Retrieval from Spoken Documents Using Content And Speaker Information

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

Abstract

There has been a recent upsurge in the deployment of emerging technologies such as speech and speaker recognition which are reaching maturity. We discuss the details of the components required to build a system for audio indexing and retrieval for spoken documents using content and speaker based information facilitated by speech and speaker recognition. The real power of spoken document analysis is in using both content and speaker information together in retrieval by combining the results. The experiments described here are in the broadcast news domain, but the underlying techniques can easily be extended to other speech-centric applications and transactions.

1. Introduction

The goal of an audio-indexing system is to provide the capability of searching and browsing through audio (and video) content. The system is formed by integrating information retrieval methods with large vocabulary continuous speech recognition and speaker recognition techniques. A large vocabulary continuous speech recognition system is used to produce time-aligned transcripts of the speech. The speech signal is also divided into acoustically homogeneous segments and classified and labeled. Information retrieval techniques are then employed on these recognized transcripts, combined with the labeled speaker segments to identify locations in the text that are relevant to the search request. These locations with time-alignments then specify regions of the speech that are relevant for the request. The time information is used to extract the appropriate audio and video segments from the source audio/video.

Conventional information retrieval has mainly focused on retrieving text documents from large collections of text. The basic principles of text retrieval are well established and have been well documented [1]. The index is a mechanism to match descriptions of documents with descriptions of queries. The indexing phase describes documents as a list of words or phrases, and the retrieval phase describes the query as a list of words or phrases. A document (or a portion thereof) is retrieved when its description matches the description of the query.

Data and retrieval models required for multimedia objects are quite different from those required for text objects. Typically, the indexing phase analyzes the multimedia object for certain features which are then converted into feature vectors for comparison with the feature vectors derived from the retrieval station. Then, similarities between feature vectors of these objects are computed to rank the data set. There is little consensus on a standard set of features for multimedia data. One approach to index an audio database might be to use certain audio cues such as an applause, music or speech. Similarly, for video it might be key frames, or shot changes. However, for real world audio derived from radio or TV broadcasts which are predominantly speech, the text may be generated using speech recognition (and the speaker labeled using speaker recognition) and the terms (and segments) so derived can be used for indexing the associated audio (and video). See [2, 3] for other approaches to indexing content from audio and video. A description of our approach follows.

2. Content-Based Information Retrieval

Our content-based information retrieval system consists of two components: (1) A large vocabulary continuous speech recognition system, and (2) a text-based information retrieval system.

2.1. Speech Recognition System

Speech recognition systems are typically guided by three components: a vocabulary, a language model, and set of pronunciations for each word in the vocabulary. A vocabulary is a set of words that is used by the recognizer to translate speech to text. As part of the decoding process, the recognizer matches the acoustics from the speech input to words in the vocabulary. Therefore, the vocabulary defines what words can be transcribed. If a word that is not in the vocabulary is to be recognized, it must first be added to the vocabulary. A language model is a domain-specific database of sequences of words in the vocabulary. A set of probabilities of the words occurring in a specific order is also required. The output of the recognizer will be biased towards the higher probability word sequences when the language model is operative.

The IBM speech recognition system for broadcast news uses 70 hours of broadcast news data for training acoustic models [4, 5]. The language model (LM) has a vocabulary of 65K most frequent words from the broadcast news language model corpus. The baseline LM is trained on the 70 hours of acoustic training data transcriptions plus 400 million words of broadcast data exhibiting a wide variety of speaking styles, environmental and background noise conditions, intended to model real-world spontaneous audio.

The speech recognition system runs at better than real-time on a 266 MHz Pentium II PC. The system was tested on the 1997 Hub4 evaluation test set which consists of two hours of broadcast news. The word error (deletions, substitutions, and insertions) rates for various speech and background conditions are given below in Table 1.

