Speaker Indexing Using Neural Network Clustering of Vowel Spectra

Deb K. Roy
MIT Media Lab
20 Ames St., Cambridge, MA 02139
dkroy@media.mit.edu

Abstract

Speaker indexing refers to the process of separating speakers within a recording and assigning indices to each unique speaker. This paper describes a new speaker indexing algorithm which dynamically generates and trains a neural network to model each postulated speaker found within a recording. Each neural network is trained to differentiate the vowel spectra of one specific speaker from all other speakers. A method for combining speaker indexing and other annotations of a recording in a general framework is also presented. The speaker indexing system is currently being incorporated into several application systems in the Speech Group at the MIT Media Lab.

1. Introduction

The Speech group at the MIT Media Lab is exploring methods for accessing large amounts of recorded speech efficiently (Arons, 1994; Mullins, 1995; Schmandt, 1994). One approach we are taking is to tag salient segments of a speech recording, and then design interfaces to navigate through the speech using those tags (Arons, 1994; Mullins, 1995). Early versions of these systems relied primarily on pause and pitch information to locate salient segments of audio. For example, SpeechSkimmer plays short segments of a speech recording which directly follow long pauses as a way of skimming the entire contents of the recording (Arons, 1994). This skimming method assumes that a salient event such as a change in topic, a point of emphasis, or a change in speaker usually follows a long pause. SpeechSkimmer also uses pitch analysis to locate other salient segments.

This report describes a new algorithm which performs speaker indexing, a type of annotation of speech recordings which can be used by interfaces such as SpeechSkimmer. The term speaker indexing (SI) refers to the process of separating speakers within a recording and assigning labels, or indices, to each unique speaker. For example consider a recording which contains the voices of four people as shown in Figure 1. The top strip represents the sequence of speakers in the recording (time flows from left to right). In this example, Speaker A begins talking, followed by Speaker B, then Speaker C, then back

to Speaker A and so on. Changes in speakers are indicated by vertical bars in the top strip. Given the audio recording as input, the ideal output of the SI system is shown in the lower strip. Each speaker change boundary is located, and indices are assigned to each segment which are consistent with the original identities of the speakers. Since the SI system has no prior models of the speakers, it does not identify the speakers, but rather separates them from each other within the recording.

An important distinction between the SI problem and conventional speaking identification is that there is no assumed prior knowledge about the speakers in the input recording. In speaker identification, a set of models of all possible speakers is created using training samples of each speaker. Identification of an unknown sample is performed by comparing the speech sample to each speaker model and finding the closest match. For the class of applications we are interested, we could not assume the a priori availability of training data for speakers. Thus conventional speaker identification techniques cannot be directly applied.

