or 20-second analysis window contains multiple homogeneous segments, the segments are usually derived from different speakers. For SeqDACDec2, the results for 10-second, 20-second, and 30-second analysis windows are similar, as shown in Fig. 15 (b). The ROC curves obtained by the different approaches are shown in Fig. 15 (c). We observe that the proposed approaches, namely, SeqDACDec1, SeqDACDec2, and DACDec3, outperform the other approaches, while SeqDACDec2 performs the best. Table II shows the speeds of all the approaches in terms of "times real-time" (real-time factor, xRT[3]) in the EER case. All the programs were implemented with MATLAB, except that the MAP training of GMMs and calculation of mixture likelihood in FixSlidHAC_pR was implemented with C language via MATLAB's API. The programs were run on a machine with a 3.2GHz Intel Xeon CPU. From the table, we observe that SeqDACDec1, SeqDACDec2, and DACDec3 are more efficient than WinGrow. DACDec3 in particular runs much faster than WinGrow. Moreover, FixSlidHAC_pR is much slower than the other approaches.

Next, we conducted experiments on RT03 with the parameters tuned with MATBN3hr. Fig. 16 shows the ROC curves for all approaches. Again, we observe that the proposed approaches, namely, SeqDACDec1, SeqDACDec2, and DACDec3, outperform the other approaches. Table III shows the real-time factor of all the approaches in the EER case. Comparing Table III to Table II, it is clear that every approach achieves a higher speedup over WinGrow on RT03 than on MATBN3hr. This is because the homogeneous segments in RT03 are longer than those in MATBN3hr on average, as shown in Fig. 12, and these approaches achieve higher speedup over WinGrow for an audio stream comprised of longer homogeneous segments, as mentioned in Sec. IV (*cf.* Eqs. (7), (11), (12), (13), and (16)).

[3] xRT=T_s/T_d, where T_s is the system run-time and T_d denotes the time duration of the test data set.

VI. APPLICATION TO SPEAKER DIARIZATION

Speaker diarization, also known as the "who spoke when" task, aims to group together speech segments produced by the same speaker within an audio stream [8]. It has been studied in various data domains, e.g., conversational telephone speech [16], broadcast news data [7], [35], and meeting data [37].

Speaker diarization systems usually consist of two core components, namely speaker segmentation, which chops the audio stream into homogeneous segments, and speaker clustering, which groups the homogeneous segments into speaker clusters. Currently, leading speaker diarization systems usually apply hierarchical agglomerative clustering (HAC) to perform speaker clustering after segmentation [7], [35], [30]. Here, we would like to evaluate the performance of the segmentation approaches discussed above in terms of speaker diarization error by integrating them with the multi-stage HAC in Fig. 13. The diarization system that combines SeqDACDec1 and the multi-stage HAC is denoted as SeqDACDec1_HAC. Similarly, the diarization systems based on the segmentation methods SeqDACDec2, DACDec3, WinGrow, DISTBIC_pR, and FixSlid are denoted as SeqDACDec2_HAC, DACDec3_HAC, WinGrow_HAC, DISTBIC_pR_HAC, and FixSlid_HAC, respectively.

In the implementation, following the speech activity detection (SAD) method in [7], the GMMs for speech, noisy speech, speech over music, pure music, and silence/noise were trained beforehand, and the non-speech regions in the audio segments were pruned by using Viterbi decoding.

A. Experiments on speaker diarization

1) Data set description and performance evaluation: We used RT03 described in Section V-B1 in the speaker diarization experiments. The audio recordings from channels ABC, NBC, and CNN were used as the development set (RT03_Dev); while the recordings from PRI, VOA, and MNB were used as the evaluation set (RT03_Eval).

For the performance evaluation, we used the diarization evaluation tool (md-eval-v21.pl) released by NIST [38] to evaluate the diarization error rate (DER), which takes into account three kinds of error, namely missed speech (MiS), false alarm speech (FaS), and speaker error (SpE). Readers can refer to [7] for a detailed description of these error types.

