Speaker, Channel and Environment Change Detection

Homayoon S.M. Beigi and **Stéphane H. Maes**
IBM T.J. Watson Research Center
P.O. Box 218 - Yorktown Heights
New York 10598 U.S.A.
Phone: +1 (914) 945-1894
Fax: +1 (914) 243-4965
EMail: beigi@watson.ibm.com

Abstract

This paper presents a classification technique for segmenting an audio stream into pieces containing different speakers, music, noise and different background conditions. This segmentation is useful in tasks such as the *ARPA'95 HUB4* evaluation consisting of automatic transcription of radio broadcast news shows from the *Market Place* program.[1] The segmentation method used here is based on a discriminative distance measure between two adjacent sliding windows operating on preprocessed speech. The preprocessing is done by the *IBM* speech recognition front-end which converts raw speech into vectors of Mel-Warped Cepstral features at a rate of 100 frames per second.

Keywords: Speech, Recognition, Segmentation, Classification, Modeling, Preprocessing, HUB4

1 Introduction

Automatic segmentation of an audio stream into parts containing the speech of distinct speakers, music, noise, and different background conditions has attracted quite a bit of attention. This type of segmentation can aid the performance of recognizers which operate best with some a-priori knowledge of the speaker. Also different specialized recognizers may be used for performing a better recognition. An example of such task is the *ARPA'95 HUB4* evaluation task consisting of automatic transcription of radio broadcast news shows from the *Market Place* program.[1]

In multiple applications, detection of speaker changes will soon become a crucial element of speech recognition and speaker recognition engines. Speaker change detection is useful, in performing adequate un-supervised adaptation, using appropriate models for decoding and adaptive processing of the input associated with different speakers (e.g. using an appropriate dialog based on the identity of the speaker).

It is worth noting that most speech recognizers will break down if they are presented with music instead of speech. Therefore, it is important to separate the music from recognizable speech. In addition, one may wish to remove all the music and only store the speech in an archiving scenario to save space.

A typical radio broadcast news contains speech and non-speech signals from a large variety of sources like clean speech, band-limited speech (produced by some types of microphones), telephone speech, music segments, speech over music, speech over ambient noise, speech over speech, etc...

It is important to segment the data according to different categories [2, 3]: clean speech, telephone-quality speech (telephone speech and some band-limited microphones), speech with music and speech with noise. Different models (e.g., mixtures of Gaussians) can then be trained over each of these classes and used for decoding each condition.

Furthermore, in order to perform un-supervised adaptation on the data, it is useful to classify the data into segments associated with same speakers. We qualify this operation as speaker classification since it performs a classification of the speech associated to an unknown number of unknown speakers.

Different strategies may be considered for detecting speaker changes. These techniques fall into two general categories:

- Detection of speaker identity changes, by performing sequential speaker identifications and detecting a change of decision. Such approaches are described in [2, 3, 4].

- Detection of significant changes in the acoustic behavior followed by interpretation of the nature of these changes. Such approaches have been previously proposed in [5].

The algorithm described in this paper belongs to the second category. A draw-back of the method described in [5] is its extremely poor detection of the exact location of speaker changes. Furthermore, the resulting segmentation end-times rarely correspond to pauses, silences or breath noises. Unfortunately, this is unacceptable when the segmentation is followed by a speech recognition phase. Indeed, every time that a boundary is introduced in the middle of a word, it introduces one or two word errors. It is also a characteristic of methods which arbitrarily cut the input speech into small segment and then re-cluster them.[6, 7]

2 Boundary detection

The proposed algorithm uses adjacent windows which are shifted in time over the whole data. Acoustic features are computed over these windows and clustered. Different clustering algorithms are compared in terms of their speed and performance.

As we will illustrate later, the number of resulting clusters plays a significant role in the segmentation performance. For example, with too few clusters it is impossible to detect speaker/condition changes when the change occurs amid silences; with too many clusters, the selected end-times first become erratic and then random.

We have tried several distance measures between clusters associated with adjacent windows. Choice of the distance measure is crucial: different choices lead to the detection of different changes. Only a few distances correctly detect speaker, channel and condition changes and place the end-times amid pauses, silences and breath noises. Most will miss events or cover the speech data with a plethora of irrelevant end-times. We present a discriminative distance measure which is quite suitable for this task.