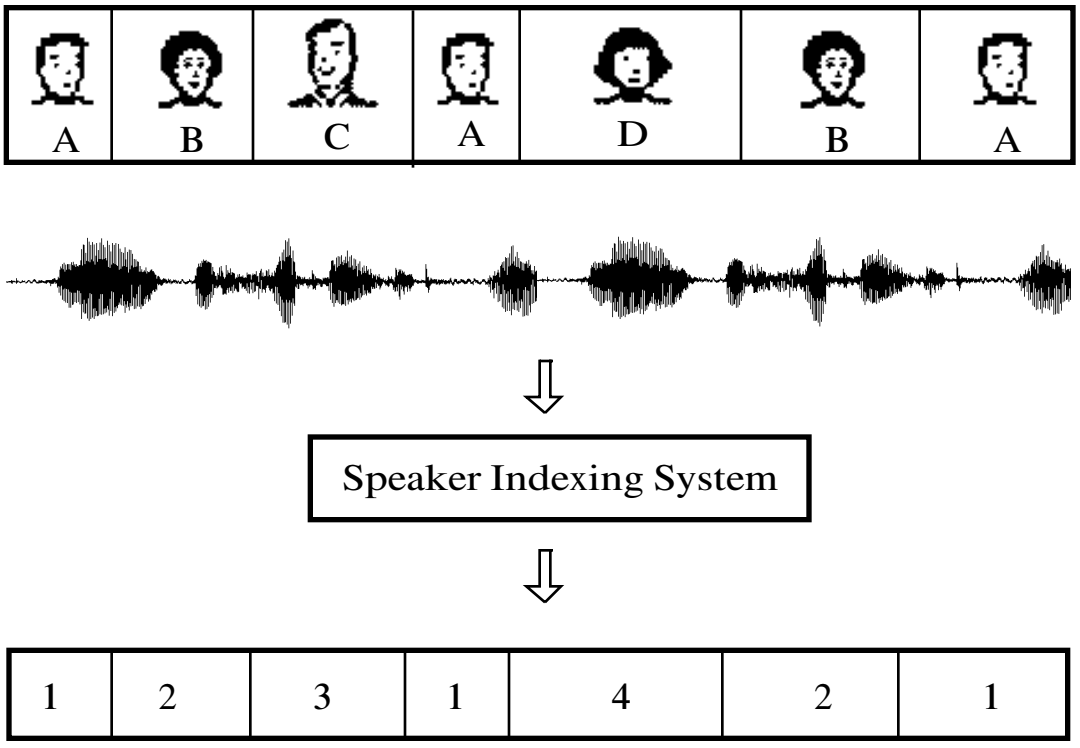

Figure 1 Ideal Output of the SI System: The top strip represents the sequence of four speakers in a recording (time flows from left to right). The audio recording (shown as a speech wave) is processed by the SI system which outputs a sequence of indexed segments. Ideally each segment output from the SI system corresponds to a speaker turn in the input recording, and the indices assigned to each segment correspond to a actual speaker identity (in this example Index 1 corresponds to Speaker A, Index 2 to Speaker B, 3 to C, and 4 to D). A simple application of this system would be to play short audio segments directly following each speaker turn to get a summary of the recording.

2. Related Work

This section reviews several research systems which have addressed issues related to this paper.

Arons (1994) designed a hand-held interface for interactively skimming recorded speech called SpeechSkimmer. The interface enables the user to listen to a speech recording at four levels of skimming. At the lowest level the entire recording is heard. At the second level pauses are shortened. At the third level, only short segments (highlights) of the recording following long pauses are played; the portions of the recording between these highlights are skipped. In level four highlights selected by finding speech containing maximum variations in pitch are played. The speaker indexing algorithm presented in this paper is now providing a new level of skimming in SpeechSkimmer by locating highlights from speech recordings which follow speaker changes.

Gish, et. al. (1991) have developed a method for segregating speakers engaged in dialog. Their method assumes no prior knowledge of the speakers. A distance measure based on likelihood ratios is developed which is used to measure the distance between two segments of speech. Agglomerative clustering based on this distance measure is used to cluster a long recording by speaker. The method has been successfully applied to an air traffic control environment where the task is to separate the controller's speech from all pilots. Since the controller speaks more often than any of the pilots, the largest cluster is labeled as the controller.

Wilcox, et. al. (1994) also uses a likelihood ratio based agglomerative clustering algorithm to index speakers. Additionally, they use a hidden markov model to model speaker transition probabilities.

Chen and Withgott (1992) describe a method for summarizing speech recordings by locating and extracting emphasized portions of the recording. Hidden markov models (HMMs) are used to model emphasis regions. The energy, delta energy, pitch and delta pitch parameters are extracted from the speech and used as parametric input to the HMM. Training data was collected by manually annotating the emphasized portions of several speech recordings.

Hawley (1993) designed a set of audio processing tools called sound sensors which extract structural information from audio recordings. Hawley implemented three sensors: a polyphonic pitch extractor, a music detector, and a pitch based speaker recognizer. The output of these sensors are combined and encoded in an ASCII text file which can be used by an application to access the contents of the recording.

3. Initial Task

The initial task set for the SI system is to index speakers in BBC newscasts. The newscasts are 20 minutes long and contain between 12 and 20 unique speakers each. The broadcasts are digitized from a local FM radio station (which rebroadcasts the original newscast from England) using an 8-bit mu-law 8kHz analog to digital converter on a Sparc workstation. Each broadcast is hosted by two speakers, and the remaining speakers are foreign correspondents, special reports, and interviews. The background noise level varies widely, from very clean signals for the studio recordings of the hosts, to highly degraded signals of some field reports.

