TranSegId: A System for Concurrent Speech Transcription, Speaker Segmentation and Speaker Identification

Mahesh Viswanathan, Homayoon S.M. Beigi, Alain Tritschler
IBM Thomas J. Watson Research Labs Research Center
Yorktown Heights, New York 10598, USA
maheshv,beigi,tritschl@watson.ibm.com

Fereydoun Maali
Signal Recognition Corporation
Box 7010, New York, New York 10128, USA

Abstract

A system to analyze live audio feeds using speech and speaker recognition is described. Multi-modal analysis of this nature calls for concurrent operations to transcribe speech, to partition it into acoustically homogeneous, single-speaker segments, and to identify the speaker in these segments. The architecture of such a system built at IBM is described along with the techniques used in the individual modules involved. The output of this system can be used in searching through audio and video clips, tracking speakers in multiple broadcast feeds, and in other cataloging tasks.

1. Introduction

With the approaching maturity of speech and speaker recognition systems combined with the delivery of faster and faster processors to the marketplace, we now have the tools to build complex systems for audio analysis. The architecture of such a PC-based system for real-time, automatic, multi-modal analysis of live broadcast audio called *TranSegId* (or **Tran**scription, **Seg**mentation, and **Id**entification) is the focus of this article. By integrating speech and speaker recognition, the system serves as an ingest process for cataloging video and audio.

The components of *TranSegId* include speech recognition, speaker segmentation, and speaker identification and verification. A live audio stream from a VCR or equivalent audio source is the input to this system. The system uses a single common front-end signal processing module which converts the input audio into feature vectors that are simultaneously delivered to the three components in a multi-process and multi-threaded programming environment. These three components are all programmable via application programming interfaces called SMAPI (www.software.ibm.com/voicetype), SEGAPI, and SVAPI (www.srapi.com/svapi) respectively.

Figure 1 provides an overview of the architecture used in *TranSegId*. A large vocabulary continuous speech recognition system is used to produce time-aligned transcripts of the speech. The segmentation module divides the input speech signal into acoustically homogeneous segments. The speaker identification module uses these homogeneous segments to classify and label the speaker based on an extant database (or data store) of enrolled speakers. A snapshot of the user-interface built for this system showing the multiple analyses in progress is shown in Figure 2.

All three components run in real-time to analyze live audio on a 333+ MHz Pentium II class machine with Windows 95/98/NT without any additional hardware. Our experimental set up includes a VCR playing a recorded broadcast news show from one of the major networks with the audio output directly plugged into the computer's sound card. The speech transcription accuracy is about 80%, the segmentation component has an 80% success rate, and speakers are identified correctly 95% of the time (in matched channel conditions).

Applications for such systems are numerous. The output from the speech recognition phase can be gathered into documents which can be indexed for retrieval by content. This is the main thrust of most Web-search engines, however, in this case, the transcriptions are approximate. These approximations can be somewhat compensated by building into the user-interface the ability to display the original video or audio clip in response to a user query in addition to the retrieved text. The speaker identification component yields labeled segments spoken by a single speaker. The accuracy of speaker segmentation plays a big role in determining how unsullied each segment is. The identified segments can be indexed for speaker-based retrieval. Moreover, these two indexes can be combined to respond to user queries which include both content and speaker introducing a different type of boolean search [1].

A review of the related literature in this area of audio (and video) processing indicates a lot of activity in exploring techniques for searching audio (and video) databases. These include a related approach in which various sounds found in everyday life are categorized and later retrieved [2], and a system for indexing video and audio using both video-based indexing and speech transcription [3]. Yet another approach is to focus on one class of sounds such as music and then applying search techniques most appropriate for that class [4, 5]. Generally, broadcast news audio does not lend itself into simple classifications - at best it can be described as being predominantly speech. The following sections describe *TranSegId* and the techniques behind it.

2. Front-End Processing

TranSegId accepts an uncompressed audio signal sampled at 22 KHz either from a live audio feed or a disk-resident PCM or wav file. A common signal processing front-end is used for all of its three modules. The audio input is converted into 24-dimensional mel-cepstral feature vectors with each feature vector (or frame) representing 10 ms of audio. The generation of feature vectors takes about 5% of the time required for speech recognition and the output is transmitted to all three modules simultaneously. No other initial processing of the audio is performed at this stage. Since live audio data can only be derived via the sound card in the computer, a portion of the front-end is hardware dependent. No such restriction exists for disk-resident audio files.

