A System for Searching and Browsing Spoken Communications

Lee Begeja
Bernard Renger
Murat Saraclar
AT&T Labs – Research
180 Park Ave
Florham Park, NJ 07932
{lee, renger, murat}
@research.att.com

David Gibbon
Zhu Liu
Behzad Shahraray
AT&T Labs – Research
200 Laurel Ave S
Middletown, NJ 07748
{dcg, zliu, behzad}
@research.att.com

Abstract

As the amount of spoken communications accessible by computers increases, searching and browsing is becoming crucial for utilizing such material for gathering information. It is desirable for multimedia content analysis systems to handle various formats of data and to serve varying user needs while presenting a simple and consistent user interface. In this paper, we present a research system for searching and browsing spoken communications. The system uses core technologies such as speaker segmentation, automatic speech recognition, transcription alignment, keyword extraction and speech indexing and retrieval to make spoken communications easy to navigate. The main focus is on telephone conversations and teleconferences with comparisons to broadcast news.

1 Introduction

Archiving and organizing multimedia communications for easy user access is becoming more important as such information sources are becoming available in amounts that can easily overwhelm a user. As storage and access become cheaper, the types of multimedia communications are also becoming more diverse. Therefore, it is necessary for multimedia content analysis and navigation systems to handle various forms of data.

In this paper we present SpeechLogger, a research system for searching and browsing spoken communications, or the spoken component of multimedia communications. In general, the information contained in a spoken communication consists of more than just words. Our goal is to make use of all the information within a spoken communication. Our system uses automatic speech recognition (ASR) to convert speech into a format which makes word and phonetic searching of the material possible. It also uses speaker segmentation to aid navigation.

We are interested in a wide range of spoken communications with different characteristics, including broadcast material, lectures, meetings, interviews, telephone conversations, call center recordings, and teleconferences. Each of these communication types presents interesting opportunities, requirements and challenges. For example, lectures might have accompanying material that can aid ASR and navigation. Prior knowledge about the speakers and the topic may be available for meetings. Call center recordings may be analyzed to create aggregate reports.

Spoken document retrieval (SDR) for Broadcast News type of content has been well studied and there are many research and commercial systems. There has also been some interest in the Voicemail domain (Hirschberg et al., 2001) which consists of typically short duration human-to-machine messages. Our focus here is on telephone conversations and teleconferences with comparisons to broadcast news.

The paper is organized as follows. In Section 2, we motivate our approach by describing the user needs under various conditions. Then we describe our system in Section 3, giving the details of various components. Experimental results for some components are given in Section 4. Finally, in Section 5 we present a summary.

2 User Needs

We are primarily interested in situations in which a person needs to gather information from audio data but the quality of that data is not always sufficient to produce good ASR results. In the case of telephone conversations, the information gatherer needs to know who was on the call, how long the call was, what was said, a summary of the call, the ability to listen to any part of the call based on search parameters that s/he specifies, etc. Our users want to be able to scan a database of many calls, across a long period of time to look for specific phrases, speakers, or patterns of speech.

In many cases, it is difficult to gather this type of information from teleconference calls since the audio quality is poor because of speaker phones, cell phones and line noise. All of these combine to lower ASR results to a point where the text of the call is not fully representative

of the conversation. Thus, using standard information retrieval techniques may not provide sufficient information to the user. We focus on the navigation aspect of information gathering with the goal of compensating for lower ASR accuracy by presenting user interface elements relevant to the specific task at hand (Stark et al., 2000).

Rather than looking at the recorded conversation as merely audio information, we view it as a source of linguistic information to which we can apply information retrieval and data mining techniques. We use all available metadata to enhance the search and the presentation.

We wanted to have a set of interface elements that would be useful no matter what the ASR accuracy was. The main interface elements are:

- Timeline with tick marks indicates search hits within the spoken document which allows for many search results to be displayed without overwhelming the user. This is particularly useful for cases where there are many false positives.

- Keyword extraction summarizes a given communication, enables differentiation among a collection of many spoken documents, and detects subtopics in a large spoken document.