Speech Conditions	WER (%)
Prepared speech	22.2
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 Rates for Reco System

2.2. Some Issues for Retrieval Systems

Retrieval systems that use the text generated by speech recognition systems face several issues and constraints. Unless otherwise derived from the acoustic information (lip smacks, pauses) or special language constructs, all sentence structure information is lost in the decoding. With audio and video, the granularity of the retrieved element is another issue to contend with. One approach is to assume that the input audio or video clip has to be retrieved as is - like a single document. Then, in response to a query, the relevant audio or video clip would be retrieved from a repository of such clips. Alternately, the transcript from each audio or video clip in the repository may be broken down into several chunks (documents) during indexing. In this case, the search results would be obtained as a time-aligned document within a specific audio or video clip. A third approach would be to retrieve the relevant clip and the points of interest within it.

2.3. Content-based Retrieval

Information retrieval systems work in two phases - an off-line indexing phase and an on-line searching and retrieving phase. In the indexing phase, the text output from the speech recognition output, a continuous stream of time-aligned words, is processed to derive a document description which is used in the retrieval phase for rapid searching. There are several operations performed in sequence in this processing: tokenization to detect sentence boundaries, part-of-speech tagging, followed by morphological analysis, and then stop-word removal using a standard stop-word list. In morphological analysis, nouns are decomposed into their roots along with a tag to indicate the plural form. Verbs are decomposed into units designating person, tense and mood, along with the root of the verb. For example, the statement "*Today I am launching an effort to ban land mines*", after processing becomes: "*Today I be launch an effort to ban land mine*".

In general, our retrieval system used a two-pass approach, but in the system discussed in this paper, we use just the first-pass, principally to improve performance albeit at the cost of lower average precision. The following version of the Okapi formula [6], for computing the matching score between a document d and a query q is used:

$$S(d,q) = \sum_{k=1}^{Q} c_q(q_k) \frac{c_d(q_k)}{\alpha_1 + \alpha_2 \frac{l_d}{\bar{l}} + c_d(q_k)} \; idf(q_k).$$

Here, q_k is the kth term in the query, Q is the number of terms in the query, $c_q(q_k)$ and $c_d(q_k)$ are the counts of the kth term in the query and document respectively, l_d is the length of the document, $\bar{l}$ is the average length of the documents in the collection, and $idf(q_k)$ is the inverse document frequency for the term q_k which is given by:

$$idf(q_k) = log(\frac{N - n(q_k) + 0.5}{n(q_k) + 0.5}),$$

where N is the total number of documents and $n(q_k)$ is the number of documents that contain the term q_k. The inverse document frequency term thus favors terms that are rare among documents. (For unigrams, $\alpha_1 = 0.5$ and $\alpha_2 = 1.5$.) Clearly, the idf can be pre-calculated and stored as can most of the elements of the scoring function above except for the items relating to the query.

Each query is matched against all the documents in the collection and the documents are ranked according to the computed score from the Okapi formula above. The scoring function takes into account the number of times each query term occurs in the document normalized with respect to the length of the document to remove bias that generally favor longer documents. This function also favors terms that are specific to a document and rare across other documents.

2.4. Evaluation and Results

Approximately 20 hours of Voice of America radio news broadcasts were collected over a span of May-June 1996. There were 10 main speakers, male and female, with scores of correspondents and interviewees, both American and foreign speakers of English. Each broadcast was typically 6 or 10 minutes long. The entire speech collection was transcribed with time-alignments for each word in the transcript. These were automatically collected into overlapping segments of a fixed number of words and treat each segment as a separate document [7].

Search requests were generated by having users read the documents and compose queries. This approach is also known as "known item retrieval." Overall there were 49 topics or questions and 1451 documents. The performance of this system is shown in Table 2. The average precision after the first pass was computed to be 69.92%.

Known items found at rank:	
≤ 1	37
≤ 5	41
≤ 10	46
≤ 20	47
≤ 100	47
Not found:	0

Table 2. Known Item Retrieval Performance

3 Speaker-Based Retrieval

The current speaker-based information retrieval system consists of two components: (1) an acoustic-change detection system also called speaker segmentation, and (2) a speaker-independent, language-independent, text-independent speaker recognition system. The following is a discussion of the constituent elements.

