Acoustic Segmentation for Audio Browsers

Don Kimber

Xerox PARC
Palo Alto, CA 94304

Lynn Wilcox

FX Palo Alto Laboratory
Palo Alto, CA 94304

Abstract

Online digital audio is a rapidly growing resource, which can be accessed in rich new ways not previously possible. For example, it is possible to listen to just those portions of a long discussion which involve a given subset of people, or to instantly skip ahead to the next speaker. Providing this capability to users, however, requires generation of necessary indices, as well as an interface which utilizes these indices to aid navigation.

We describe algorithms which generate indices from automatic acoustic segmentation. These algorithms use hidden Markov models to segment audio into segments corresponding to different speakers or acoustics classes (e.g. music). Unsupervised model initialization using agglomerative clustering is described, and shown to work as well in most cases as supervised initialization.

We also describe a user interface which displays the segmentation in the form of a timeline, with tracks for the different acoustic classes. The interface can be used for direct navigation through the audio.

1 Introduction

Online digital audio is rapidly gaining importance as an information resource. Rather than being restricted to real time sequential listening, as with traditional audio media, it is now possible to listen to recordings in entirely new ways. For example, recordings may be played faster than real time without pitch distortion [1], although comprehension is lost after about a factor of two. It is also possible to skim speech by listening only to segments following a long silence [2]. Random access to audio allows a user to instantly skip forward or backward to a desired location within a recording, without requiring fast forward or reverse as in sequential media. Taking advantage of this capability, however, requires generation of necessary indices, as well as an interface which utilizes these indices to aid navigation.

Audio indices may be generated by human effort, as part of an authoring process. However this is extremely time consuming, and methods of producing indices automatically or semi-automatically are desirable. There are a number of ways of generating useful indices automatically, such as keyword spotting [19], detection of regions of emphatic speech [4], and alignment of speech with textual transcription [7]. Another method is based on segmentation of audio into regions corresponding to different speakers or acoustic classes [20]. In this paper, we review this method and describe our experiences with its use on various types of audio recordings. We also describe a graphical audio tool which both assists in the process of generating an audio segmentation, and once such a segmentation is provided, utilizes the indices to aid in navigation and special playback modes. These include skipping to the next speaker, and playing only portions of recordings involving a subset of speakers.

Our basic framework for audio segmentation involves hidden Markov models [12]. In this framework, speech is represented by sequences of feature vectors, each of which provides a short term characterization of the signal.[1] A hidden Markov model (HMM) includes a set of states, transition probabilities among the states, and output probabilities, which specify the conditional probability density of feature vectors for each given state. Such models induce probability densities over sequences of feature vectors.

A hidden Markov model is created for each speaker or acoustic class. Given a training sequence for a class, the HMM for the class is trained using a maximum likelihood estimation procedure known as the Baum-Welch algorithm. These class HMMs are then combined into a larger network, which is itself a hidden Markov model. Segmentation is achieved by using the Viterbi algorithm to determine the maximum likelihood state sequence through this network, given an observed sequence of feature vectors, and noting those times at which the state sequence passes between states associated with different classes.

This segmentation technique is similar to that used by

[1] Our feature vectors correspond to cepstra computed over short (20 msec) windows of sampled speech.

Sugiyama *et al* [17]. However, in contrast to Sugiyama, where single state class models were used, our acoustic class models have multiple states with Gaussian output distributions, as shown in Figure 2. This form of model was used by Wilcox and Bush [19] to model non-keyword speech for speaker dependent word spotting. Similar non-phonetic [9] and phonetic [5] models have also been applied to speaker identification, but segmentation was not considered. The number of states in the HMM depends on the acoustic class. We use 32 states for speaker models, 3 states for silence, and 64 states for music.

The models must be trained initially given some labeled data for each acoustic class. This may be obtained by hand labeling a portion of the audio recording, or by an enrollment procedure in which a sample of sound from each speaker or acoustic class must be provided. The class models are then trained and used to segment the audio signal. The segmentation can be performed in a single pass in real time, and if the models are well trained, this segmentation may be adequate. However, the quality of the segmentation can often be improved by retraining the models based on the computed segmentation and then using the new models to resegment the data. We refer to this as iterative resegmentation.

When iterative resegmentation is used, it is important to have good initial estimates for the acoustic class models. One might hope that random initialization of the models could be used, and that iterative resegmentation would lead to a convergence of the models to the desired classes. Indeed this sometimes works, but we have found that it is unreliable, and sometimes leads to quite poor segmentations. Hand labeling can be used to provide good initialization, but unfortunately labeling sufficient quantities of data[2] can be quite time consuming. For example, in a one hour recording in which a given person speaks for a total of only two minutes, even locating thirty seconds of that person's speech may be difficult to do quickly.