The assumptions afforded in the BBC indexing task are:

(1) The minimum speaker turn is 5 seconds

(2) The minimum pause between speaker turns is 0.2 seconds

(3) The entire audio recording is available before processing begins

Assumption (1) was found to be true through empirical analysis of several BBC news broadcasts; no speaker talks for less than 5 seconds except when an interview is conducted within the news program (in which cases the system is expected to miss segments). Also through empirical measurements, Assumption (2) was found to be valid for BBC news except during interviews; there is generally a clean break between speakers. Assumption (3) can be made for our applications since the results of the SI algorithm are only used to access previously-recorded audio (the SI algorithm will not produce indices in real time).

4. The Speaker Indexing Algorithm

The speaker indexing algorithm dynamically generates and trains a neural net to model each postulated speaker found in the recording. Each trained neural net takes a single vowel spectrum as input, and outputs a binary decision indicating whether the vowel belongs to the speaker or not.

4.1 Signal Processing

Figure 2 shows the signal processing front end which extracts mel-scaled vowel spectra, and locates pauses in the speech recording. The speech input is sampled at 8000 samples per second using a 8-bit mu-law encoded digital-to-analog converter. On the far left, the adaptive speech and silence detector

computes the speech/silence energy threshold of the recording by generating a histogram of the energy distribution over the entire recording, and tagging the low 20% of the distribution as silence. The energy of the input signal is computed over a 64ms frame, overlapped 32ms. A pause detector locates contiguous frames of silence which last longer than 0.2 seconds (this is used to train the neural nets, as explained below). Each set of vowel spectra delimited by such pauses will be referred to as a "sentence" in the remainder of this paper. Note that based on assumption 2 from Section 2, we can infer that each sentence must be spoken by only one speaker.

On the right hand side of Figure 2, a fast Fourier transform (FFT) of the input signal is computed using a 64ms Hamming window with 32ms overlap. The resultant spectrum is passed through a mel-scaled filter bank which produces a 19 coefficient spectral vector. In the time domain, a peak picker estimates the location of vowels by picking peaks of the energy of the speech signal (vowels have relatively high airflow and thus a corresponding peak in the energy contour). The "logical and" of the outputs of the peak picker and the speech/silence detector is computed in order to eliminate false vowel detection by the peak picker during background noise.

Only the mel-scaled spectra corresponding to each vowel is output to the neural network portion of the system. This is depicted by the sample mel-scaled spectrogram in the figure which represents several seconds of speech. Four frames have been identified by the peak picker as vowels and are output to the neural network portion of the system. Non-vowel information is discarded in order to reduce the size of the neural networks.

Although most vowels in the recording will occupy more than a single 64ms frame, the current implementation only selects the single frame corresponding to the center of the energy peak.