A decision algorithm is then used to select the appropriate distance peaks as segmentation end times.

2.1 Distance Measure

The raw speech sampled at normal sampling rates such as 8, 11 or 16 kHz is fed into the *IBM* speech recognition front end which produces a vector of Mel-Warped Cepstral features for each $10ms$ of the speech data. In the experiments performed in this paper, an n-dimensional (n=24) Mel-Warped Cepstral Feature vector is used.[8]

Two adjacent windows with the width of $w = 100$ frames are then placed at the beginning of the stream of feature vectors (see Figure 1). To maximize discriminability, there is no overlap between the windows. This is in contrast to other reports [5] which have used such overlap in the past. The general idea is to find the distance between the two groups of features in the adjacent windows and to identify those locations where this distance grows. These locations are potential segmentation points. The distance measure which is used here should be a good measure of the difference between the two adjacent windows in the sense that is important for telling different conditions apart. Let us consider the different types of features we may expect in the set of feature vectors present in a window. There are features associated with silence which are basically similar in both adjacent windows. There are also features associated with parts of speech which are common among different speakers. In addition, there are features which are related with the variable part of speech which would tell apart different individuals. Therefore, one may consider clustering the feature vectors in each of the adjacent windows into three distinct classes. These classes may be assumed to have a Gaussian distribution and hence we would be able to parameterize them with their means, variances and counts.

To obtain these clusters, we perform a bottom-up clustering by randomly picking three cluster centers in each window and running the K-Means algorithm [8] to come up with new cluster centers, in an iterative fashion. Eventually, after the convergence of the K-Means algorithm, three 24-dimensional vectors of means and their corresponding variances are available. The variance vector is also 24-dimensional since a diagonal covariance matrix is assumed. In addition, each cluster has a count associated with it.

Next, a set of three distance measures are computed for the two closest, the next closest and furthest clusters between the two adjacent windows. The assumption is that the largest distance is a good measure

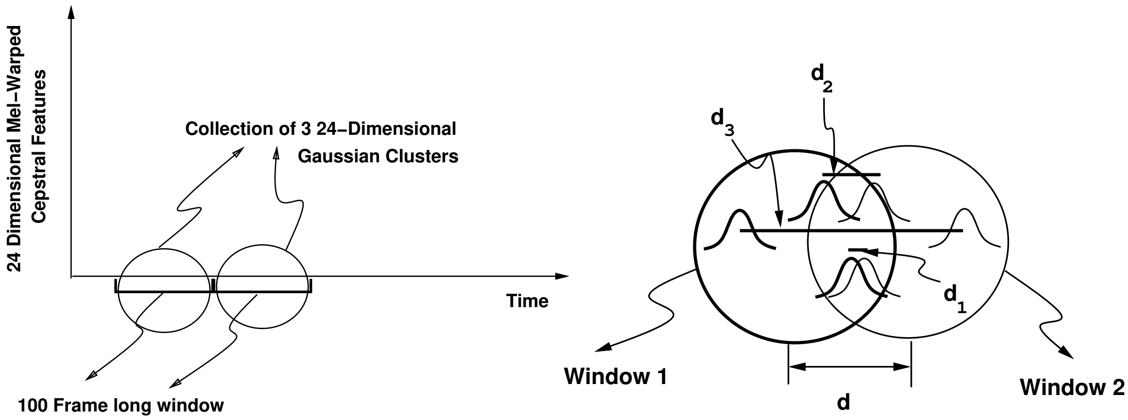

Figure 1: (left) Adjacent Windows Used for Discriminating Between Sounds (right) Distances Between Gaussians

of the distance between the two sets of features which are associated with differences among speakers. This distance is normalized with the smallest distance which is representative of segments of speech which are most alike such as silences. A separate distance, d, is also computed which is based on a one Gaussian cluster in each window. This distance is basically the total difference between the two windows.

Therefore the discriminative distance D is given by,

$$D^i = \frac{\mu d_3^i / d_1^i}{d^i} \tag{1}$$

where for the ith adjacent window pair, d_3^i is the largest distance, d_1^i is the smallest distance, d^i is the distance between the two adjacent windows with a single Gaussian fit to each window. μ is the mean value of all d^i computed on all window pairs i in the sample (see figure 1).