- Speaker segmentation and speaker identification separate a long spoken document into inherently useful pieces.

- Lattice search and phoneme search expand the possible search space.

In this paper we examine three classes of spoken documents and consider what tasks a user might want to perform on them.

- Broadcast News - excellent ASR conditions, one speaker at a time, good audio quality and generally a good speaker. Task involves primarily inter-document navigation. User needs to search text for information with metadata possibly used to enhance the search.

- Telephone Conversations - fair ASR conditions, two speakers, decent quality audio. User needs to search text but also wants speaker identification and some classification (call type, urgency, importance).

- Teleconferences - poor ASR conditions, multiple speakers, mixed to poor audio quality. Most time is spent in intra-document navigation. User needs to navigate through the calls and find relevant information in the audio.

3 System Description

The system overview is shown in Figure 1. Our system is flexible enough to support various forms of live (via a VoiceXML Gateway) or prerecorded spoken communications including the three classes of spoken documents discussed above. It can record the audio via telephone for two-party or multi-party calls. Alternatively, the system can support prerecorded audio input from various sources including telephone conversations or video content in which case the audio is extracted from the video. Once various speech processing techniques are applied and the speech is indexed, it is possible to search and browse the audio content. Our system is scalable and supports open source/industry standard components (J2EE, VXML, XML, Microsoft SAMI, Microsoft Media Player). It is also flexible enough to support other forms of audio as input or to support new speech processing techniques as they become available. The system was designed with modularity in mind. For instance, it should be possible to add a speaker identification module to the processing.

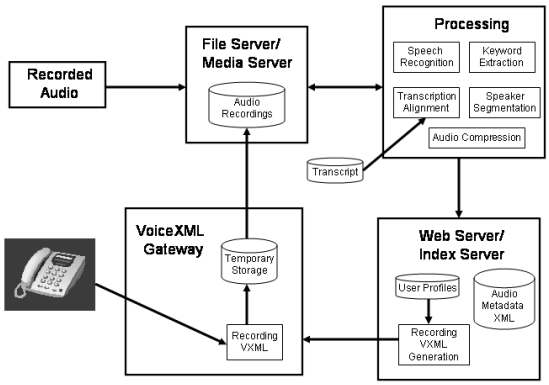

Figure 1: System Overview

Once a new audio recording is available on the File Server, the following processing steps can begin: speaker segmentation, speech recognition, transcription alignment, keyword extraction, audio compression, and speech indexing. Each step will be described in more detail below. We attempt to distinguish the different speakers from each other in the speaker segmentation component. The speech recognition component converts the audio into a word or phone based representation including alternative hypotheses in the form of a lattice. If a transcript is available, the transcript can be synchronized (or aligned) in time with the speech recognition output. The keyword extraction component generates the most salient words found in the speech recognition output (one-best

word) or transcript (if available) and can be used to determine the nature of the spoken communications. The audio compression component compresses the audio file and creates an MP3 audio file which is copied to the Media Server. The final step in the processing is text and lattice indexing. This includes creating indices based on one-best word and one-best phone strings or word and phone lattices.

After processing, the user can search and browse the audio using either the text index or the lattice index. The audio is played back via media streaming. Alternatively, the user can playback the audio file over the phone using the VoiceGenie VoiceXML Gateway.

3.1 Speaker Segmentation

Speaker-based segmentation of multi-speaker audio data has received considerable attention in recent years. Applications that have been considered include: indexing archived recorded spoken documents by speaker to facilitate browsing and retrieval of desired portions; tagging speaker specific portions of data to be used for adapting speech models in order to improve the quality of automatic speech recognition transcriptions, and tracking speaker specific segments in audio streams to aid in surveillance applications. In our system, speaker segmentation is used for more effective visualization of the audio document and speaker-based audio playback.