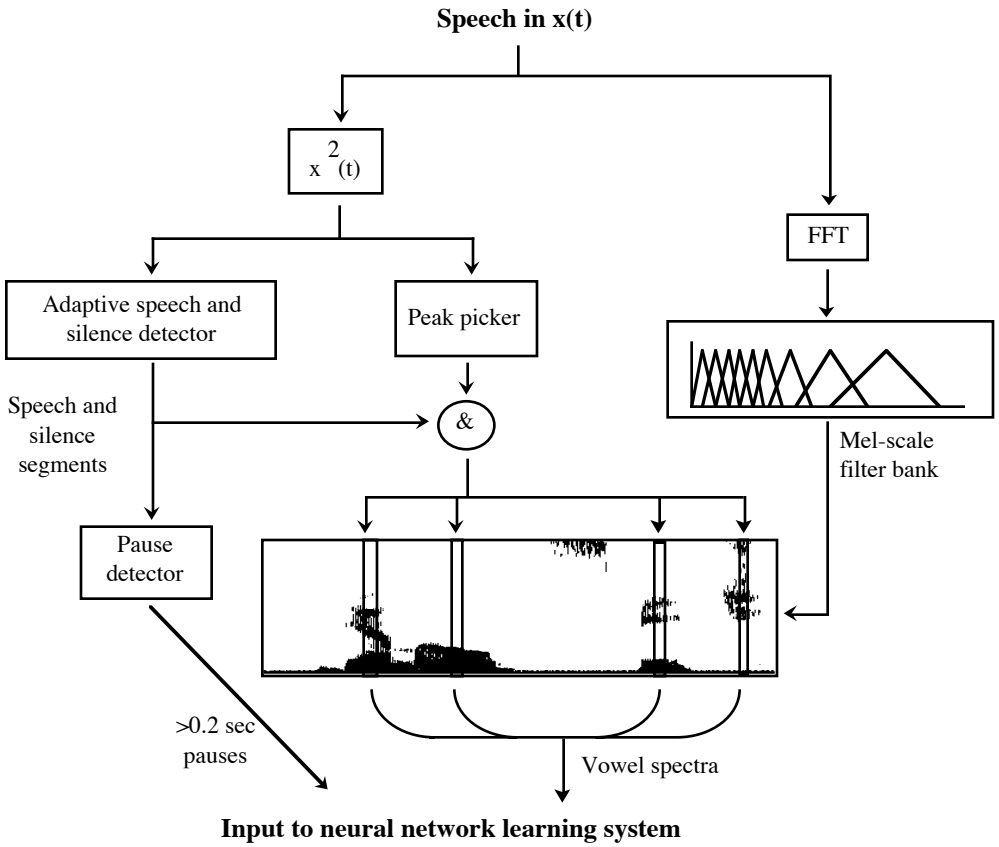

Figure 2: Signal Processor extracts mel-scaled spectra of vowels, and locates pauses longer than 0.2 seconds.

4.2 Training the Neural Networks

The SI system employs back propagation neural networks to model each postulated speaker in the input recording. Back propagation neural networks are trained through a supervised process (Rumelhart, 1986). For a network with binary output, a set of positive and negative training examples are required. The examples are presented in sequence to the network. The weights of the network are adjusted by back-propagating the difference between the network's output and the expected output for each training example in order to minimize the error over the entire training set.

If the positive training examples are a subset of the vowels spoken by some Speaker X, and the negative examples are a subset of the vowels spoken by all the other speakers, we can expect the trained network to differentiate vowels generated by Speaker X from all other speakers (including vowels that were not in the training set).

However, since there is no a priori knowledge of the speakers, training data must be selected automatically. This selection process begins by assuming that the first 5 seconds of the recording was

spoken by a single speaker, Speaker 1. The spectra of the vowels from this 5 second segment comprise the positive training data for the first neural net. A random sampling of 25% of the remainder of the recording is used as negative training data. Note that the negative training set selected in this manner will probably contain some vowels which belong to Speaker 1, leading to a sub-optimal speaker model.

Once the neural network has been trained using this training set, the network is used to classify every vowel in the recording as either belonging to Speaker 1 or not (true or false). The resultant sequence of classification tags is then filtered to eliminate tags which do not conform to Assumption 2 (Section 3). This is accomplished by applying a "majority rules" heuristic; for each sentence in the recording, if the majority of tags belong to Speaker 1, then all of the vowels in that sentence are tagged as true. On the other hand, if the majority are classified as false, then all tags for that sentence are set to false. This filtering process has two effects: (1) possible false-positive tags generated by the neural network are removed, and (2) vowels which were not recognized as Speaker 1 are "picked up" in cases where the majority (but not all) of the vowels in a sentence were positively tagged. This filtering process partially compensates for errors in the training set. A second filter is then applied which enforces Assumption 1: any sequence of tags which is shorter than the minimum speaker turn is inverted.

Once the two levels of filters have been applied, the neural network is re-trained. All of the vowels which have been classified as Speaker 1 (after filtering) are collected and constitute the new positive training set, and again 25% of the remaining vowels (randomly selected) constitute the negative training set. This entire training, tagging, and filtering cycle is repeated until no further positive training vowels are found.

Once the first speaker is located using the above method, the audio corresponding to that speaker is removed from the input recording, and a new neural network (for Speaker 2) is created and trained on the remaining audio using the exact same procedure. This cycle is repeated until all audio in the input recording has been indexed.

5. Results

Initial tests have been performed on a set of ten 20-minute BBC newscasts recorded over a period of two weeks in the summer of 1994. Each speaker change location and index was hand labeled. A set of utilities were written to make the following accuracy measurements:

Speaker indexing: This measures the number of frames of the recording that were indexed correctly as a percentage of the total number of frames. The SI system currently indexes with an accuracy of 64%.

Speaker changes detected: The percentage of speaker changes which are detected (ignoring the index assigned to the speakers). A speaker change must be within one second of the hand marked location to be considered correct. The SI algorithm currently detects 50% of speaker changes.

False Alarm Speaker Changes: The percentage of detected speaker changes which do not correspond to a speaker change in the actual audio. For the speaker change detection accuracy of 50% stated above, the false alarm rate is 57%.