3.1. Speaker Segmentation

In order to completely automate the process of speaker identification, for speaker retrieval for audio-indexing systems, it is necessary to detect the boundaries (turns) between non-homogeneous speech portions. Each homogeneous segment must ideally correspond to a single speaker. Once delineated, each segment - if it meets the minimum segment length requirement required for speaker recognition - can be classified as having been spoken by a particular speaker. These segments are in some sense the equivalent of "documents" in that they form individual units of retrieval.

3.2 The Bayesian Information Criterion

The model-selection criterion used in this audio-indexing system is called Bayesian Information Criterion (BIC) [8]. The input audio stream can be modeled as Gaussian process in the cepstral space. BIC is a maximum likelihood approach to detect (speaker) turns of a Gaussian process. The problem of model identification is to choose one among a set of candidate models to describe a given data set. It assumes the frames (10 ms) derived from the input audio signal are independent and result from a single-gaussian process. In order to detect if there is a speech change in a window of N feature vectors after the frame i, $1 \leq i < N$, two models are built: one which represents the entire window by one Gaussian characterized by its mean and full covariance $\{\mu, \Sigma\}$, and a second which represents the first part of the window, up to frame i, with a first Gaussian $\{\mu_1, \Sigma_1\}$, and the second part of the window with another Gaussian $\{\mu_2, \Sigma_2\}$. The criterion is then expressed as: $\Delta BIC(i) = -R(i) + \lambda P$, where $R(i) = \frac{N}{2} log|\Sigma| - \frac{N_1}{2} log|\Sigma_1| - \frac{N_2}{2} log|\Sigma_2|$ and $P = \frac{1}{2}(d + \frac{d(d+1)}{2}) log N$ is the penalty associated to the window, $N_1 = i$ is the number of frames of the first part of the window, and $N_2 = (N - i)$ is the number of frames of the second part; d is the dimension of the frames. Therefore, P reflects the complexity of the models, as $d + \frac{d(d+1)}{2}$ is the number of parameters used to represent the Gaussians.

$\Delta BIC < 0$ implies, taking the penalty into account, the model splitting the window into two Gaussians is more likely than the model representing the entire window with only a single Gaussian. The BIC therefore behaves like a thresholded likelihood ratio criterion, where the threshold is not empirically tuned but has a theoretical foundation. This criterion is robust and requires no prior training.

3.3 BIC Implementation

The feature vectors used are 24-dimensional mel-cepstra frames. No other processing is done on these vectors. The algorithm works on a window-by-window basis, and in each window, a few frames are tested to check whether there are BIC-prescribed segment boundaries. If no segment boundary is found ($\Delta BIC > 0$), then the window size is increased. Otherwise, the old window location is recorded, which also corresponds to the start of a new window (with original size). A set of broadcast news clips were used to test out the BIC algorithm and implementation for this audio indexing task. The news clips had in all 104 segments. BIC segmentation yielded 84 segments of which 5 were erroneous (non-existent turns), and 25 segments were missed.

3.4 Speaker Classification and Training

The speaker recognition engine has two different implementations: model-based and frame-based [9]. The engine is both text and language independent, for live audio indexing of material such as broadcast news. The engine also shares the signal processing front-end and feature vector extraction phases with the IBM speech recognition engine and is SVAPI compliant. The engine offers several functions including speaker enrollment, classification, identification, and verification. In order to be able to index an audio stream by speaker, a database of speakers' audio samples must be present in a speaker database. The process of adding speaker's voice samples to such a database is called enrollment. This is an off-line process and our audio indexing system assumes such a database exists for all speakers of interest. About a minute's worth of audio is required from each speaker from multiple channels and microphones encompassing multiple acoustic conditions. The training data or database of enrolled speakers is stored using a hierarchical structure so that accessing the models is optimized for efficient recognition and retrieval.

3.5 The Model-Based Approach

To create a set of training models for the population of speakers in the database, a model $\mathcal{M}_i$ for the i^{th} speaker based on a sequence of M frames of speech, with the d-dimensional feature vector $\{\vec{f}_m\}_{m=1,\ldots,M}$, is computed. These models are stored in terms of their statistical parameters, such as, $\{\vec{\mu}_{i,j}, \mathbf{\Sigma}_{i,j}, \vec{C}_{i,j}\}_{j=1,\ldots,n_i}$, consisting of the Mean vector, the Covariance matrix, and the Counts, for the case when a Gaussian distribution is selected. Each speaker, i, may end up with a model consisting of n_i distributions. Now, using the distance measure proposed in [10] for comparing two such models, an hierarchical structure is created to devise a speaker recognition system with many different capabilities including speaker identification (attest a claim), speaker classification (assigning a speaker), speaker verification (second pass to confirm classification by comparing label with a "cohort" set of speakers whose characteristics match those of the labeled speaker), and speaker clustering.