Figure 2 gives an overview of the speaker segmentation algorithm we developed. It consists of two steps: preprocessing and iterative speaker segmentation. During the preprocessing step, the input audio stream is segmented into frames and acoustic features are computed for each frame. The features we extracted are energy, 12 Mel-frequency cepstral coefficients (MFCC), pitch, and the first and second order temporal derivatives. Then, all speaker boundary candidates are located, which include silent frames and frames with minimum energy in a window of neighboring frames. The preprocessing step generates a set of over-segmented audio segments, whose durations may be as short as a fraction of a second to as long as a couple of seconds.

The iterative speaker segmentation step, as depicted in the bigger dotted rectangle in Figure 2, detects all segments of each speaker in an iterative way and then marks the boundaries where speakers change. At the beginning, all segments produced by the preprocessing step are unlabeled. Assuming that the features within each segment follow a Gaussian distribution, we compute the distances between each pair of segments using the Kullback Leibler distance (KLD) (Cover and Thomas, 1991). Here, we just consider features extracted from voiced frames since only voiced frames have pitch information. Based on the segment distance matrix, a hierarchical agglomerative clustering (HAC) (Jain and Dubes, 1988) algorithm is applied

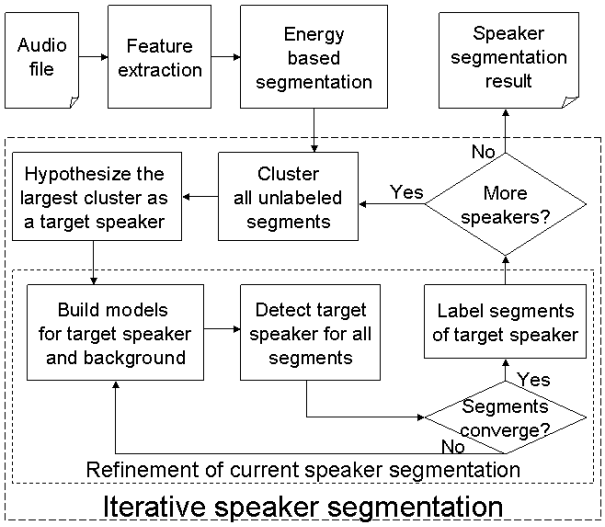

Figure 2: Overview of the Speaker Segmentation Algorithm.

to all unlabeled segments. The biggest cluster will be hypothesized as the set of segments for a new speaker and the rest of the segments will be considered as background audio. Accordingly, each unlabeled segment is labeled as either the target speaker or background. Then an embedded speaker segment refinement substep is activated to iteratively refine the segments of the target speaker.

The refinement substep is depicted in the smaller dotted rectangle in Figure 2. For each iteration, two Gaussian mixture models (GMM) are built based on current segment labels, one for the target speaker, one for background audio. Then all segments are relabeled as either the target speaker or background audio using the maximum likelihood method based on the two GMM models. If the set of segments for the target speaker converges or the refinement iteration number reaches its maximum, the refinement iteration stops. Otherwise, a new iteration starts. Before the refinement substep terminates, it assigns a new speaker label for all segments of the target speaker, and sets the background audio as unlabeled. Then the iterative speaker segmentation step needs to test for more speakers or needs to stop. The termination criteria could be the given number of speakers (or major speakers) in an audio document, the percentage of unlabeled segments to the number of all segments, or the maximum distance among all pairs of unlabeled segments. If any of the criteria are met, the speaker segmentation algorithm merges all adjacent segments if their speaker labels are the same, and then outputs a list of audio segments with corresponding speaker labels.

Obviously, one advantage of our speaker segmentation method is that the speaker labels are also extracted. Although the real speaker identities are not available, the

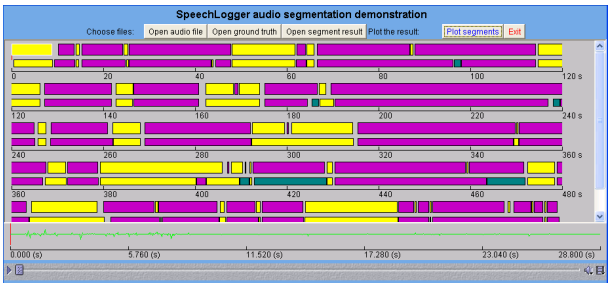

Figure 3: Presentation of Speaker Segmentation Results.

labels are very useful for presenting, indexing, and retrieving audio documents. For more detailed description of the speaker segmentation algorithm, please refer to Rosenberg et al. (2002).