The two speaker change measures are useful since we expect some applications to use only the location of speaker changes (and discard the index assignments). Although the error rates are presently quite high, the speaker change annotations have been successfully used in initial test applications at the Media Lab.

6. A General Framework for Combining SI Annotations with Other Types of Annotations

The goal of a structured representation is to have "handles" into a large media stream. If placed in meaningful or salient locations, these handles can be used to browse and search the stream. We now present a structured representation to enable skimming and searching speech recordings at any arbitrary level of granularity. The basic function of the framework is to locate the next place to jump to within the recording from the current position. The jump locations can be used by applications to enable efficient access to the contents of the speech recording. For example, a recording can be skimmed by playing short segments following each jump. Similarly, a recording can be summarized automatically by extracting and concatenating speech segments following each jump location.

The first concept which needs to be defined is the salience of a sample. The salience of the i^{th} annotated sample of the recording, $S[i]$, is defined as:

$$S[i] = \sum_{j=0}^{n-1} w_j \cdot A_j[i] \qquad \textbf{Equation 1}$$

where there are n types of annotation, w_j is the weight of the j^{th} annotation type, and $A_j[i]$ is the value of the j^{th} annotation for sample i of the recording.

In the present system there are two types of annotations: pauses, and speaker changes. For this system Equation 1 may be rewritten as:

$$s[i] = (w_{sc} \cdot sc[i]) + (w_{pause} \cdot pause[i]) \qquad \textbf{Equation 2}$$

where *sc[i]* and *pause[i]* are the values of the annotations for sample *i* of the recording, and w_{SC} and w_{pause} are the corresponding weights for each annotation. The value of *sc[i]* is binary: 1 if a speaker change has been detected for the ith frame, 0 otherwise. For consistency, *pause[i]* is a value between 0 and 1. This is achieved by scaling all pauses in the recording to the unit range. Samples which are tagged as pauses have *pause[i]* set to zero.

The weights w_{sc} and w_{pause} may be assigned any value greater than or equal to zero. The weights for each annotation type is chosen to reflect both the importance and reliability of the annotation. Pauses would be weighted less than speaker changes in this system to reflect the greater salience of speaker changes versus pauses. The salience is also proportional to the length of the pause to reflect the assumption that longer pauses precede more salient events in speech streams.

Figure 3 illustrates a sample set of annotations for a speech recording that might be produced by the SI system. Time flows horizontally from left to right. Two types of annotations are shown at the top of the drawing: speaker changes, and pauses. The length of the pause marks are proportional to the length of the associated pause in the recording. Below these two layers is an overlay of both pauses and speaker changes.

The two gray bars indicate “jump ranges” from the current position in the recording; they show the maximum distance of the jump from the current position. The location with the highest salience (defined in Equation 2) within the jump range is selected as the destination of jump.

As figure 3 shows, the granularity of the jumps is proportional to the jump range. By using a small jump range, the average jump size will be shorter, and thus more locations within the file will be selected. For each of the jump ranges, an arrow marks the destination of the jump: for the fine granularity jump range, there are no speaker changes present, so the longest pause is selected. For the course granularity jump range, there are two speaker changes, so the one with the longer pause is selected (by applying equation 2).

Note that the framework is extendable; it can combine an arbitrary number of annotation layers, and it can be used with any type of media stream including audio, video, and text streams.