2) Parameter setting and system description: We used RT03_Dev to tune the parameters for each system, and then evaluated the diarization performance on RT03_Eval. These parameters include λ in the ΔBIC-based inter-cluster distance measure in HAC-BIC, the stopping threshold in HAC-SID, λ in BIC-based segmentation in SeqDACDec1_HAC, SeqDACDec2_HAC, DACDec3_HAC, DISTBIC_pR_HAC and

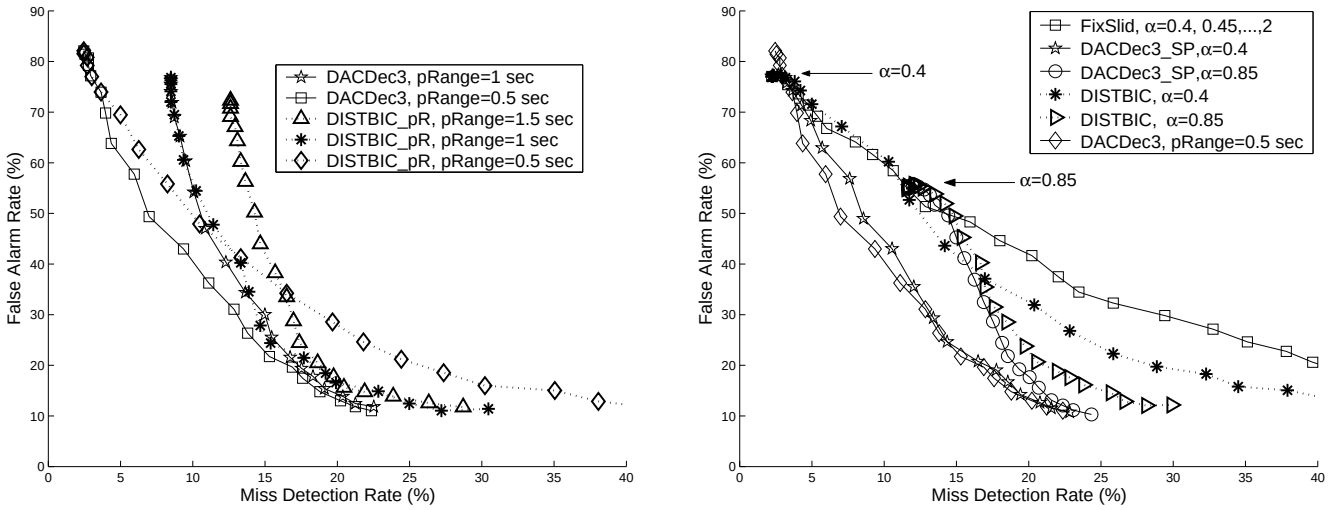


Fig. 14. ROC curves for MATBN3hr obtained by (a) DACDec3 and DISTBIC_pR with different $pRange$ values, and (b) FixSlid, DACDec3_SP, DISTBIC and DACDec3.

TABLE II
THE REAL-TIME FACTOR (xRT) OF DIFFERENT SEGMENTATION APPROACHES EVALUATED ON MATBN3HR IN THE EER CASE AND THE ASSOCIATED EERS, WHERE M AND F DENOTE THE MISS DETECTION RATE AND THE FALSE ALARM RATE, RESPECTIVELY.

Approach	WinGrow	SeqDACDec1	SeqDACDec2	DACDec3	DISTBIC_pR	FixSlid	FixSlidHAC_pR
xRT	0.38	0.1	0.19	0.023	0.026	0.02	1.81
Speedup over WinGrow	1	3.8	2	16.52	14.61	19	0.21
EER (in %)	M:18.08, F:18.65	M:17.84, F:17.21	M:16.42, F:15.92	M:17.61, F:17.46	M:19.19, F:18.44	M:29.42, F:29.82	M:21.09, F:22.22

TABLE III
THE REAL-TIME FACTOR (xRT) OF DIFFERENT SEGMENTATION APPROACHES EVALUATED ON RT03 IN THE EER CASE AND THE ASSOCIATED EERS.

Approach	WinGrow	SeqDACDec1	SeqDACDec2	DACDec3	DISTBIC_pR	FixSlid	FixSlidHAC_pR
xRT	0.53	0.11	0.22	0.022	0.025	0.019	1.87
Speedup over WinGrow	1	4.82	2.41	24.09	21.2	27.89	0.28
EER (in %)	M:17.79, F:16.59	M:17.34, F:18.32	M:16.44, F:15.95	M:18.47, F:17.24	M:22.3, F:21.08	M:34.68, F:33.12	M:23.19, F:24.88