2.2 Automatic Segmentation

Figure 2 shows the plot of distance measure D given by equation 1. Once this distance D is computed for a whole audio segment, we would like to find probable segmentation points along the time line. These segmentation points may be over-segmenting the audio, but we would like to avoid missing true segmentation points. The algorithm for automatic determination of the segmentation points is as follows:

The distance signal is swept from left to right and as soon as a threshold, α, is reached, the maximum value is found until the point when the threshold, α, is reached again from the top. At this point, this maximum value entry is added to a list. After traversing the whole distance curve, D, the list is sorted by the magnitude of D. Then, looking at the largest magnitude, the indices of all the other elements of the list are compared to see if they are within some threshold, β, away from the entry with the largest magnitude. If their indices are within some threshold, β, of the index of the largest entry, they will be taken out of the list and the list is started again with the next largest entry in the list in the same manner.

The automatic segmentation points obtained by the above algorithm have been plotted in figure 2. As it is apparent from figure 2, the threshold of the segmentation location dynamically changed and the segmentation algorithm ignores the locations around frames 280 and 350. This is partly due to the fact that the local maximum is picked by the algorithm.

Once these segmentation points are obtained, adjacent segments may be re-merged. This merging is due to the initial design of the system to perform an over-segmentation to avoid missing important segmentation points. One method for merging alike segments is to perform a speaker clustering on the segments produced by the segmenter. In whatever speaker clustering which is used, one should make sure that there is a high penalty for merging non-adjacent segments. The following section describes one such algorithm.

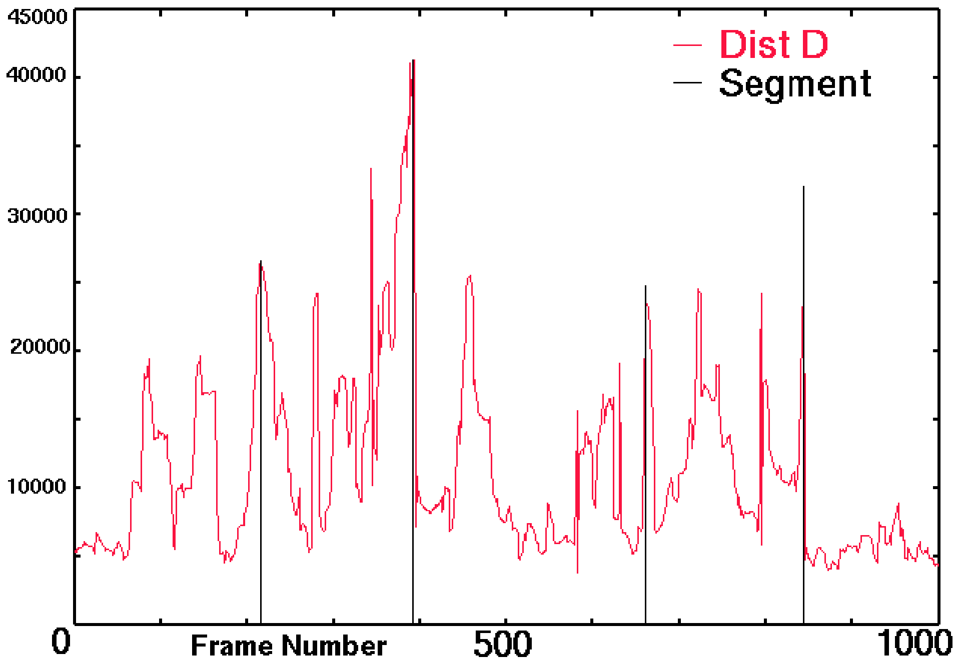

Figure 2: Plot of the Discriminative Distance D

3 Clustering and Integration

Once the initial segmentation is done by the method prescribed in the previous section, the over-segmented pieces are re-clustered to obtain clusters containing the same speakers, music or other non-speech segments.

Depending on the application, resulting end-times can be further processed to separate segmentation points attributed to pauses, silences and breath noises from the end-times associated with changes.