We have found that automatic clustering is an effective way of generating the required labeled data for model initialization. In this approach, the recording is divided into short equal length segments of a few seconds each, and a bottom up (agglomerative) clustering algorithm is used to repeatedly merge clusters until a single cluster remains. This produces a tree structure, from which any desired number of clusters can be found. Ideally, the number of clusters chosen would be equal to the number of acoustic classes in the audio. However, due to inaccuracies in the agglomerative clustering, resulting in part from some initial segments containing data from multiple classes, more clusters than classes are required to obtain clusters that are well correlated with a single speaker or sound. Clusters are associated with speakers or sounds by using the audio browsing tool described in Section 4 to listen to the segments within clusters. This is much easier than hand labeling data for each speaker or sound in the recording.

The results for initialization based on agglomerative clustering depend on the distance measure between speech segments. We use the likelihood ratio statistic proposed by Gish *et al* [6], but extend it by replacing the Gaussian distributions with tied Gaussian mixtures. We also recompute distances between merged segments at each level of the hierarchical clustering and augment the distance with a duration model. When hierarchical clustering using the tied mixtures in the likelihood ratios was used to initialize the acoustic class models, acoustic segmentation accuracy equaled that obtained with supervised initialization of the class models.

2 Acoustic Segmentation

2.1 Segmentation Network

The segmentation network is composed of a hidden Markov model for each acoustic class. Types of acoustic classes include speakers, silence, laughter, non-speech sounds such as music, and garbage. Garbage is defined as speech or sound not explicitly modeled by the other class models, for example, unknown speakers in the audio. Figure 1 shows the structure of a segmentation network for N acoustic classes. The transition probabilities from the initial null state to the acoustic classes are uniform. [3] The transition probability out of each class is set to a constant ϵ. In principle, these transition probabilities could depend on the class, and could be learned during training. However, for simplicity, the prior probabilities of acoustic classes are assumed to be uniform, and the exiting probability ϵ is selected empirically to discourage class change based on isolated samples.

Figure 2 shows a model for an acoustic class. The model consists of L states $S_1, ...S_L$ connected in parallel. Each state has a self transition and an exiting transition. The output distribution for each state is Gaussian, parameterized by a mean vector and a diagonal covariance matrix. States correspond roughly to the different sounds produced by the acoustic class. For speech, these states can be thought of as the phones produced by the speaker. The sound associated with a state is characterized by the mean and covariance matrix of the Gaussian

[2] We have found that approximately one minute of labeled data is required for good initialization.

[3] A null state is a state not associated with any output or observation vector.

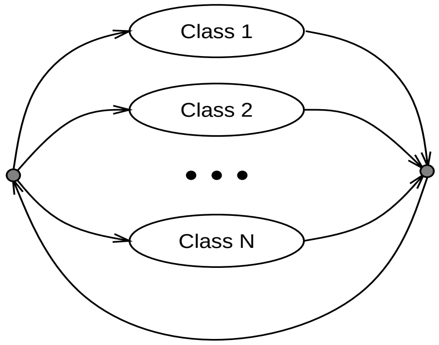

Figure 1: Segmentation Network

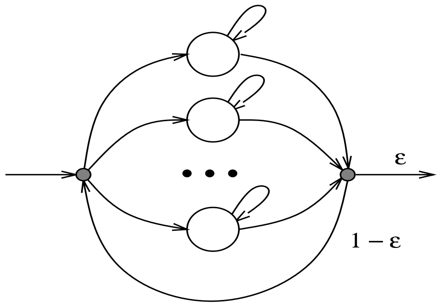

Figure 2: Acoustic Class Model

output distribution, while the duration of the sound is modeled by the self transition probability.

2.2 Basic Segmentation

We now describe how the segmentation networks are used to generate segmentation indices. First it is necessary to train the acoustic class models, which requires labeled training sequences for each class. Given these sequences, the model parameters are estimated using the Baum-Welch algorithm [12].

Once the class models have been trained, they are combined into the segmentation network. Then, given this network and the sequence of features corresponding to an acoustic signal, segmentation is performed by using the Viterbi algorithm to find the maximum likelihood sequence of states through the network. Segmentation indices are produced by simply noting the times at which the optimal state sequences passes from a state in one class model to a state in another. Note that this can be performed as a real time operation, by using a continuous traceback in the Viterbi algorithm [3]. This allows segmentation to be performed during recording, or in fact on signals which are not being recorded.

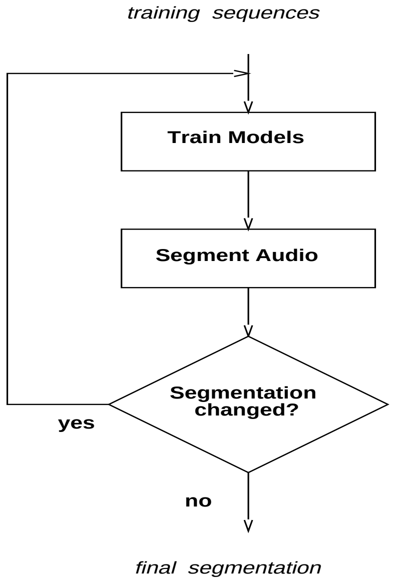

Figure 3: Iterative Resegmentation

2.3 Iterative Resegmentation

Once a recording has been segmented, the portions of the audio associated with each model can be used to retrain that model. These models can then be used to resegment the data. This process can be repeated to produce successive segmentations as shown in Figure 3. We have found that this iterative resegmentation leads to improved results, particularly when the initial models are not well trained. Poorly trained models can arise because the initial models are trained on a limited amount of hand labeled data, or because the initial models are trained on sequences produced by unsupervised clustering which may produce good but not perfect results.