Figure 3 illustrates a graphic interface for presenting the speaker segmentation results. The audio stream is shown in colored blocks along a timeline which goes from top to bottom, and from left to right. Color is used to differentiate the speaker labels. There are two layers for each line: the bottom layer shows the manually labeled speaker segments and the top layer displays the automatically generated segments. This allows the segmentation performance to be clearly observed.

3.2 Automatic Speech Recognition

We use two different state-of-the-art HMM based large vocabulary continuous speech recognition (LVCSR) systems for telephone and microphone recordings. In both cases the front-end uses 9 frames of 12 MFCC components and energy summarized into a feature vector via linear discriminant analysis. The acoustic models consist of decision tree state clustered triphones and the output distributions are mixtures of Gaussians. The models are discriminatively trained using maximum mutual information estimation. The language models are pruned backoff trigram models.

For narrow-band telephone recordings we use the first pass of the Switchboard evaluation system developed by Ljolje et al. (2002). The calls are automatically segmented prior to ASR. The acoustic models are trained on 265 hours of speech. The recognition vocabulary of the system has 45K words.

For wide-band recordings, we use the real-time Broadcast News transcription system developed by Saraclar et al. (2002). The acoustic models are trained on 140 hours of speech. The language models are estimated on a mixture of newspaper text, closed captions and high-accuracy transcriptions from LDC. Since the system was designed for SDR, the recognition vocabulary of the system has over 200K words.

Both systems use the same Finite State Machine (FSM) based LVCSR decoder (Allauzen et al., 2003). The output of the ASR system is represented as a FSM and may be in the form of a one-best hypothesis string or a lattice of alternate hypotheses. The lattices are normalized so that the probability of the set of all paths leading from any state to the final state is 1. The labels on the arcs of the FSM may be words or phones and the conversion between the two can easily be done using FSM composition using the AT&T FSM Library (Mohri et al., 1997). The costs on the arcs of the FSM are negative log likelihoods. Additionally, timing information can also be present in the output.

3.3 Alignment with Transcripts

Manual transcriptions of spoken communications are available for certain application domains such as medical diagnosis, legal depositions, television and radio broadcasts. Most audio and video teleconferencing providers offer transcription as an optional service. In these cases, we can take advantage of this additional information to create high quality multimedia representations of the archived spoken communications using parallel text alignment techniques (Gibbon, 1998). The obvious advantage is increased retrieval accuracy due to the lower word error rate (manual transcriptions are seldom completely error free.) What is more compelling, however, is that we can construct much more evolved user interfaces for browsing speech by leveraging the fact that the transcription is by its nature readable whereas the one-best hypothesis from ASR is typically useful only in small segments to establish context for a search term occurrence.

There are several methods for aligning text with speech. We use dynamic programming techniques to maximize the number or word correspondences between the manual transcription and the one-best ASR word hypothesis. For most applications, finding the start and end times of the transcript sentences is sufficient; but we do alignment at the word level and then derive the sentence alignment from that. In cases where the first or last word of a sentence is not recognized, we expand to the nearest recognized word to avoid cropping even though we may include small segments from neighboring sentences during playback. The accuracy of the resulting alignment is directly related to the ASR word error rate; more precisely it can be thought of as a sentence error rate where we impose a minimum percentage of corresponding words per sentence (typically 20%) before declaring a sentence a match to avoid noise words triggering false matches. For sentences without correspondences, we must fall back to deriving the timings from the nearest neighboring sentences with correspondences.