The distance measure devised for speaker recognition permits computation of an acceptable distance between two models with different number of distributions n_i. Comparing two speakers solely based on the parametric representation of their models obviates the need to carry the features around making the task of comparing two speakers computationally less intensive. A short-coming of this distance measure for the recognition stage is that the entire speech segment has to be used to build the model of the claimant before computation of the comparison can begin. The method described in the next section alleviates this problem.

3.6 The Frame-By-Frame Approach

Let $\mathcal{M}_i$ be the model corresponding to the i^{th} enrolled speaker. $\mathcal{M}_i$ is entirely defined by the parameter set, $\{\vec{\mu}_{i,j}, \mathbf{\Sigma}_{i,j}, p_{i,j}\}_{j=1,\ldots,n_i}$, consisting of the mean vector, covariance matrix, and mixture weight for each of the n_i components of speaker i's Gaussian Mixture Model (GMM). These models are created using training data consisting of a sequence of M frames of speech, with the d-dimensional feature vector, $\{\vec{f}_m\}_{m=1,\ldots,M}$, as described in the previous section. If the size of our population is N_p, then the set of models we choose from, the model universe, is $\{\mathcal{M}_i\}_{i=1,\ldots,N_p}$. The fundamental goal is to find the i such that $\mathcal{M}_i$ best explains the test data, represented as a sequence of N frames, $\{\vec{f}_n\}_{n=1,\ldots,N}$, or to make a decision that none of the models describes the data adequately. The following frame-based weighted likelihood distance measure, $d_{i,n}$, is used in making the decision: $d_{i,n} = -log\left[\sum_{j=1}^{n_i} p_{i,j} p(\vec{f}_n | j^{th} \text{ component } of \mathcal{M}_i)\right]$, where, using a Normal representation, $p(\vec{f}_n|\cdot) = \frac{1}{(2\pi)^{d/2}|\Sigma_{i,j}|^{1/2}} e^{-\frac{1}{2}(\vec{f}_n - \vec{\mu}_{i,j})^t \Sigma_{i,j}^{-1} (\vec{f}_n - \vec{\mu}_{i,j})}$ The total distance, D_i, of model $\mathcal{M}_i$ from the test data is then taken to be the sum of all the distances over the total number of test frames. $D_i = \sum_{n=1}^{N} d_{i,n}$.

For classification, the model with the smallest distance to that of the speech segment is chosen. By comparing the smallest distance to that of a background model, one could provide a method to indicate that none of the original models match very well. Alternatively, a voting technique may be used for computing distance. For verification, a predetermined set of members that form the cohort of the labeled speaker is augmented with a variety of background models. Using this set as the model universe, the test data is verified by testing if the claimant's model has the smallest distance; otherwise, it is rejected. In both cases, the speaker data must match the acoustic conditions used in enrollment.

3.7 Implementation and Results

The speaker recognition engine was first tested on 104 10-second broadcast news audio clips against an enrolled database of 199 speakers with 30 seconds of training data per speaker. Using the frame-based approach, 101 out of 104 files were correctly classified and all of these were subsequently correctly verified. In the model-based approach, 92 were correctly classified, 12 were mis-classified, 1 out of the 92 was mis-verified (correct classification was negated by verification). Six out of the mis-classified 12 were not verified (the erroneous classification was detected by the verification module), and the remaining 6 were erroneously verified (both mis-classified and mis-verified).

For speaker-based audio indexing, nine of 84 segments generated during the segmentation process fell below the minimum required 8-second test window, leaving 75 segments to be identified during speaker recognition. Of these, 70 were identified correctly and verified. Four of the five mis-classifications were detected during verification, while the fifth was mis-verified (erroneous classification and verification). Both classification *and* verification were used in all experiments.