Re-Clustering of the features associated with each new segment can be used to eliminate spurious end-times. The clustering algorithm that perform the best is different from that which has previously been used with the *"chop and re-cluster"* method [6, 7]. In this paper we use a distance measure we have developed to provide a good measure of the difference between two mixtures of Gaussians. Also, based on the same concept from which this distance measure was derived, we have developed a merging technique for merging models (mixtures of Gaussians) without the need for the original data. The distance measure and the merging technique have been described in [9].

In the particular case of radio or TV data, we also take advantage of the segmentation technique described in [2, 3, 4] as an additional stopping criterion for the clustering.

To reduce the chance of mis-segmentation, the alignment information from the IBM continuous speech recognizer's Hidden Markov Models was used to produce conjectures for the locations of silences. This information was then combined with the distance information produced by equation 1 to come up with estimates for the segmentation location which would not cut the speech unless there is a silence at that location. Figure 3 shows all the steps taken in determining the segmentation of the speech stream.

4 Results

The *ARPA'95 HUB4* data of the *Market Place* program was used for testing the proposed segmentation. Eighteen audio streams were available for this test. The typical length of each stream is about 450 seconds. Figure 2 shows distance D for the first 10 seconds of one of these streams. Overall, there were no errors made in finding the boundary between speech and non-speech segments. However, there was an error of about 30% in identifying the correct segmentation point between two adjacent speakers. Also, in some cases, although the distance measure, D, showed definite local maxima, these maxima were ignored by

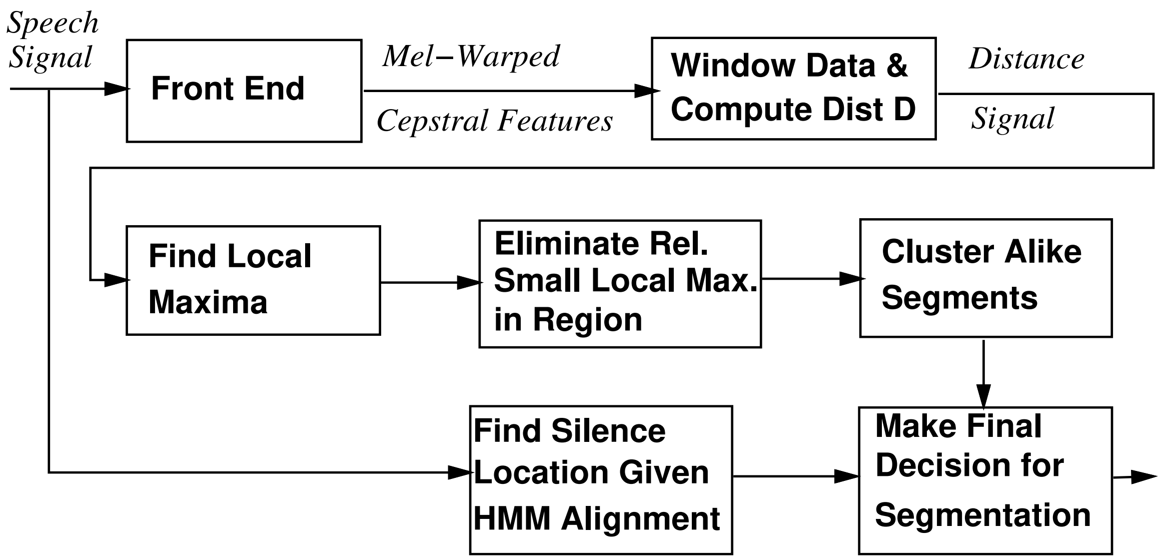

Figure 3: Segmentation Procedure

the automatic peak-finding algorithm. The nature of some of these errors was mainly due to delays of a fraction of a second in the segmentation location and not due to a total miss.

The authors believe that the behavior of the peak-finding algorithm is due to scaling problems between *abrupt* changes such as music to speech and *more subtle* changes such as from one speaker to another. The impulsive value of the distance measure D is much larger for change of conditions such as music to speech than that created by a speaker change. Therefore, the threshold, α, taken to locate the local maxima of D may only be optimized for either of these cases. If the threshold is taken to be very large to correctly find drastic changes of conditions, it may miss the more subtle peaks generated by the change of speakers. On the other hand, if the threshold is taken to be small such that it won't miss any of the speaker changes, it may produce too many spurious segments based on the background condition mismatches such as change of music or background noise. One way to deal with this problem is to reduce the threshold and let the re-clustering take care of the extraneous segments in the music part.