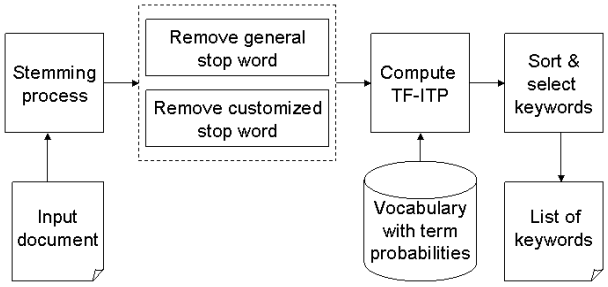

Figure 4: Illustration of Keyword Extraction.

3.4 Keyword Extraction

Playing back a spoken document or linearly skimming the corresponding text transcript, either from automatic speech recognition or manual transcription, is not an efficient way for a user to grasp the central topics of the document within a short period of time. A list of representative keywords, which serve as a dense summary for a document, can effectively convey the essence of the document to the user. The keywords have been widely used for indexing and retrieval of documents in large databases. In our system, we extract a list of keywords for each audio document based on its transcript (ASR or manual transcript).

There are different ways to automatically extract keywords for a text document within a corpus. A popular approach is to select keywords that frequently occur in one document but do not frequently occur in other documents based on the term frequency - inverse document frequency (TF-IDF) feature. Our task is slightly different. We are interested in choosing keywords for a single document, independent of the remaining documents in the database. Accordingly, we adopt a different feature, which is term frequency - inverse term probability (TF-ITP) to serve our purpose. The term probability measures the probability that a term may appear in a general document and it is a language dependent characteristic. Assuming that a term $\mathbf{T_k}$ occurs tf_k times in a document, and its term probability is tp_k, the TF-ITP of $\mathbf{T_k}$ is defined as $w_k = tf_k/tp_k$.

Figure 4 illustrates the keyword extraction method that we have developed. For the transcript of a given document, we first apply the Porter stemming algorithm (Porter, 1980) to remove word variations. Then, the stop words, which are common words that have no impact on the document content (also called noise words), are removed. Here we use two lists of noise words, one for general purposes, which apply to all varieties of documents, and one for specific domains, which can be customized by the user when prior knowledge about the document is available. For each remaining term in the document, a value of TF-ITP is calculated. A vocabulary is created based on the transcripts of 600 hours of broadcast news data and corresponding term probabilities are estimated using the same corpus. If a term in the document is not in the vocabulary, and its term frequency is more than 2, then a default term probability value tp_d will be used. The tp_d we use is the minimum term probability in the vocabulary. After we get a list of terms and their TF-ITP values, we sort the terms based on their TF-ITP values, such that the most representative terms (highest TF-ITP values) are on the top of the list. Depending on certain criteria, for example, the number of keywords desired or the minimum TF-ITP value required, a list of keywords can be chosen from the top of the term list. In our system, we choose the top ten terms as the keywords for a document.

3.5 Speech Indexing and Retrieval

Two different indexing and retrieval modules are utilized depending on the type of ASR output. In the case of one-best word or phone strings, we use an off-the-shelf text-based index server called Lucene (http://jakarta.apache.org/lucene). In the case of word and phone lattices, we use the method described in Saraclar and Sproat (2004). Here we give a brief description of the latter.

The lattice output is a compact representation of likely alternative hypotheses of an ASR system. Each path in the lattice corresponds to a word (or phone) string and has a probability attached to it. The expected count for a substring can be defined as the sum of the probabilities of all paths which contain that substring. Lattice based retrieval makes the system more robust to recognition errors, whereas phonetic search allows for retrieving words that are not in the vocabulary of the recognizer.

The lattice index is similar to a standard inverted index but contains enough information to compute the expected count of an arbitrary substring for each lattice. This can be achieved by storing a set of index files, one for each label (word or phone) l. For each arc labeled with l in a lattice, the index file for l records the lattice number, the previous and next states of the arc, along with the probability mass leading to the previous state of the arc and the probability of the arc itself. For a lattice, which is normalized so that the probability of the set of all paths leading from any state to the final state is 1, the posterior probability of an arc is given by the multiplication of the probability mass leading to the previous state and the probability of the arc itself. The expected count of a label given a lattice is equal to the sum of the posterior probabilities of all arcs in the index for that label with the same lattice number.