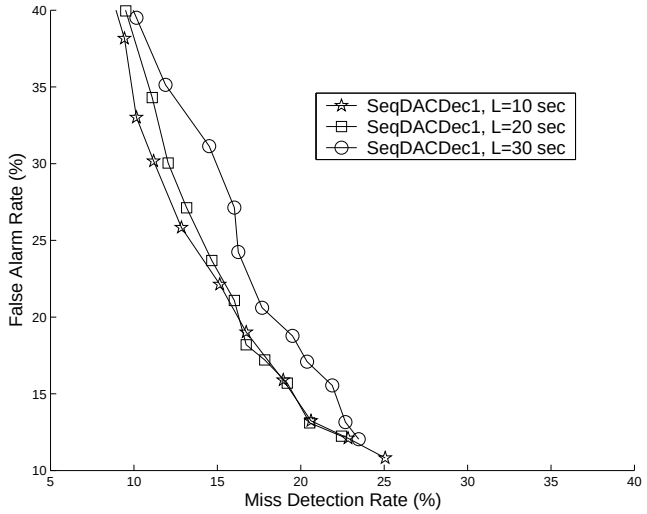
WinGrow_HAC, and α in FixSlid segmentation in FixSlid_HAC. For each system, the remaining parameters in the segmentation stage were the same as those yielding the segmentation results in Figs. 15 (c) (for MATBN3hr) and 16 (for RT03).

3) Post processing by Viterbi re-segmentation: As reported in [35], one can use Viterbi re-segmentation after speaker clustering to improve the diarization accuracy; thus, we used this technique as a post processing step and evaluated how it effects on each diarization system. For the re-segmentation, the speech in each cluster was used to train a MAP-adapted GMM from a gender- and channel-independent UBM first, which represents one state in the applied ergodic HMM. Then, Viterbi decoding was applied to perform the re-segmentation (diarization). The GMM training and re-segmentation were done iteratively.

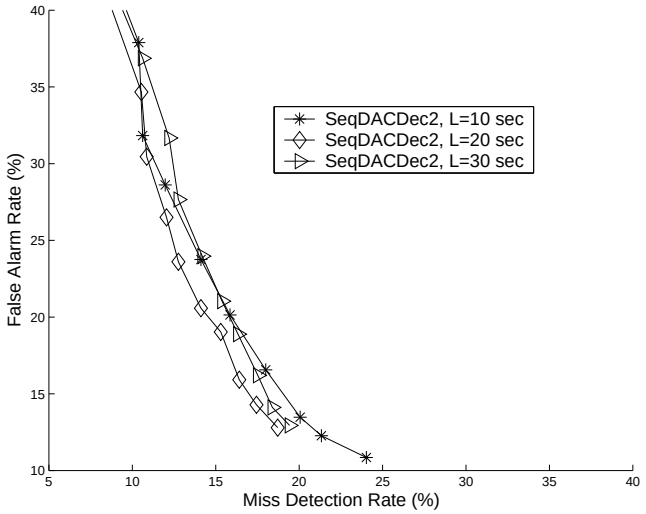
4) Experiment results: Tables IV and V show the DERs of the diarization systems without and with the Vitrebi re-segmentation based post processing step, respectively. From these two tables, two observations can be drawn. First, a more accurate speaker change detection algorithm leads to better diarization accuracy. For example, FixSlid_HAC obtains a higher DER than the other systems. As shown in Fig. 16, its segmentation method, FixSlid, achieves a higher segmentation error. Second, Vitrebi re-segmentation consistently improves the diarization accuracy of all the systems. The improvement is more significant on FixSlid_HAC, which achieves a higher DER originally; however, its DER is still higher than those of the other systems that are based on more accurate speaker segmentation methods.

VII. CONCLUSION

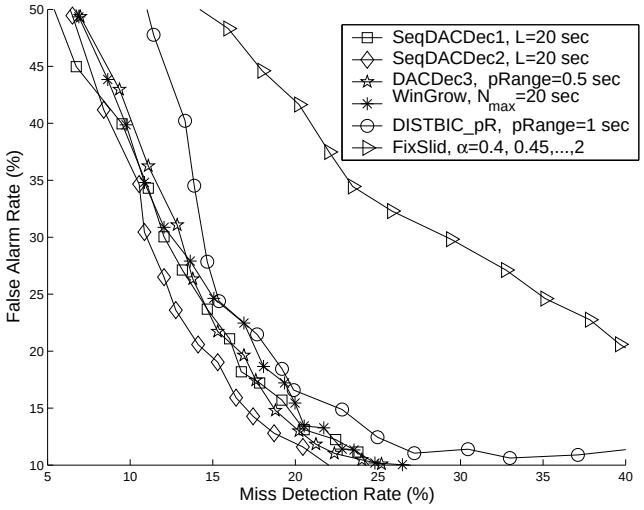
We have proposed three BIC-based speaker segmentation approaches that employ divide-and-conquer strategies for speaker change detection. In contrast to the well-known and highly accurate window-growing-based approach (WinGrow), which searches for change points in a bottom-up manner by using a sequentially growing analysis window, the proposed DACDec1 and DACDec2 approaches search for change points in a top-down manner. The proposed DACDec3 approach is