3 Unsupervised Segmentation

3.1 Agglomerative Clustering

When labeled audio sequences are unavailable for training, unsupervised clustering can be used instead, to provide data for training the initial class models. The audio recording is first divided into equal length segments of several seconds each. These segments can then be clustered by either a top down (k-means) or bottom up (agglomerative) method. We were unable to obtain good results using k-means clustering, but found agglomerative clustering to be very effective. The agglomerative clustering described here is an elaboration of the method suggested by Gish *et al* [15]. In this procedure, the equal length segments are used to initialize a set of clusters, where a cluster $\mathcal{X}$ consists of a set of one or more segments $\mathcal{X} = \{x_1, x_2, \ldots\}$. Each singleton $\{x_i\}$ is an initial cluster. After the initial clusters are formed, the number of clusters is reduced by repeatedly joining the two 'nearest' clusters.

The distance $d(\mathcal{X}, \mathcal{Y})$ between two clusters $\mathcal{X}$ and $\mathcal{Y}$ is derived from a likelihood ratio test. Let H_0 denote the hypothesis that the data in $\mathcal{X}$ and $\mathcal{Y}$ were generated by a single speaker and let H_1 denote the hypothesis that the data were produced by two distinct speakers. Let $\mathcal{X} = v_1, ..., v_r$ denote the cepstral vectors in one cluster, $\mathcal{Y} = v_{r+1}, ..., v_n$ denote the vectors in the other, and $\mathcal{Z} = v_1, ..., v_n$ denote the combined collection of vectors. The vectors are assumed to be *i.i.d.* and are not necessarily time adjacent. Let $L(\mathcal{X} : \theta_x)$ be the likelihood of the $\mathcal{X}$ cluster, where the likelihood is based on a Gaussian distribution. Here θ_x denotes maximum likelihood estimates for the mean and covariance matrix based on samples in that cluster. Let $L(\mathcal{Y} : \theta_y)$ and $L(\mathcal{Z} : \theta_z)$ be similarly defined. The likelihood L_1 that the two segments were generated by different speakers is $L_1 = L(\mathcal{X} : \theta_x)L(\mathcal{Y} : \theta_y)$. The likelihood L_0 that the segments were generated by the same speaker is $L_0 = L(\mathcal{Z} : \theta_z)$. Thus the likelihood ratio $\lambda_L = L_0/L_1$ is given by

$$\lambda_L = \frac{L(\mathcal{Z} : \theta_z)}{L(\mathcal{X} : \theta_x)L(\mathcal{Y} : \theta_y)}. \qquad (1)$$

The distance measure used in the hierarchical clustering is then taken as $d_L(\mathcal{X}, \mathcal{Y}) = -\log(\lambda_L)$. Of course $d_L(\cdot, \cdot)$ is not a true distance, but a measure of dissimilarity.

Our definition of distance follows Gish *et al* [6], but differs in that equation (1) is used to define distance between any two clusters, whether they contain single or multiple segments. By contrast, the earlier work used equation (1) to define distance between all pairs of segments, and then used standard hierarchical clustering. The standard hierarchical clustering algorithms, for example "hclust" in the *S Interactive Environment for Data Analysis and Graphics* [14], compute the distance between clusters as the maximum, minimum or average of the pairwise distances between segments comprising the clusters. Thus the distance between clusters $\mathcal{X}$ and $\mathcal{Y}$ using the maximum pairwise distance is

$$d_M(\mathcal{X}, \mathcal{Y}) = \max_{x \in \mathcal{X}, y \in \mathcal{Y}} d_L(\{x\}, \{y\}). \qquad (2)$$

Using equation (1) to define the distance between all clusters has the advantages that it allows a consistent statistical interpretation of distance throughout the clustering process, and that as the number of clusters becomes small, the cluster parameters are estimated using greater amounts of data.

3.2 Gaussian Mixtures

Rose and Reynolds [13] found that a Gaussian mixture model provided a more accurate method for speaker identification than a single Gaussian. Thus we extend the likelihood ratio of equation (1) to tied mixtures of Gaussians. Rather than computing the likelihood of a segment of speech assuming a single Gaussian, the likelihood is based on a mixture of K Gaussians. Let $N_k(v) = N(v : \mu_k, \Sigma_k)$ be the Gaussian distribution for a vector v associated with the k^{th} mixture component for each $k = 1, \ldots, K$. The means μ_k and covariance matrices Σ_k for components of the Gaussian mixture are estimated using the entire set of unsegmented data. These parameters are then fixed. Let $g_k(x)$ be the weight for the k^{th} mixture estimated using segment x. The likelihood of $x = v_1, ..., v_r$ is

$$L(x : \theta_x) = \prod_{j=1}^{r} \sum_{k=1}^{K} g_k(x) N_k(v_j). \qquad (3)$$

The likelihood $L(y : \theta_y)$ is computed similarly.