4 Content and Speaker Retrieval

4.1 Indexing for Content-Based Retrieval

The recognizer generates words along with time-alignments for each word (the start time of each word relative to the start of the audio or video clip) which are collected into "documents". For each of these "documents" statistics required by the Okapi equation are gathered and recorded in the index files along with the audio source file name. The time involved in generating the various index files is around 1–2% of the time required in transcription.

4.2 Indexing for Speaker-Based Retrieval

The index file for speaker-based retrieval is built from the results of speaker classification and verification. Each classification result is accompanied by a score which is the distance from the original enrolled speaker model to the audio test segment, start and end times of the segment relative to the beginning of the audio clip concerned, label (name of the speaker supplied during enrollment), and audio source file. For any given audio clip, all the segments assigned the same (speaker) label are gathered. They are then sorted by their scores and normalized by the segment with the best score. (For every new audio clip processed by the system and added to the index, all the labeled segments are again sorted and re-normalized.) The speaker index is a database of speakers, with multiple segments for each speaker (and possibly across multiple media files). During retrieval, where the identity of the speaker is specified as part of the query request, the specified speaker's list of segments are selected from the database and scanned for the appropriate audio segment using the recorded start and end times.

4.3 The Retrieval Engine

The retrieval engine is architected to process content-based and speaker-based queries either sequentially or concurrently. Search requests can take three forms: search by text content or the traditional information retrieval scenario, search by speaker which entails specifying a speaker by name, or a search by text *and* speaker. The retrieval engine is primed by first loading into its memory the pre-computed portion of the Okapi formula and other vocabularies.

Each word in the search string is tokenized, stemmed, and then matched against each of the fixed word-size documents in the index set. All of the document scores are then ranked and normalized. The top N documents alone are returned to the user. Also available are the start and end times of each of the N documents, scores, matched words that contributed to the relevance score, and the associated audio or video file names (stored during indexing). If the query contains an entry in the speaker name field, then the speakers' database (index) is searched and the top N segments are retrieved for that speaker along with the segment scores.

The top N retrieved items include all the gathered information about the retrieved document including a portion of the text of the document. We have developed a user interface to render this data effectively.

4.4 Combined Speaker and Content Searching

In combined speaker and content search, the user is looking for relevant audio/video segments which contain certain words spoken by a certain speaker. First, the text query is processed and *all* (and not just the top N) the scored documents are collected. For the specified speaker, all the segments' information is available from the speakers' index. The common elements between the content and speaker indexes are the start and end times of document chunks and speaker segments respectively. If there is a time overlap between a document from the text search and a segment from the speaker's index (when derived from the same audio source file), then the text corresponding to the overlapped segment satisfies the query. The degree to which the two segments overlap is also significant. A single segment from speaker retrieval may overlap with multiple segments from text retrieval or vice versa.

Next, the combined score from the two individual content and speaker searches is computed. For each document from the text search results, run down the segments for the specified speaker computing time overlaps given by: $C_s = (c_s + (\lambda * s_s)) * (o_f)$, where c_s is the score for the retrieved document from the content-based search, s_s is the score for the speaker segment, o_f is a fraction ($0 < o_f < 1$) that specifies by how much the speaker segment overlaps with the content segment. λ is a factor which handicaps s_s based on confidence in the speaker scores. Currently, a λ of 0.75 is used. (When better cohort models in the speaker recognition system and more speakers overall become available, λ will be closer to 1.) The resulting combined scores are re-sorted and normalized so that the best score is 100. The retrieval engine then returns the information about the top N composite segments for display to the user.

4.5 Results

We ran experiments for the entire system using five hours of broadcast news video data digitized from VHS tapes. (The amount of video data used was restricted simply because of the paucity of publicly distributed and freely usable broadcast news video material.) The video and audio were digitized separately, the video using MPEG-1, and the audio at 22 KHz PCM. Ten speakers were selected from the video material and enrolled into the speaker recognition system at least 15 seconds of the speaker's voice sample.