To search for a multi-label expression (e.g., a multi-word phrase) $w_1 w_2 \dots w_n$ we seek on each label in the expression and then for each (w_i, w_{i+1}) join the next

states of w_i with the matching previous states of w_{i+1}. In this way, we retrieve just those path segments in each lattice that match the entire multi-label expression. The probability of each match is defined as the multiplication of the probability mass leading to the previous state of the first arc and the probabilities of all the arcs in the path segment. The expected count of a multi-label expression for the lattice is computed as above.

The answer to a query contains an audio segment only if the expected count of the query for the lattice corresponding to that audio segment is higher than a threshold.

3.6 User Interface

The user interface description will apply for the three types of spoken communications (Telephone Conversations, Teleconferences, Broadcast News) although the audio and speaker quality do vary for each of these types of spoken communications. Once the user has found the desired call (or spoken communication) using one of the retrieval modules (one-best word, one-best phone string, word lattice, phone lattice, or both word and phone lattice), the user can navigate the call using the user interface elements described below.

For the one-best word index, the Web page in Figure 5 shows the user interface for searching, browsing, and playing back this call. The user can browse the call at any time by clicking on the timeline to start playing at that location on the timeline. The compressed audio file (MP3) that was created during the processing would be streamed to the user. The user can at any time either enter a word (or word phrase) in the Search box or use one of the common keywords generated during the keyword extraction process. The text index would be queried and the results of the search would be shown. The timeline plot at the top would show all the hits or occurrences of the word as thin tick marks. The list of hits would be found under the keyword list. In this case, the word "chapter" was found 4 times and the time stamps are shown. The time stamps come from the results of the automatic speech recognition process when the one-best words and time stamps were generated. The search term "chapter" is shown in bold with 5 context words on either side. The user can click on any of these 4 hits to start playing where the hit occurred. The solid band in the timeline indicates the current position of the audio being played back. The entire call, in this case, is 9:59 minutes long and the audio is playing at the beginning of the fourth hit at 5:20 minutes. As part of the processing, caption data is generated in Microsoft's SAMI (Synchronized Accessible Media Interchange) format from the one-best word output in order to show caption text during the playback. The caption text under the timeline will be updated as the audio is played. At this point in the call, the caption text is "but i did any chapter in a". This caption option can be disabled by clicking on the CC icon and can be enabled by clicking on the CC icon again. The user can also speed up or slow down the playback at any time by using the "Speed" button. The speed will toggle from 50% (slow) to 100% to 150% (fast) to 200% (faster) and then start over at 50%. The speed, which is currently "fast", will be shown next to the current time above the "Stop" button. This allows the user to more quickly peruse the audio file.

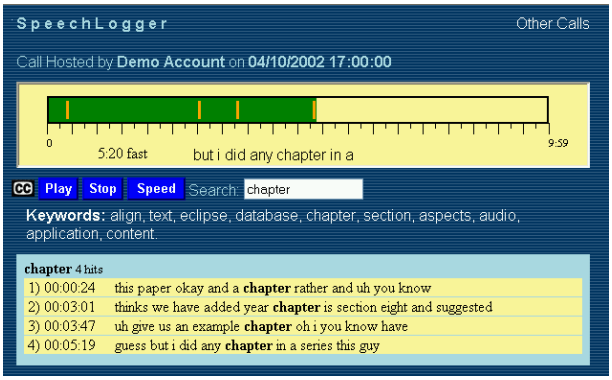

Figure 5: User Interface for ASR One-Best Word Search.

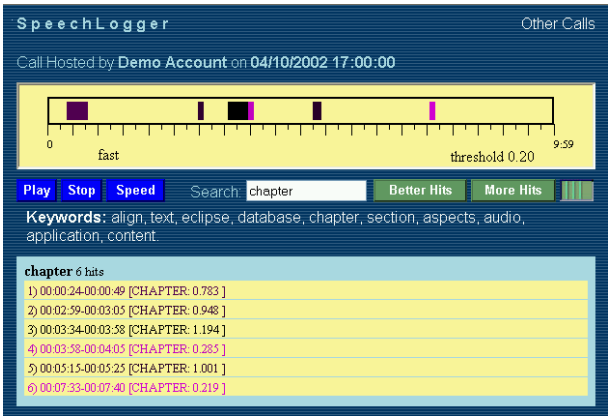

Figure 6: User Interface for Lattice Search.