Since the means and covariance matrices of the Gaussian mixture are fixed, the only free parameters to be estimated from the x segment are the mixture weights. Thus $\theta_x = (g_1(x), \ldots, g_K(x))$. The weight $g_k(x)$ is estimated by the proportion of samples v in the x segment for which the probability of the k^{th} component, $N_k(v)$, is maximum. Thus the mixture weights $g_k(z)$ can be derived from the weights $g_k(x)$ and $g_k(y)$ as

$$g_k(z) = (\frac{r}{n})g_k(x) + (\frac{n-r}{n})g_k(y). \qquad (4)$$

The distance measure $d_L(x, y) = -\log(\lambda_L)$ can then be computed using the mixture model of equation (3) in equation (1).

3.3 Duration Bias

It is often the case that adjacent segments are from the same speaker. In order to take advantage of this information at the level of the hierarchical clustering, the likelihood ratio of equation (1) was biased using a simple duration model based on speaker changes over the original equal length segments. Let S_i denote the speaker during segment i, and M the number of speakers. Assume that S_i is a Markov chain with $\Pr[S_{i+1} = a | S_i = a] = p$ for each speaker a, and $\Pr[S_{i+1} = b | S_i = a] = (1-p)/(M-1)$ for each a and $b \neq a$. The probability $\Pr[S_{i+n} = S_i]$, that the speaker for segment i is also speaking for segment $i+n$, may be computed by using a two state Markov chain, where state 1 of the chain represents the speaker at time i, and state 2 represents all other speakers. (This reduction of the M state chain to a 2 state chain is only possible because of the complete symmetry.) The transition probability matrix P for this chain is

$$\mathrm{P} = \begin{pmatrix} p & 1-p \\ \frac{(1-p)}{M-1} & 1 - \frac{(1-p)}{M-1} \end{pmatrix}. \qquad (5)$$

In terms of this matrix, $\Pr[S_{i+n} = S_i] = (\mathrm{P}^n)_{11}$. By diagonalizing P this may be expressed in closed form as

$$f(n) \equiv \Pr[S_{i+n} = S_i] = \frac{1 + (M-1)(\frac{Mp-1}{M-1})^n}{M}. \qquad (6)$$

Using this equation we can compute the prior probabilities that two given clusters $\mathcal{X}$ and $\mathcal{Y}$ are produced by either the same speaker (hypothesis H_0) or by two different speakers (hypothesis H_1). Let $\mathcal{Z}$ be the cluster formed by merging $\mathcal{X}$ and $\mathcal{Y}$. There will be segments $z_j \in \mathcal{Z}$ such that $z_j \in \mathcal{X}$ and $z_{j+1} \in \mathcal{Y}$ (or vice versa), corresponding to intervals in which the beginning and ending speakers are different according to H_1. Let n_i be the difference between time indices of the first and last segments of the i^{th} such interval, and let C be the number of intervals. A duration bias is then defined as

$$\lambda_D = \frac{\Pr[H_0]}{\Pr[H_1]} = \frac{\prod_i^C f(n_i)}{(M-1)\prod_i^C (1 - f(n_i))/(M-1)}. \qquad (7)$$

The duration biased distance between clusters $\mathcal{X}$ and $\mathcal{Y}$, $d_D(\mathcal{X}, \mathcal{Y})$ is defined as $d_D(\mathcal{X}, \mathcal{Y}) = -\log(\lambda_L) - \log(\lambda_D)$.

4 User Interface

Figure 4 shows a graphical audio tool which may be used both for labeling audio recordings and for browsing previously segmented recordings. The tool runs on Sun workstations, and was implemented with the Tk interface in the Python language [8]. The upper panel of the tool contains the play control buttons. Beneath it is an overview timeline showing the full recording, and beneath that is a detailed timeline showing a limited time span within the recording. The detailed timeline contains "tracks" which correspond to the different acoustic classes. Each track is composed of colored bands indicating spans of time. The band in the overview timeline indicates the span of time displayed in the detailed timeline, and may be manipulated to control panning and zooming. The bands in the lower timeline indicate segments associated with speakers or acoustic classes. Each timeline also contains a vertical line which indicates current playback time.

Playback can be controlled in a number of ways. Clicking on a band causes the corresponding audio segment to be played. There are also buttons for skipping ahead (or back) a fixed amount of time, or to the next (previous) speaker. Also, a subset of the speakers can be selected (for example "chen" and "wilcox" in the figure), and "Skip/Play" used to play only portions corresponding to those speakers.

The tool also allows for labeling and editing of segments. Mouse actions allow bands to be created, deleted or adjusted. Adjusting may correspond to changing one or both endpoints, or assigning to a different track, which implies a relabeling of the segment. This is useful both for hand labeling initial training data and for making adjustments in the case of segmentation errors. The tool also allows tracks to be "collapsed" so that they are overlayed. In this case, only the color of the bands indicates speaker or class.