3. Transcription

The transcription engine translates the frames delivered by the front-end processor to recognized text. The IBM ViaVoice Broadcast News engine is used for transcription [6, 7]. This engine relies on a vocabulary of about 60,000 words; an acoustic model trained with 70 hours of broadcast news data; and a language model built using the transcripts for the aforesaid 70 hours plus a corpus of 400 million words of broadcast news text.

The vocabulary defines the words that can be transcribed, i.e., if a word is not in the vocabulary, it cannot be recognized. The acoustic model generates candidate words by combining phonemes to form words. A mixture of both continous and spontaneous speech is found in broadcast news. Other speech conditions modeled include are low fidelity speech, non-native speakers, speech plus music, speech with background noise, telephonic speech, and various combinations of the above. The language model is a domain-specific database of sequences of words in the vocabulary. Using the 400 million words, probabilities of word sequences are abstracted and recorded when the recognition system is built. The IBM broadcast news speech transcription system uses trigram language model.

The output of this module is a succession of time-stamped words. Table 1 presents the transcription performance on a standard two-hour broadcast news evaluation test.

Speech Conditions	WER (%)
Prepared Speech	22.3
Spontaneous Speech	29.6
Low fidelity Speech	39.6
Speech+Music	37.5
Speech+Background noise	35.1
Non-native speakers	29.7
Overall	29.7

Table 1. Word error rate (WER) for IBM's real-time system for broadcast news

4. Speaker Segmentation

The aim of speaker segmentation is to partition the incident speech into segments uttered by different speakers automatically. The segmentation scheme is more of an acoustic change detector. That is, the incident speech is segmented not only due to speaker change but also due to changes in the underlying acoustics of the speech signal such as changes in speech delivery (volume change or slow, pausing speech), or changes in background conditions occurring while a speaker is speaking. For example, hesitation or volume change may over-segment the speech of the single speaker.

The segmentation engine uses the Bayesian Information Criterion (BIC) to partition the frames produced by the front-end [8]. The basic problem may be viewed as a two-class classification where the object is to determine whether N consecutive audio frames constitute a single homogeneous window of frames (or segment) W or two such windows: W_1 and W_2 with the boundary frame or "turn" occurring at the ith frame.

In order to detect if a speaker change occurred within a window of N frames, two models are built. One which represents the entire window by a Gaussian characterized by $\{\mu, \Sigma\}$; a second which represents the window up to the ith frame, W_1 with $\{\mu_1, \Sigma_1\}$ and the remaining part, W_2, with a second Gaussian $\{\mu_2, \Sigma_2\}$.

The details of this classifier can be formulated as:

$$\begin{aligned}\Delta BIC = \quad & -\tfrac{N}{2} log|\Sigma| + \tfrac{i}{2} log|\Sigma_1| + \tfrac{i}{2} log|\Sigma_2| \\ & + \tfrac{1}{2}\lambda \left(d + \tfrac{d(d+1)}{2}\right) log N;\end{aligned}$$

where, d is the dimension of the cepstral vectors; N is the number of frames in window W; and λ is a penalty function which should be nominally 1, but for 24-dimensional feature vectors it is empirically determined that 1.3 yields better results. It should be noted that this formulation assumes independent feature vectors but not uncorrelated feature elements.

A turn is confirmed at frame i when i is not only the minimizer of ΔBIC but also drives it negative. Otherwise the window size is increased incrementally by a fraction of N, and the test is applied again. When the window exceeds a predefined size without a segment boundary, the next batch of N frames are concatenated with the first, the model paramters are recomputed using a small overlap with the first N frames, and the BIC test is re-applied.

The performance of such a segmenter can be gauged by the extent that it tracks the true segments. This is measured in terms of over-segmentation, missed segments, and segmentation resolution. Different costs can be associated with each of these errors, based on the application at hand. In *TranSegId*, the least desirable are the missed segments, as it can cause two distinct speaker segments to be merged into one, which in turn engenders a single identification tag for the whole segment, generating a label for the speaker in the initial portion and ignoring the second. This issue is touched upon again in the speaker identification section.

Table 2 presents the performance of the BIC segmenter in testing five hours of broadcast news data with commercials excluded.

Missed segments	15%
Over segmentation	6%
Turn resolution	+/- 50 frames

Table 2. Speaker segmentation results

5. Speaker Identification

The speaker recognition module receives the frames from the front-end directly while obtaining the turns information from the segmentation module (Figure 1). The IBM speaker recognition engine is text-independent and language-independent and is SVAPI-compliant.