A similar Web page in Figure 6 shows the user interface for searching a lattice index. Note that for the same audio file (or call) and the same search term "chapter", the results of the query show 6 hits compared to the 4 hits in the text index in Figure 5. In this particular case, the manual transcript does indeed contain these 6 occurrences of the word "chapter". The search terms were found in audio segments, which is why the time of the hit is a time range. The information in brackets is the expected count and can exceed 1.0 if the search term occurs more than once in the audio segment. The time range is reflected in the timeline since the thin tick marks have been replaced with colored segments. The colors of the segments correspond to the colors of the hits in the list. The darker the color, the higher the count and the lighter

the color, the lower the count. Finally, the search can be refined by altering the threshold using the "Better Hits" and "More Hits" buttons. In this example, the threshold is set to 0.2 as can be seen under the timeline. If the user clicks on the "Better Hits" button, the threshold is increased so that only better hits are shown. If the "More Hits" button is used, the threshold is decreased so more hits are shown although the hits may not be as good. The lattice index only returns hits where each hit has a count above the threshold.

The lattice search user interface allows the user to more easily find what the user wants and has additional controls (threshold adjustments) and visual feedback (colored segments/hits) that are not possible for the text search user interface.

4 Experimental Results

We used three different corpora to assess the effectiveness of different techniques.

The first corpus is the DARPA Broadcast News corpus consisting of excerpts from TV or radio programs including various acoustic conditions. The test set is the 1998 Hub-4 Broadcast News (hub4e98) evaluation test set (available from LDC, Catalog no. LDC2000S86) which is 3 hours long and was manually segmented into 940 segments. It contains 32411 word tokens and 4885 word types.

The second corpus is the Switchboard corpus consisting of two-party telephone conversations. The test set is the RT02 evaluation test set which is 5 hours long, has 120 conversation sides and was manually segmented into 6266 segments. It contains 65255 word tokens and 3788 word types.

The third corpus is named *Teleconference* since it consists of multi-party teleconferences on various topics. A test set of six teleconferences (about 3.5 hours) was transcribed. It contains 31106 word tokens and 2779 word types. Calls are automatically segmented into a total of 1157 segments prior to ASR.

4.1 Speaker Segmentation

The performance of the speaker segmentation is evaluated as follows. For an audio document, assume there are N true boundaries, and the algorithm generates M speaker boundaries. If a detected boundary is within 1 second of a true boundary, it is a correctly detected boundary, otherwise it is a falsely detected boundary. Let C denote the number of correctly detected boundaries, the recall and precision of the boundary detection can be computed as $R = C/N$ and $P = C/M$, respectively. We can combine these two values using the F-measure $F = 2 \times P \times R/(P + R)$ to measure the speaker segmentation performance.

We evaluated the developed method on three different types of audio documents: Broadcast News recordings (16KHz sampling rate, 16 bits/sample), two-party telephone conversations (8KHz, 16bps), and multi-party teleconference recordings (8KHz, 16bps). Due to the high audio quality and well controlled structure of the broadcast news program, the achieved F-measure for broadcast news data is 91%. Teleconference data has the worst audio quality given the various devices (headset, speakerphone, etc.) used and different channels (wired and wireless) involved. There are also a lot of spontaneous speech segments less than 1 second long, for example, "Yes", "No", "Uh", etc. These characteristics make the teleconference data the most challenging one to segment. The F-measure we achieved for this type of data is 70%. The F-measure for two-party telephone conversations is in the middle at 82%.

4.2 Automatic Speech Recognition

For evaluating ASR performance, we use the standard word error rate (WER) as our metric. Since we are interested in retrieval, we use OOV (Out Of Vocabulary) rate by type to measure the OOV word characteristics.