The tool is useful for assigning speaker labels to the classes produced by hierarchical clustering. In Figure 4, two such classes are shown. In practice, a number of clusters is chosen which is larger than the number of speakers (or desired label classes), and the tool is configured to show a track for each of the clusters, as well as for each speaker.[4] It is possible to listen to those segments of a given cluster, and move the segments to the appropriate track. (This is easy because there are operations for selecting all bands on a given track, and for moving all selected bands.) In some cases a cluster will be found to contain speech from two speakers. Because the agglomerative clustering algorithm is inherently hierarchical, such a cluster can be divided into the constituent clusters from which it was produced during the merging process. Through popup menus, the tool allows the track corresponding to this cluster to be split into two tracks corresponding to its constituents.

5 Experimental Results

5.1 Recorded Panel Discussion

Our initial set of tests were performed on a video-taped panel discussion from Siggraph [11]. There were five main speakers: a moderator and four panel members.

[4] Once a final segmentation has been produced, the tool is configured for an "end user" to show only the labeled classes.

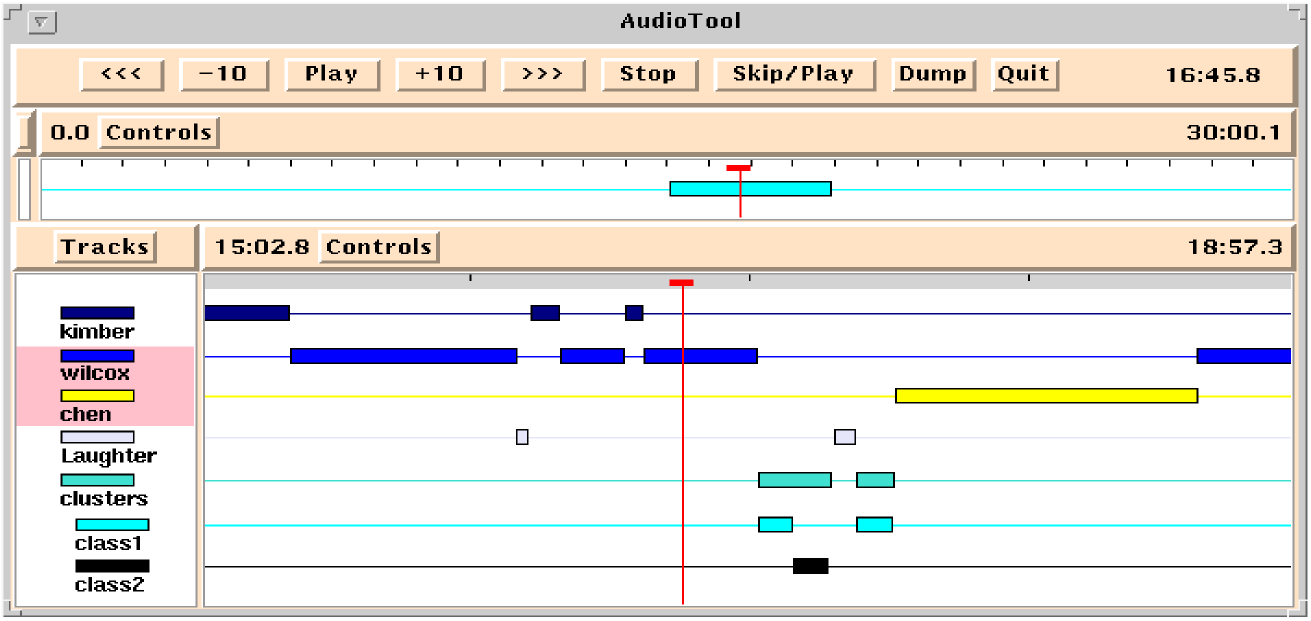

Figure 4: Audio Browsing Tool

In addition, there were speakers from the audience who asked questions of the panel members. The moderator and panel members each gave a short talk prior to the discussion. One minute from each speaker's short talk was used to train the speaker models for real time segmentation. The test data for the various experiments discussed below consisted of subsets of the first 18 minutes of the panel discussion. The recorded audio was hand labeled according to speaker. Silences longer than half a second were also labeled. The data was digitized using the Sun Sparc-10 audio device. The sampling rate was 8 kHz, with mu-law encoding and 8 bits per sample. Twelve cepstral coefficients were computed every 20 ms.

Real time segmentation was tested using the training data described above to train a model for each of the five speakers. Since the data contained speech from members of the audience, a garbage model was used to separate these speakers from the five members of the panel. Training data to initialize the garbage model was obtained by concatenating short portions of data from each of the five speakers. The speaker and garbage models were of the form shown in Figure 2 and each contained 32 states. A three state silence model was also used. Segmentation of the entire 18 minute test data resulted in 26 percent error, where error was the percent of the total time the wrong speaker was chosen. After iterative resegmentation, however, the error rate dropped to 6.1 percent. Thus the iterative segmentation algorithm provides a substantial improvement in segmentation accuracy.

Next we compared the schemes for initializing the speaker models for unsupervised speaker segmentation. To test random initialization of the speaker models, a 3 minute portion of the test data containing only two speakers was selected. No silence or garbage models were used. The error rates varied greatly depending on the randomizer. In five trials, the error rates were 3.4 percent, 1.5 percent, 48.8 percent, 1.5 percent and 9.0 percent. The error rate using supervised training for the initial estimates was .6 percent. Such sensitivity to initial estimates was also noted by Sugiyama *et.al.* [17].