A 30-minute video segment was used in testing. The results are shown in Table 3. For speaker-based retrieval, the top 10 segments for 10 enrolled speakers were retrieved using both identification *and* verification. (Channel mismatch accounts for the performance degradation.) For content search, 10 user-defined queries from the video were used. The top 5 and top 25 results are presented below. Queries like "defense secretary" returned three hits for the whole phrase, and the remaining seven pertaining to just the word "defense" or "secretary". In the combined search, errors were both due to speaker mis-classifications and irrelevant documents being retrieved. Sample queries include "land mines/Bill Clinton" and "Boris Yeltsin/Natalie Allen").

Search	Relevant/Retrieved
Speaker	77/99
Content Top 5	198/200
Content Top 25	143/200
Combined Top 5	51/62
Combined Top 10	78/93

Table 3. Retrieval Performance

5 Conclusion and Further Work

We have described our experience with using large vocabulary speech recognition and speaker recognition for spoken document retrieval. Our goal of extracting information from audio material using imperfect automatic transcription and speaker recognition has been demonstrated. The current set of documents derived from the speech recognition output can be augmented by including the next-best guesses for each word or phrase from the recognizer. This information can be used for weighting the index terms, query expansion, and retrieval. Also, better recognition accuracy can be had by detecting segments with music or mostly noise so that only pure speech is indexed for retrieval. One limitation with the current approach to audio-indexing is the finite coverage of the vocabulary used in the speech recognizer. Words such as proper nouns and abbreviations that are important from an information retrieval standpoint are often missing in the vocabulary and hence in the recognized transcripts. One method to overcome this limitation is to complement the speech recognizer with a wordspotter for the out of vocabulary words. For this approach to be practical, however, one has to have the ability to detect spoken words in large amounts of speech at speeds many times faster than real-time.

A technique for combining results from two different modes of processing audio (and video) has been presented. In order to express the confidence in a recognized speaker in quantifiable terms, we have to normalize across speakers rather than within speaker segments. This can be done by running the training set as a test set, taking the mean of the top scores for all the correctly classified speakers, and then using this number to normalize all the scores. Our results for the combined retrieval across systems is based on five hours of video. However, the techniques presented here are scalable and we expect that larger video libraries can be indexed for search and retrieval using our system.

References

[1] G. Salton. *Automatic Text Processing*. Addison-Wesley, Reading, Massachusetts, 1989.

[2] M. Wechsler, E. Munteanu, and P. Schauble. New Techniques for Open-Vocabulary Spoken Document Retrieval Evaluation *Proc. SIGIR*, 1998.

[3] E. Wold, E. Blum, T. Keislar, and J. Wheaton. Content-Based Classification, Search, and Retrieval of Audio. *IEEE Multimedia, Volume 3, No. 3, pp. 27–36* 1996.

[4] L. R. Bahl, P. V. deSouza, P. S. Gopalakrishnan, D. Nahamoo, and M. A. Picheny. Robust Methods for Context-dependent Features and Models in a Continuous Speech Recognizer. *Proc. ICASSP*, 1994.

[5] P. S. Gopalakrishnan, L. R. Bahl, and R. Mercer. A Tree Strategy for Large Vocabulary Continuous Speech Recognition. *Proc. ICASSP*, 1995.

[6] S. E. Robertson, S. Walker, K. Sparck-Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. *Proc. Third Text Retrieval Conference (NIST special Publication)*, 1995.

[7] S. Dharanipragada, M. Franz, and S. Roukos. Audio-indexing for Broadcast News. *Proc. SDR97*, 1997.

[8] S. S. Chen and P. S. Gopalakrishnan. Speaker, Environment and Channel Change Detection and Clustering Via the Bayesian Information Criterion. *Proc. DARPA Workshop*, 1998.

[9] H. S. M. Beigi, S. Maes, U. V. Chaudhari, and J. S. Sorensen. IBM Model-based and Frame-by-frame Speaker Recognition. *Proc. Speaker Recognition and its Commercial and Forensic Applications*, 1998.

[10] H. S. M. Beigi, S. Maes, and J. S. Sorensen. A Distance Measure between Collections of Distributions and its Application to Speaker Recognition. *Proc. ICASSP*, 1998.