Speaker identification calls for a pre-existing data store of labeled speakers and their (broadcast news originated) voice prints (at least 30 seconds worth). The datastore is built in the course of a prior off-line enrollment process. PCM files representing the voice prints of the speakers of interest are used to enroll the speakers. Also generated is a verification binary tree which includes each speaker and a cohort set of speakers "closest" to each speaker based on a distance measure. At run time, the data store along with the utterance derived from successive speech segments are submitted for identification.

Each enrolled speaker is modeled by a set of multi-dimensional Gaussian distributions for which the number of distributions, mean vectors, covariance matrices and priors are retained in the data store [9].

Let $\{M_i \mid i = 1..I\}$ denote the models pertaining to each of the enrollees. Each model M_i can have N_j distributions associated with it. Let ω_{ij} refer to the jth distribution of model i. Also, let $\{\vec{x}_t \mid t = 1..T\}$ denote the frames constituting the test utterance whose label is sought. During run-time, a test utterance $\vec{x}_t$ is identified with model i according to:

$$\vec{x}_t \,\epsilon\, M_q \;\; iff \;\; D_q = \min_{i=1..I} [D_i],$$

where

$$D_i = \sum_{t=1}^{T} d(i,t), \; i = 1..I,$$

and

$$d(i,t) = \sum_{j=1}^{N_j} P(\omega_{ij})\, p(\vec{x}_t|\omega_{ij}),$$

with $P(\omega_{ij})$ being the prior of the jth distribution of model i, and $p(\vec{x}_t|\omega_{ij})$ being the conditional pdf of utterance conditioned on the jth component of model i. Then,

$$p(\vec{x}_t|\omega_{ij}) = \frac{e^{-\frac{1}{2}(\vec{x}_t-\vec{\mu}_{i,j})^t \Sigma_{i,j}^{-1}(\vec{x}_t-\vec{\mu}_{i,j})}}{(2\pi)^{d/2}|\Sigma_{i,j}|^{1/2}}$$

Identification comprises two stages. First, in the class assignment stage, the test utterance is assigned to one of the prototype classes established in the course of prior training (enrollment). This stages produces an ID for the speaker. Next, in a verification stage the resultant class assignment (ID) from the first stage is subjected to a verification test. During verification the claimed speaker ID is confirmed using a second pass over the same data. The result of verification is a boolean.

An utterance must persist for a minimum of eight seconds to qualify for identification. Otherwise it is dismissed as "inconclusive" without further processing. Hence, only the first eight seconds of each segment derived from the segmentation process is used in speaker identification. The improved accuracy obtained in holding off the decision until the entire segment is analyzed for speaker identification is too small to offset the effectiveness and value of displaying a speaker ID as soon as it is possible to obtain it. This approach also penalizes turns missed during segmentation since the second portion of mistakenly merged segments is completely ignored by the speaker segmentation module (provided the putative first segment is at least eight seconds long).

Although the first stage of the identification process is inherently a closed-set, i.e., the only possible labels are those in the database of enrolled speakers), the subsequent verification stage transforms it into an open-set, as unverified speaker labels can be rejected. The identification result for five hours of broadcast news data is shown in Table 3.

Segments	Rate (%)
Correctly Identified	83
Correctly Verified	85

Table 3. Speaker identification performance

It is important that the audio channels from which speaker sample data are extracted for enrollment match those expected during speaker identification. In our experiments, the enrollment data was derived by gathering one continuous 30-second speech segment per speaker from a broadcast news video program. The degradation evident from Table 3 as compared to the numbers cited earlier is because under test conditions, the channels over which some of the speakers were broadcasting did not match the enrollment conditions. It is therefore recommended that multiple sources of speaker samples covering multiple channel conditions be used during enrollment. *TranSegId* includes a feature to enroll speaker using multiple files. When no channel mismatch occurs, the speaker identification performance is 95%.

6. Results

The experimental set up included a Pentium II 333 MHz single-processor IBM PC running Windows NT with no additional hardware. A VCR playing a recorded broadcast news show was the source of the audio input. The audio was fed directly into the computer's sound card input. Our test runs did not include commercial segments of news shows.

The three modules operate on the input frames derived from the common signal processing front-end concurrently but asynchronously. The time-stamped words from the transcription engine are routed through a FIFO so that they can be interleaved with the turns reported by the segmenter as they arrive. Each new turn generates a line break in the display screen, giving the displayed result a paragraph-like structure (Figure 2).

As turns are located with resolutions coarser than one frame (nominally +/- 50 frames), they can occur not only in between transcribed words but also within them. In the latter case, a turn is reassigned to either before or after the word based on its proximity to the word's leading or trailing boundary.