To test unsupervised initialization using agglomerative clustering, a 6 minute segment of the panel discussion containing three speakers was selected. No silence or garbage models were used. Five second intervals of speech were used as the initial uniform segmentation. Table 1 shows the results for the distance d_L based on a single Gaussian model. When the distance between clusters was computed as the maximum pairwise distance between segments comprising the cluster, *i.e.* d_M was used, the segmentation error was initially 31.4 percent. The error after convergence of the iterative resegmentation algorithm was 30.3 percent. When the distance between clusters was recomputed, *i.e.* d_L was used, the error was initially 10.1 percent and dropped to 3.6 percent after iterative resegmentation. When the recomputed distance

Single Gaussian			
	Maximum	Recomputed	
	d_M	d_L	d_D
Initial	31.4%	10.1%	6.0%
Converged	30.3%	3.6%	.9%

Tied Gaussian Mixture			
	Maximum	Recomputed	
	d_M	d_L	d_D
Initial	10.7%	4.2%	4.2%
Converged	.8%	.5%	.5%

Table 1: Unsupervised segmentation error for panel discussion.

was biased by the duration model, the initial error was 6.0 percent, and .9 percent after convergence. For comparison, the segmentation error using supervised data to initialize the iterative algorithm was .5 percent.

Results with the distance d_M using the tied mixture model with 32 components are also given in Table 1. The initial error rate using the maximum pairwise distance was 10.7 percent. After iterative resegmentation, the error rate was .8 percent. When the distance between clusters was recomputed, the error was initially 4.2 percent and dropped to .5 percent after iterative resegmentation. Adding the duration bias did not further improve results.

5.2 Recorded Meetings

Our recent work has been with a series of recorded meetings having to do with the intellectual property assessment process at Xerox PARC. The use of meeting recording and reaccess as part of this process was reported by Minneman *et al* [10]. The audio for these meetings was obtained from the mixture of two microphones placed on the meeting table, and digitized using a Sun Sparc-10 as before. We report here on experiments with the initial hour of one such meeting involving 7 people. The audio was hand labeled by speaker; laughter, simultaneous talkers, and noise were all labeled as "junk". Segmentation error was computed as the percentage of the time that the estimated speaker did not agree with the true speaker. Those times when the true label was junk or silence were ignored in these evaluations.

We tried several variations on agglomerative clustering to obtain initial training data for the speakers. Preliminary experiments showed that equal length segments of 3 seconds each were best for initializing the agglomerative clustering. We also tried removing low energy segments from the initial set, since these contain mostly silence and thus provide little speaker information. (In this case, however, silence must be detected separately.)

Single Gaussian		
	All	No Sil.
Initial	30.5%	30.9%
Converged	22.4%	13.7%

Tied Gaussian Mixture		
	All	No Sil.
Initial	23.8%	29.6%
Converged	13.7%	11.9%

Table 2: Unsupervised Segmentation Error for Meeting.

After clustering, training data for each speaker was obtained by comparing the resulting clusters with the hand labeled data. A single cluster was associated with the speaker provided that: 1) the cluster contained a higher percentage of data from that speaker than from any other speaker, and 2) the cluster contained the highest percentage of the speaker among all clusters. In practice, this labeling would be done by a user listening to the clusters. If necessary the number of clusters used would be increased until a cluster satisfying these criteria was found for each speaker. Table 2 displays the results for both tied Gaussian and single Gaussian distance measures using all of the initial 3 second intervals, and using only the non-silence intervals. The row labeled "Initial" is the segmentation error for the initial training data. This error is only for a subset of the meeting, and would be zero for hand-labeled initialization. The column labeled "Iterative" is the segmentation error for the entire meeting after three iterations of the iterative resegmentation algorithm.

For the tied Gaussian mixture distance using all of the three second intervals, the error in the initialization data produced by clustering was 23.8 percent, and resulted in a 13.7 percent error after resegmentation. In comparison, the initial error in the case of clustering using non-silence intervals only was 29.6 percent, and resulted in 11.9 percent error after resegmentation. Thus silence deletion resulted in a better final segmentation. Similar results were obtained for the single Gaussian distance. For comparison, the error using one minute of hand labeled data for initialization is 14.3 percent.

5.3 Recorded Media

In order to investigate the ability to segment non-speech sounds, we tested the system on the audio portion of a recorded television news show, and on an archival vitaphone movie. While no statistical analysis of the segmentation performance for this media is available, we will provide an empirical discussion of the results.

The first media experiment was made on a recording of the MacNeil-Lehrer News Hour. In this program, a musical theme is used to separate different news segments. The ability to index according to these segments allows the user to easily skip portions of the news. The musical theme (5 seconds) was used to train one model, and an introduction by the newscasters (20 seconds) was used to train another. The News Hour was then segmented according to these models. The musical theme was correctly classified in all cases. However, a 10 second segment of the regular news was also classified as the musical theme. This portion was actually a location piece, in which a band was playing.