(a)



(b)



(c)

Fig. 15. The ROC curves for MATBN3hr obtained by (a) SeqDACDec1 with $N_{min} = 2$ seconds and analysis windows of different size (L); (b) SeqDACDec2 with $N_{min} = 2$ seconds and analysis windows of different size (L); and (c) SeqDACDec1 with $N_{min} = 2$ seconds and $L = 20$ seconds, SeqDACDec2 with $N_{min} = 2$ seconds and $L = 20$ seconds, DACDec3 with $pRange = 0.5$ seconds, WinGrow with $N_{min} = 3$ seconds and $N_{max} = 20$ seconds, DISTBIC_pR with $pRange = 1$ second, and FixSlid with a 2-second sliding window.

TABLE IV
THE DERS (IN %) OF DIFFERENT DIARIZATION SYSTEMS. VITERBI RE-SEGMENTATION IS NOT APPLIED.

Approach	RT03_Dev				RT03_Eval			
	MiS	FaS	SpE	DER	MiS	FaS	SpE	DER
SeqDACDec1_HAC	0.6	0.4	7.9	8.86	0	4	9.3	13.34
SeqDACDec2_HAC	0.6	0.4	7.7	8.7	0	4	9.4	13.39
DACDec3_HAC	0.6	0.4	7.5	8.46	0	4	9.7	13.69
WinGrow_HAC	0.6	0.4	8.3	9.29	0	4	10.1	14.12
DISTBIC_pR_HAC	0.6	0.4	8.2	9.19	0	4	9.9	13.94
FixSlid_HAC	0.6	0.4	10.5	11.52	0	4	13.3	17.57

a recursive variant of another popular approach, DISTBIC. We compared our approaches to these well-known approaches analytically by performing computational cost analysis. The results of experiments conducted on broadcast news data demonstrate that the proposed approaches are more efficient and achieve higher segmentation accuracy than the existing approaches discussed in this paper. In addition, we applied the segmentation approaches to the speaker diarization task. The experiment results show that a more accurate segmentation approach leads to better diarization accuracy.

TABLE V
THE DERS (IN %) OF DIFFERENT DIARIZATION SYSTEMS. VITERBI RE-SEGMENTATION IS APPLIED AS A POST PROCESSING STEP.

Approach	RT03_Dev				RT03_Eval			
	MiS	FaS	SpE	DER	MiS	FaS	SpE	DER
SeqDACDec1_HAC	0.6	0.4	7.4	8.37	0	4	9.2	13.15
SeqDACDec2_HAC	0.6	0.4	7.4	8.35	0	4	9.2	13.16
DACDec3_HAC	0.6	0.4	7	7.96	0	4	9.7	13.67
WinGrow_HAC	0.6	0.4	7.7	8.65	0	4	9.8	13.79
DISTBIC_pR_HAC	0.6	0.4	7.5	8.51	0	4	9.1	13.06
FixSlid_HAC	0.6	0.4	8.2	9.22	0	4	10.9	14.91

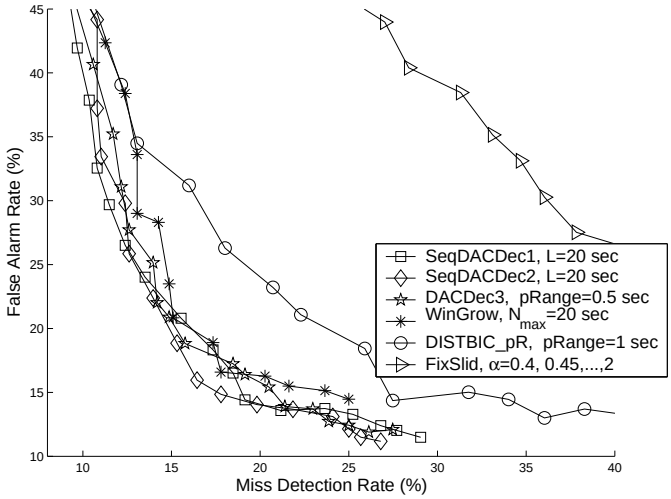


Fig. 16. The ROC curves for RT03.