Another source of errors is due to the fact that for the sake of practicality, we used the alignment information from the IBM continuous speech recognizer to come up with hypotheses for silence locations. Once these silences were found, only the speaker changes coincident with these silence locations were taken. However, there are many cases when there is a change of speakers, but there is no silence at that time, since one speaker would be followed by another with some overlap. In these situations, for some fraction of time both speakers are speaking at the same time. This speaker change would be detected by the segmentation algorithm, but rejected since there is no silence present at the boundary. In fact, looking at some part of the segmentation results without, the imposition of the silence criterion, almost no errors were detected. The reason for using this silence criterion is that we would not like to break a word into two and increase the chance of occurance of errors due to this un-natural breakage.

5 Conclusion

It is important to note the following items in the production of above results:

- Only silence location were considered. This eliminates the non-silence transition, hence increasing the error.

- Since there is no automatic method of finding the statistics, the streams were listened to and only the segmentation which coincided with silences were counted as correct if they were exactly at the right place. This means that the accuracy would increase by increasing the tolerance level for mistakes by a few frames.

Also, as mentioned earlier, sometimes the correct speaker change segmentation is performed with a small delay. This may be due to the window size. That is, if the clustering windows are taken too small, then they would give better results with the abrupt changes such as music to speech transitions. However, to get a good segmentation of speakers, larger windows should be used such that there is enough data in each window to be able to decide on a segmentation point. This problem may also be addressed by using more Gaussians in order to get a better model for the speakers and the change in the speakers. This may be done by doing a better modeling with higher number of Gaussians or a Hidden Markov Model.

Unfortunately, it is very hard to come up with a standard for counting errors in segmentation since segmentation is a purely subjective task. Even two people listening to the same speech may segment it differently depending on what they are looking to find in a segment. Some work is in progress to use this segmentation algorithm for many different applications. One most important application is to train and decode the different segments using more suitable models for the extracted segment so that the overall speech recognition accuracy is increased for such segments with a variety of different audio sources.

References

[1] ARPA, Proceedings of the Speech Recognition Workshop, Arden House, Harrsiman, NY, February 1996.

[2] P.S. Gopalakrishnan and R. Gopinath and S. Maes and M. Padmanabahn and L. Polymenakos, "Acoustic models used in the *IBM* system for the ARPA HUB4 task," Proceedings of the Speech Recognition Workshop, ARPA, February 1996.

[3] R. Bakis and S. Chen and P.S. Gopalakrishnan and R. Gopinath and S. Maes and L. Polymenakos, "Transcription of broadcast news - *System robustness issues and adaptation techniques*," preprint submitted to ICASSP'97, ARPA, August 1996.

[4] R. Bakis, S. Chen, P. Gopalakrishnan, R. Gopinath, S. Maes and L. Polymenakos, "Transcription of Broadcast News Shows with the *IBM* Large Vocabulary Speech Recognition System," Proceedings of the Speech Recognition Workshop, Chantilly, VA, February 1997.

[5] M. A. Sigler and U. Jain and B. Raj and R. M. Stern, "Automatic Segmentation, Classification and Clustering of Broadcast News Audio," Proceedings of the Speech Recognition Workshop, Chantilly, VA, February 1997.

[6] H. Jin and F. Kubala and R. Schwartz, "Automatic Speaker Clustering," Proceedings of the Speech Recognition Workshop, Chantilly, VA, February 1997.

[7] L. Heck and A. Sankar, "Acoustic Clustering and Adaptation for Improved Speech Recognition," Proceedings of the Speech Recognition Workshop, Chantilly, VA, February 1997.

[8] L. Rabiner and B.H. Juang, Fundamentals of Speech Recognition, Prentice Hall Signal Processing Series, Alan V. Oppenheim, Series Ed., New Jersey, 1993, pp. 125-128.

[9] Homayoon S. M. Beigi, Stéphane H. Maes and J. Sorensen, "A Distance Measure Between Collections of Distributions and Its Application to Speaker Recognition," ICASSP'98, Detroit, 1998.