The input to the identification engine is the data store and the test utterance computed from the input frames and turn information. Only the first eight seconds of each speech segment is of concern in identification. The process does not wait for the trailing segment boundary. As an additional precaution, to compensate for the segmenter's resolution of +/- 0.5 seconds around a computed turn, the first second and last second of the initial eight seconds of each speaker segment are sliced off before submission to the speaker identification component. The effective duration of a test utterance for speaker identification is six seconds. Segments shorter than eight seconds are marked with the label "Inconclusive". Utterances are not computed for such segments nor speaker identification tests applied.

Identified speakers are subjected to verification test. The label depicting the speaker's name is displayed only after the start location of the corresponding segment is established on the screen. Relative delays in throughput are managed using internal program buffers to make sure that this is always true to avoid the absurd condition of naming a speaker before any transcribed words show up on the user interface. Verified speaker labels are displayed in green, identified (but unverified) speakers in red.

The word error rate for speech transcription is about 20%, the segmentation component generates about 15% more turns than warranted while missing 5% of the true segments, and the speaker identification accuracy is 83%. All of these results are on five hours of different taped news programs recorded over a 30-day period in 1996 all from the same broadcast network. The shows included both morning and evening shows, some prime time and others not. The transcription engine training data was mutually exclusive from our test data though the engine did use data from this broadcast network as part of its training. 27 speakers were enrolled for testing our system, with a general rule that the enrollment and test data are not extracted from the same news story or show. We had to make five exceptions to this rule. In these cases - with the speakers appearing just once in the entire five hours - the training data did not include the first eight seconds of the speech segment.

7. Conclusion

We have discussed the design and implementation details of a system for concurrent transcription, speaker segmentation and speaker identification for audio, principally speech. The system provides the first step in the indexing of audio material for search and retrieval. *TranSegId* processes 60 minutes of a broadcast news show in real-time with the audio transcribed with about 80% accuracy and various speaker segments identified and labeled with an accuracy of about 85%. We have also used the output of this system, time stamped words and labeled sections, to index broadcast video. While we expect the accuracy numbers for all three modules to increase over time, demonstrably good results are achieved in audio cataloging with even the current performance.

One usability issue is the requirement that speakers be enrolled before they can be identified. While this cannot be avoided completely, building a good interface to provide new speaker labels and voice samples will be helpful. Our system provides an off-line batch training capability to add new speakers from the user interface. The dependence of speaker segmentation on the energy component of the signal makes it vulnerable to amplitude changes in the input signal. This is not so for the speaker identification component per se, but as a function that is dependent on the output of the segmentation process, its performance suffers correspondingly. This is one area worth addressing in the future.

Closed-set identification by definition can only label speakers that belong in the enrollment database. Of course, this can lead to absurd labels when run against a new broadcast show with hitherto unseen speakers. While using the verification process as a second pass to eliminate this problem is one approach, building a rejection model can address this issue during the class assignment stage itself. A large collection of speakers' data are pooled together to form a background models. This generally drops the identification rates, but performance overall improved since unknown speakers are handled more elegantly.

References

[1] M. Viswanathan, et al. "Retrieval from spoken documents using content and speaker information." To appear: *Proc. Int'l Conf. on Document Analysis and Recognition*, Bangalore, India, September 1999.

[2] S. Srinivasan, et al. "The CueVideo Spoken Media Retrieval System." IBM Almaden Research Report ARC6292, April 1999. Also: *Proc. SIGIR 99*, August 99, Berkeley, CA.

[3] E. Wold, et al. "Content-Based Classification, Search, and Retrieval of Audio." *IEEE Multimedia, Volume 3, Number 3, pp. 27-36*, 1996.

[4] B. Arons. "SpeechSkimmer: A System for Interactively Skimming Recorded Speech." *ACM Transactions on Computer-Human Interaction, Volume 4, Number 1, pp. 3-38*, 1997.

[5] S. Pfeiffer, et. al. "Automatic Audio Content Analysis." *Proc. MM'96, pp. 21*. ACM Press.

[6] L. R. Bahl, et al. "Robust Methods for Context-dependent Features and Models in a Continuous Speech Recognizer." *Proc. ICASSP*, 1994.

[7] P. S. Gopalakrishnan, et al. "A Tree Strategy for Large Vocabulary Continuous Speech Recognition." *Proc. ICASSP*, 1995.

[8] S. S. Chen, et al. "Speaker, Environment and Channel Change Detection and Clustering via the Bayesian Information Criterion." *Proc. DARPA Workshop*, 1998.

[9] H. S. M. Beigi, et al. "IBM Model-based and Frame-by-frame Speaker Recognition." *Proc. Speaker Recognition and its Commercial and Forensic Applications*, 1998.