APPENDIX

A. Compute $T_1(k)$

$T_1(k)$ is expressed as

$$T_1(k) = \frac{1}{k}\sum_{i=1}^{k}(T_1(i-1)+T_1(k-i)) + (k+1)^2m^2\tau$$
$$= \frac{2}{k}\sum_{i=1}^{k}T_1(i-1) + (k+1)^2m^2\tau, \quad (18)$$

where $T_1(0) = m^2\tau$. To solve this recursive equation, we can apply the technique used for analyzing the time cost of the Quicksort algorithm [39]. First, we multiply both sides of Eq. (18) by k as follows:

$$kT_1(k) = 2\sum_{i=1}^{k}T_1(i-1) + k(k+1)^2m^2\tau. \quad (19)$$

Replacing k in Eq. (19) with $k-1$, we obtain

$$(k-1)T_1(k-1) = 2\sum_{i=1}^{k-1}T_1(i-1) + (k-1)k^2m^2\tau. \quad (20)$$

Subtracting Eq. (20) from Eq. (19), we obtain

$$kT_1(k)-(k-1)T_1(k-1) = 2T_1(k-1)+(3k^2+k)m^2\tau. \quad (21)$$

Rearranging the terms in Eq. (21) yields

$$\frac{T_1(k)}{k+1} = \frac{T_1(k-1)}{k} + \frac{(3k+1)m^2\tau}{(k+1)}. \quad (22)$$

Let $a_k = T_1(k)/(k+1)$, then Eq. (22) can be rewritten as

$$a_k = a_{k-1} + (3 - \frac{2}{k+1})m^2\tau, \quad (23)$$

where $a_0 = m^2\tau$. Recursively substituting the a_ks' in Eq. (23), we obtain

$$a_k = a_0 + (3k - (\frac{2}{2}+\frac{2}{3}+\cdots+\frac{2}{k+1}))m^2\tau$$
$$= (3k+3-2\sum_{i=1}^{k+1}\frac{1}{i})m^2\tau. \quad (24)$$

Because $\sum_{i=1}^{k+1}\frac{1}{i}\approx\ln(k+1)$ [39], we have

$$a_k \approx (3k+3-2\ln(k+1))m^2\tau. \quad (25)$$

Since $a_k = T_1(k)/(k+1)$, $T_1(k)$ can be expressed as

$$T_1(k) = a_k(k+1)$$
$$\approx (3(k+1)^2 - 2(k+1)\ln(k+1))m^2\tau. \quad (26)$$

B. Compute $T'(m)$

$T'(m)$ is expressed as

$$T'(m) = \frac{1}{m}\sum_{i=1}^{m}(T'(i-1)+T'(m-i)) + m^2\tau + m\tau, \quad (27)$$

where $T'(0) = 0$. Similar to the manipulation of Eq. (18) in Appendix A, by setting $a_m = T'(m)/(m+1)$, we have $a_0 = 0$ and

$$a_m = a_{m-1} + \frac{(3m-1)\tau}{m+1}$$
$$= a_{m-1} + (3 - \frac{4}{m+1})\tau$$
$$= a_0 + (3m - (\frac{4}{2}+\frac{4}{3}+\cdots+\frac{4}{m+1}))\tau$$
$$\approx (3m+4-4\ln(m+1))\tau. \quad (28)$$

Since $a_m = T'(m)/(m+1)$, we have

$$T'(m) = a_m(m+1)$$
$$\approx (3m+4-4\ln(m+1))(m+1)\tau. \quad (29)$$

C. Compute $T_2(k)$

$T_2(k)$ is expressed as

$$T_2(k) = \frac{1}{k}\sum_{i=1}^{k}(T_2(i-1) + T_2(k-i)) + (k+1)^2 m^2\tau + 2m\tau, \tag{30}$$

where $T_2(0) \leq (3m + 4 - 4\ln(m+1))(m+1)\tau$. Similar to the manipulation of Eq. (18) in Appendix A, by setting $a_k = T_2(k)/(k+1)$, we have

$$\begin{aligned} a_k &= a_{k-1} + \frac{(3k^2+k)m^2\tau + 2m\tau}{k(k+1)}, \\ &= a_{k-1} + (3 - \frac{2}{k+1})m^2\tau + 2(\frac{1}{k} - \frac{1}{k+1})m\tau, \\ &\approx a_0 + (3k + 2 - 2\ln(k+1))m^2\tau \\ &\quad +2(\ln k - \ln(k+1) + 1)m\tau. \end{aligned} \tag{31}$$