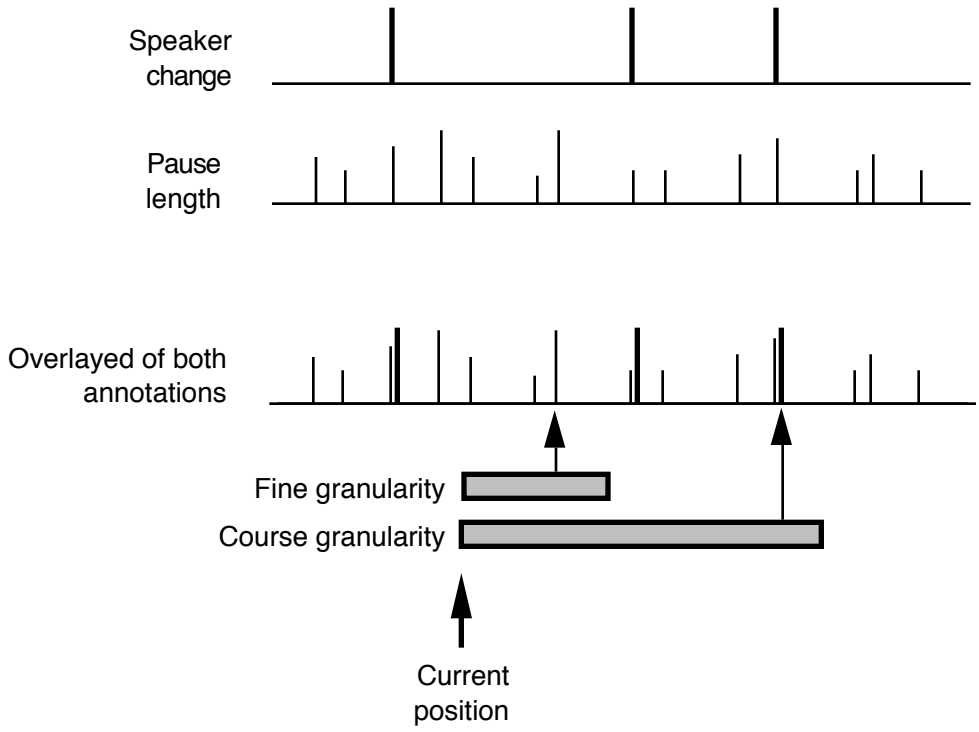

Figure 3: A sample annotation of a speech recording.

7. Current Applications

The SI algorithm is being run daily in the Speech Research Group on the day's most current BBC news broadcast. The results of the algorithm are converted to compatible formats for use with two audio browsing systems, SpeechSkimmer (Arons, 1994) and AudioStreamer (Mullins, 1995).

In an separate experiment, the Interactive Cinema Group at the Media Lab successfully segmented a video recording of a three-way interview using the SI system, allowing access to the interview by speaker.

The author is currently implementing a hand-held audio device called NewsComm which will use the SI system to preprocess speech recordings. The device will be a portable device for delivering personalized audio news. The interface will enable interactive navigation of the audio content, based on underlying structural annotations made by the SI system and human editors. The framework presented in Section 6 will be used to combine the output of the SI system and editors' annotations.

8. Future Work

We are currently analyzing errors from the system to identify causes for splitting and combining speakers. We believe that a main source of errors is the boot strapping process for selecting initial neural net training data. Randomly selected negative training data may cause a neural net to split speakers if the negative data contains too many samples of the speaker being segmented. We plan to replace the random selection process with an agglomerative clustering method which has been successfully used in other systems (Wilcox, et al., 1994; Gish, et el., 1991).

We are also looking at ways to make the system more robust for running on non-BBC audio by:

- Reducing the minimum speaker turn duration
- Reducing the minimum pause required between speakers

Acknowledgments

The author would like to thank Chris Schmandt for useful discussions about the SI system, and Sumit Basu for writing all the software for testing the accuarcy of the indexing system.

References

Arons, B. (1994). Speech Skimmer: Interactively Skimming Recorded Speech. Ph.D. thesis, MIT Media Lab.

Chen, F., Withgott, M. (1992). “The Use of Emphasis to Automatically Summarize a Spoken Discourse”. Proc. Int. Conf. Acoustics, Speech and Signal Processing. Vol. 1 (pp. 229-232).

Gish, H., Siu, M., Rohlicek, R. (1991). “Segregation of Speakers for Speech Recognition and Speaker Identification”. Proc. Int. Conf. Acoustics, Speech and Signal Processing. Vol. 2 (pp. 873-876).

Hawley, M. (1993). Structure out of Sound. Ph.D. thesis, MIT Media Lab.

Mullins, A. (1995). Audio Streamer (Media Lab Masters thesis, in progress).

Rumelhart D., Hinton, G., and Williams, R. (1986). “Learning representations by back-propagating errors,” Nature, Vol. 323 (pp. 533-536).

Schmandt, C. (1994). Voice Communication with Computers. New York: Van Nostrand Reinhold.

Wilcox, L., Bush, A. (1991). “HMM-Based Wordspotting for Voice Editing and Indexing”. Proc. Eurospeech (pp. 25-28).

Wilcox, L., Kimber, D., Chen, F. (1994). “Audio Indexing using Speaker Identification”. Xerox PARC ISTL Technical Report No. ISTL-QCA-1994-05-04.