The second media experiment involved the 18 minute vitaphone movie, "Old Lace". No prior information was provided for this movie. Rather, it was required that the audio segmentation provide a structure with which to browse the video. This is analogous to the use of video segmentation algorithms which partition video according to scene changes [16] to create indices for browsing and retrieval. Audio segmentation could also provide an additional annotation stream for the Marquee video annotation system [18].

The audio was digitized using a Sun Sparc-10 as previously described. Agglomerative clustering was performed using uniform 3 second intervals for initialization. Unlike the meeting or panel discussion data, where the number of speakers was known and could be used to select a number of clusters, here the number of clusters needed to be inferred from the data. This was done based on the merge distance at each step in the agglomerative clustering.

The audio browsing tool was then used to listen to each of the clusters. The clusters were manually labeled as follows: Music, for instrumental music, Females, for female speech, Males, for male speech, SongF1, for a song by one of the females, SongF2, for a another song by another female, SongM, for the song by a group of male singers (actually a barbershop quartet), Applause, Bells, and TalkN, for talk in noise, actually a conversation occurring in a traveling automobile. The iterative resegmentation algorithm was then used to improve this segmentation. The results are shown in Figure 5. The audio structure of the movie is thus apparent from the segmentation. For example, the beginning portion of the timeline in Figure 5 shows a song by a female, followed by applause, followed by what appears to be a conversation between males and females. This is in fact the sequence of events for this portion of the movie.

6 Summary and Conclusions

We've discussed a method of audio index generation based on segmentation of the audio into different acoustic classes, and have described an audio browsing tool which uses these indices to facilitate flexible modes of navigation and listening. As an increasing amount of audio is available in online digital form, we anticipate a greatly increased demand for audio indexing and the flexibility in listening which it affords.

The accuracy of the segmentation produced by our method varies considerably for different types of recording. For a recorded meeting with 7 people, the segmentation error was about 14%. For the recorded panel discussion, the error was under 1 percent. However, for the panel each speaker was individually miked, while for the meeting only two microphones on the meeting table were used to capture all 7 speakers. Further, the panel discussion was a formal situation, in which each speaker spoke in turn, with an average utterance length of 20 seconds. In contrast, the speakers in the meeting were fairly informal. One third of the utterances were interrupted, and the average utterance length was less than 3 seconds. Thus the increased number of speaker changes and short speaker durations, as well as poorer miking, explain the decreased accuracy in the audio segmentation of the meeting.

The human effort involved in producing the audio segmentations arises from the need for labeled data to initialize acoustic class models. This effort can be greatly reduced by using an unsupervised clustering algorithm to determine a small number of acoustic classes, and then using an audio tool, such as the one we've described, to associate labels with those classes. We've consistently found, for a variety of types of recordings, that the segmentations produced by this method of initialization compare favorably with those produced using careful hand labeling for initialization.

7 Acknowledgments

Portions of this work were done in collaboration with Vijay Balasubramanian, Francine Chen, Phil Chou and Surya Ganguli. We would also like to thank Tom Moran, Polle Zellweger, and other members of the collaborative systems area at Xerox PARC for valuable conversations.