Substituting $a_0 = T_2(0)$ into Eq. (31), we obtain

$$\begin{aligned} a_k \leq (3k + 5 - 2\ln(k+1))m^2\tau + (9 + 2\ln k - 2\ln(k+1) \\ -4\ln(m+1))m\tau + (4 - 4\ln(m+1))\tau. \end{aligned} \tag{32}$$

Since $a_k = T_2(k)/(k+1)$, we have

$$\begin{aligned} T_2(k) \leq (3k + 5 - 2\ln(k+1))(k+1)m^2\tau \\ +(9 + 2\ln k - 2\ln(k+1) - 4\ln(m+1))(k+1)m\tau \\ +(4 - 4\ln(m+1))(k+1)\tau. \end{aligned} \tag{33}$$

REFERENCES

[1] H. Meinedo and J. Neto, "Audio segmentation, classification and clustering in a broadcast news task," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, Hong Kong, China, Apr. 2003, pp. II–5–II–8.

[2] C. H. Wu and C. H. Hsieh, "Multiple change-point audio segmentation and classification using an MDL-based Gaussian model," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 14, no. 2, pp. 647–657, 2006.

[3] P. C. Woodland, M. J. F. Gales, D. Pye, and S. J. Young, "The development of the 1996 HTK broadcast news transcription system," in *Proc. DARPA Speech Recognition Workshop*, Lansdowne, VA, Feb. 1997, pp. 73–78.

[4] R. Bakis, S. Chen, P. S. Gopalakrishnan, R. Gopinath, S. Maes, L. Polymenakos, and M. Franz, "Transcription of broadcast news shows with the IBM large vocabulary speech recognition system," in *Proc. DARPA Speech Recognition Workshop*, Chantilly, VA, Feb. 1997, pp. 67–72.

[5] J. F. Bonastre, P. Delacourt, C. Fredouille, T. Merlin, and C. J. Wellekens, "A speaker tracking system based on speaker turn detection for NIST evaluation," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, Istanbul, Turkey, Jun. 2000, pp. 1177–1180.

[6] L. Lu and H. J. Zhang, "Unsupervised speaker segmentation and tracking in real-time audio content analysis," *ACM/Springer Multimedia Systems Journal*, vol. 10, no. 4, pp. 1432–1882, 2005.

[7] C. Barras, X. Zhu, S. Meignier, and J.-L. Gauvain, "Multistage speaker diarization of broascast news," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1505–1512, 2006.

[8] S. E. Tranter and D. A. Reynolds, "An overview of automatic speaker diarization systems," *IEEE Trans. Audio, Speech and Language Processing*, vol. 14, no. 5, pp. 1557–1565, 2006.

[9] P. Delacourt and C. J. Wellekens, "DISTBIC: A speaker-based segmentation for audio data indexing," *Speech Communication*, vol. 32, no. 1, pp. 111–126, 2000.

[10] M. Siegler, U. Jain, B. Raj, and R. Stern, "Automatic segmentation, classification and clustering of broadcast news audio," in *Proc. DARPA Speech Recognition Workshop*, Chantilly, VA, Feb. 1997, pp. 97–99.

[11] S. Chen and P. S. Gopalakrishnan, "Speaker, environment and channel change detection and clustering via the Bayesian information criterion," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, Lansdowne, VA, Feb. 1998, pp. 127–132.

[12] J. W. Hung, H. M. Wang, and L. S. Lee, "Automatic metric-based speech segmentation for broadcast news via principal component analysis," in *Proc. Int. Conf. Spoken Language Processing*, Beijing, China, Oct. 2000, pp. 121–124.

[13] M. Cettolo, M. Vescovi, and R. Rizzi, "Evaluation of BIC-based algorithms for audio segmentation," *Computer Speech and Language*, vol. 19, pp. 147–170, 2005.

[14] H. Gish, M. H. Siu, and R. Rohlicek, "Segregation of speakers for speech recognition and speaker identification," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, Toronto, Canada, Apr. 1991, pp. 873–876.