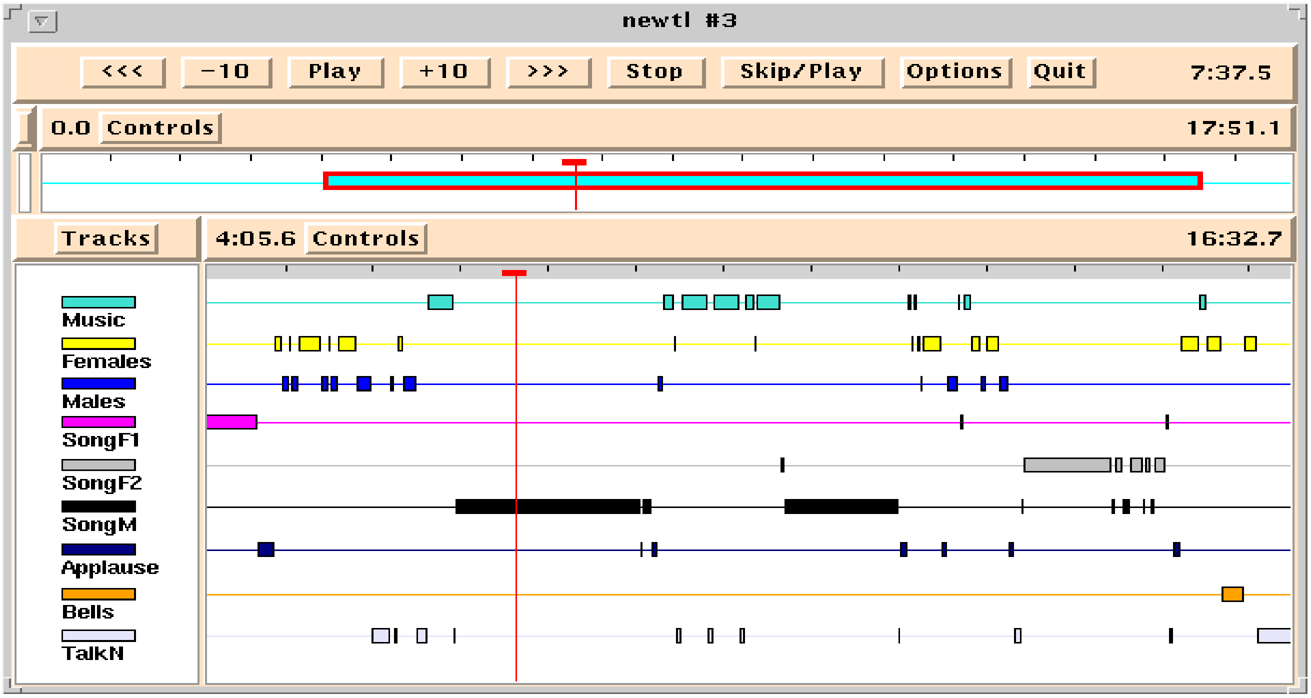

Figure 5: Segmented Movie.

References

[1] B. Arons, “Techniques, perception, and application of time-compressed speech,” *Proc. Conf. American Voice I/O Society*, pp. 169-177, September 1992.

[2] B. Arons, “SpeechSkimmer: Interactively skimming recorded speech,” *Proc. UIST: ACM Symposium on User Interface and Speech Technology*, pp. 187-196, November 1993.

[3] P. F. Brown, J.C. Spohrer, P.H. Hochschild, J.K. Baker, “Partial Traceback and Dynamic Programming,” *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Paris, France, pp. 1629-1632, May 1982.

[4] F.R. Chen, and M.M. Withgott, “The use of emphasis to automatically summarize a spoken discourse,” *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, San Fransisico, CA, pp. 229-232, March 1992.

[5] J.L. Gauvain and L.F. Lamel, “Identification of Non-Linguistic Speech Features,” *Proc. ARPA Human Language Technology Workshop*, March 1993.

[6] H. Gish, M.H. Siu, and R. Rohlicek. “Segregation of Speakers for Speech Recognition and Speaker Identification,” *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Toronto, Canada, vol.2, pp. 873-876, May 1991.

[7] A. Hauptmann, and M. Smith, “Text, Speech and Vision for Video Segmentation: The Informedia Project,” Proc. AAAI Fall 1995 Symposium on Computational Models for Integrating Language and Vision, 1995.

[8] D.G. Kimber, L.D. Wilcox, F.R. Chen and T.P. Moran, “Speaker Segmentation for Browsing Recorded Audio,” *Proc. CHI: Human Factors in Computing Systems, Conference Companion*, Denver, CO, pp. 212-213, May 1995.

[9] T. Matsui, and S. Furui, “Comparison of Text-Independent Speaker Recognition Methods Using VQ-Distortion and Discrete/Continuous HMMs,” *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, San Fransisco, CA, vol. 2, pp. 157-160, March 1992.

[10] S. Minneman, S. Harrison, B. Janssen, G. Kurtenbach, T.P. Moran, I. Smith and W. van Melle, “A Confederation of Tools for Capturing and Accessing Collaborative Activities,” *Proc. of Multimedia '95 Conference*, November 1995.

[11] “Where Do User Interfaces Come From”. Panel Discussion from Siggraph, 1987.

[12] L.R. Rabiner and B. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, 1993.

[13] R.C. Rose and D.A. Reynolds, “Text Independent Speaker Identification Using Automatic Acoustic Segmentation,” *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Albaquerque, NM, pp. 293-296, April 1990.

[14] R.A. Becker and J.M. Chambers, *S: An Interactive Environment for Data Analysis and Graphics,* Wadsworth Advanced Book Program, Belmont, CA., 1984.

[15] M.-H. Siu, G. Yu, and H. Gish, "An Unsupervised, Sequential Learning Algorithm for the Segmentation of Speech Waveforms with Multiple Speakers," *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, San Fransisco, CA, vol. 2, pp. 189-192, March 1992.

[16] S.W. Smoliar and H.J. Zhang, "Content-Based Video Indexing and Retrieval," *IEEE Multimedia*, Vol. 1, no. 2, Summer, 1994.

[17] M. Sugiyama, J. Murakami and H. Watanabe, "Speech Segmentation and Clustering Based on Speaker Features," *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Minneapolis, MN, vol. 2, pp. 395-398, April 1993.

[18] K. Weber and A. Poon, "Marquee: A Tool for real-time video logging," *Proc. CHI'94*, 58-64, 1994.

[19] L.D. Wilcox and M.A. Bush, "Training and Search Algorithms for an Interactive Wordspotting System," *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, San Fransisco, CA, vol. 2, pp. 97-100, March 1992.

[20] L.D. Wilcox, F.R. Chen, D. Kimber, and V. Balasubramanian, "Segmentation of Speech using Speaker Identification," *Proc. International Conference on Acoustics, Speech and Signal Processing*, Adelaide, Australia, vol. 1, pp. 161-164, April 1994.