[15] P. Sivakumaran, J. Fortuna, and A. M. Ariyaeeinia, "On the use of the Bayesian information criterion in multiple speaker detection," in *Proc. Eur. Conf. Speech Communication and Technology*, Aalborg, Denmark, Sep. 2001, pp. 795–798.

[16] X. Zhong, M. Clements, and S. Lim, "Acoustic change detection and segment clustering of two-way telephone conversation," in *Proc. Eur. Conf. Speech Communication and Technology*, Geneva, Switzerland, Sep. 2003, pp. 2925–2928.

[17] B. W. Zhou and J. H. L. Hansen, "Efficient audio stream segmentation via the combined T^2 statistic and Bayesian information criterion," *IEEE Trans. Speech and Audio Processing*, vol. 13, no. 4, pp. 467–474, 2005.

[18] W. N. Chan, N. Zheng, and T. Lee, "Discrimination power of vocal source and vocal tract related features for speaker segmentation," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 15, no. 6, pp. 1884–1892, 2007.

[19] X. Anguera, "XBIC: Real-time cross probabilities measure for speaker segmentation," Tech. Rep. TR-05-008, Univ. California Berkely, ICSI-Berkely, Aug. 2005.

[20] T. Zhang and C.-C. Jay Kuo, "Audio content analysis for online audiovisual data segmentation and classification," *IEEE Trans. Speech and Audio Processing*, vol. 9, no. 4, pp. 441–457, 2001.

[21] L. Lu, H. J. Zhang, and H. Jiang, "Content analysis for audio classification and segmentation," *IEEE Trans. Speech and Audio Processing*, vol. 10, no. 7, pp. 504–516, 2002.

[22] R. Huang and J. H. L. Hansen, "Advances in unsupervised audio classification and segmentation for the broadcast news and NGSW corpora," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 14, no. 3, pp. 907–919, 2006.

[23] P. Sivakumaran, A. M. Ariyaeeinia, and J. Fortuna, "An effective unsupervised scheme for multiple-speaker-change detection," in *Proc. Int. Conf. Spoken Language Processing*, Colorado, USA, Sep. 2002, pp. 569–572.

[24] L. Lu and H. J. Zhang, "Speaker change detection and tracking in real-time news broadcasting analysis," in *Proc. ACM Multimedia*, Juan-les-Pins, France, Dec. 2002, pp. 602–610.

[25] A. S. Malegaonkar, A. M. Ariyaeeinia, and P. Sivakumaran, "Efficient speaker change detection using adapted gaussian mixture models," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 15, no. 6, pp. 1859–1869, 2007.

[26] P. C. Lin, J. C. Wang, J. F. Wang, and H. C. Sung, "Unsupervised speaker change detection using SVM training misclassification rate," *IEEE Trans. Computers*, vol. 56, no. 9, pp. 1212–1223, 2007.

[27] G. Schwarz, "Estimation the dimension of a model," *Annals of Statistics*, vol. 6, pp. 461–464, 1978.

[28] C. Fraley and A. E. Raftery, "How many clusters? which clustering method? answers via model-based cluster analysis," *Computer Journal*, vol. 41, pp. 578–588, 1998.

[29] A. Tritschler and R. A. Gopinath, "Improved speaker segmentation and segments clustering using the Bayesian information criterion," in *Proc. Eur. Conf. Speech Communication and Technology*, Budapest, Hungary, Sep. 1999, pp. 679–682.

[30] Y. Huang, O. Vinyals, G. Friedland, C. Müller, N. Mirghafori, and C. Wooters, "A fast-match approach for robust, faster than real-time speaker diarization," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding*, Kyoto, Japan, Dec. 2007, pp. 693–698.

[31] H. M. Wang, B. Chen, J. W. Kuo, and S. S. Cheng, "MATBN: A Mandarin Chinese broadcast news corpus," *Int. Journal of Computational Linguistics and Chinese Language Processing*, vol. 10, no. 2, pp. 219–236, 2005.

[32] Linguistic Data Consortium, *2003 NIST Rich Transcription Evaluation Data*, http://www.ldc.upenn.edu/.

[33] S. S. Cheng and H. M. Wang, "A sequential metric-based audio segmentation method via the Bayesian information criterion," in *Proc. Eur. Conf. Speech Communication and Technology*, Geneva, Switzerland, Sep. 2003, pp. 945–948.

[34] Linguistic Data Consortium, *2001 NIST Speaker Recognition Evaluation Corpus*, http://www.ldc.upenn.edu/.

[35] S. Meignier, D. Moraru, C. Fredouille, J.-F. Bonastre, and L. Besacier, "Step-by-step and integrated approaches in broadcast news speaker diarization," *Computer Speech & Language*, vol. 20, pp. 303–330, 2006.
[36] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing (DSP), a Review JournalXSpecial Issue on NIST 1999 Speaker Recognition Workshop*, vol. 10, pp. 19–41, 2000.
[37] J. M. Pardo, X. Anguera, and C. Wooters, "Speaker diarization for multiple-distant-microphone meetings using several sources of information," *IEEE Trans. Computers*, vol. 56, no. 9, pp. 1212–1224, 2007.
[38] NIST, *Rich Transcription Spring 2006 Evaluation*, http://www.nist.gov/speech/tests/rt/2006-spring/index.html.
[39] U. Manber, *Introduction to Algorithms: A Creative Approach*, Addison-Wesley, 1989.

Shih-Sian Cheng received the B.S. degree in mathematics from National Kaohsiung Normal University, Kaohsiung, Taiwan, in 1999 and the M.S. degree in computer science from National Chiao Tung University, Hsinchu, Taiwan, in 2002. He is currently working towards the Ph.D. degree at the Department of Computer Science, National Chiao Tung University.

In 2002, he joined the Spoken Language Group, Chinese Information Processing Laboratory, Institute of Information Science, Academia Sinica, Taipei, Taiwan, as a Research Assistant. His research interests include machine learning, pattern recognition, speech processing, and neural networks.

Hsin-Min Wang received the B.S. and Ph.D. degrees in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1989 and 1995, respectively.

In October 1995, he joined the Institute of Information Science, Academia Sinica, Taipei, Taiwan, as a Postdoctoral Fellow. He was promoted to Assistant Research Fellow and then Associate Research Fellow in 1996 and 2002, respectively. He was an adjunct associate professor with National Taipei University of Technology and National Chengchi University. He was a board member and chair of academic council of ACLCLP. He currently serves as secretary-general of ACLCLP and as an editorial board member of International Journal of Computational Linguistics and Chinese Language Processing. His major research interests include speech processing, natural language processing, spoken dialogue processing, multimedia information retrieval, and pattern recognition.

Dr. Wang was a recipient of the Chinese Institute of Engineers (CIE) Technical Paper Award in 1995. He is a life member of ACLCLP and IICM and a member of ISCA.

Hsin-Chia Fu received the B.S. degree from National Chiao-Tung University in Electrical and Communication engineering in 1972, and the M.S. and Ph.D. degrees from New Mexico State University, both in Electrical and Computer Engineering in 1975 and 1981, respectively. From 1981 to 1983 he was a Member of the Technical Staff at Bell Laboratories. Since 1983, he has been on the faculty of the Department of Computer science and Information engineering at National Chiao-Tung University, in Taiwan, ROC. He is also the Taiwan representative of TEI Consortium since 2003. Form 1987 to 1988, he served as the director of the department of information management at the Research Development and Evaluation Commission, of the Executive Yuan, ROC. From 1988-1989, he was a visiting scholar of Princeton University. From 1989 to 1991, he served as the chairman of the Department of Computer Science and Information Engineering. From September to December of 1994, he was a visiting scientist at Fraunhofer-Institut for Production Systems and Design Technology (IPK), Berlin Germany. His research interests include digital signal/image processing, Multimedia information processing, and neural networks. Dr. Fu was the co-recipient of the 1992 and 1993 Long-Term Best Thesis Award with Koun Tem Sun and Cheng Chin Chiang, and the recipient of the 1996 Xerox OA paper Award. He has served as a founding member, Program co-chair (1993) and General co-chair (1995) of International Symposium on Artificial Neural Networks. He has been the Technical Committee on Neural Networks for Signal Processing of the IEEE Signal Processing Society from 1997 to 2000. He has authored more than 100 technical papers, and two textbooks "PC/XT BIOS Analysis", and "Introduction to neural networks", by Sun-Kung Book Co., and Third Wave Publishing Co., respectively. Dr. Fu is a member of the IEEE Signal Processing and Computer Societies, Phi Tau Phi, and the Eta Kappa Nu Electrical Engineering Honor Society.