In Table 1, we present the ASR performance on these three tasks as well as the OOV Rate by type of the corpora. It is important to note that the recognition vocabulary for the Switchboard and Teleconference tasks are the same and no data from the Teleconference task was used while building the ASR systems.

Task	WER	OOV Rate by Type
Broadcast News	∼20%	0.6%
Switchboard	∼40%	6%
Teleconference	∼50%	12%

Table 1: Word Error Rate and OOV Rate Comparison.

4.3 Retrieval

Our task is to retrieve the audio segments in which the user query appears. For evaluating retrieval performance, we use precision and recall with respect to manual transcriptions. Let $C(q)$ be the number of times the query q is found correctly, $M(q)$ be the number of answers to the query q, and $N(q)$ be the number of times q is found in the reference. We compute precision and recall rates for each query as $P(q) = C(q)/M(q)$ and $R(q) = C(q)/N(q)$. We report the average of these quantities over a set of queries Q, $P = \sum_{q \in Q} P(q)/|Q|$ and $R = \sum_{q \in Q} R(q)/|Q|$. The set of queries Q includes all the words seen in the reference except for a stop list of the 100 most common words.

For lattice based retrieval methods, different operating points can be obtained by changing the threshold. The

precision and recall at these operating points can be plotted as a curve. In addition to individual precision-recall values we also compute the F-measure defined above and report the maximum F-measure (maxF) to summarize the information in a precision-recall curve.

In Table 2, a comparison of the maximum F-measure (maxF) is given for various corpora. Using word lattices yields a relative gain of 3-5% in maxF over using one-best word hypotheses. Using both word and phone lattices, the relative gain over the baseline increases to 8-12%. In this approach, we first search the word index; if no matches are found then we search the phone index. This allows the system to return matches even if the user query is not in the ASR vocabulary.

Task	System		
	1-best	W Lats	W+P Lats
Broadcast News	84.0	84.8	86.0
Switchboard	57.1	58.4	60.5
Teleconference	47.4	50.3	52.8

Table 2: Maximum F-measure Comparison.

In Figure 7, we present the precision-recall curves. The gain from using better techniques utilizing word and phone lattices increases as retrieval performance gets worse.

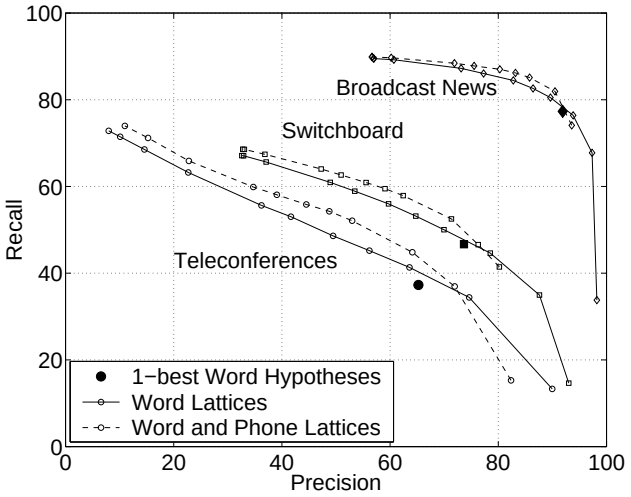

Figure 7: Precision vs Recall Comparison.

5 Summary

We presented a system for searching and browsing spoken communications. The system is flexible enough to support various forms of spoken communications. In this paper, our focus was on telephone conversations and teleconferences. We also presented experimental results for the speaker segmentation, ASR and retrieval components of the system.

Acknowledgments

We would like to thank Richard Sproat for useful discussions and for making his lattice indexing software (`lctools`) available for our system.

References

C. Allauzen, M. Mohri, and M. Riley. 2003. DCD Library – Decoder Library. http://www.research.att.com/sw/tools/dcd.

T. M. Cover and J. A. Thomas. 1991. *Elements of Information Theory*. John Wiley & Sons.

D. Gibbon. 1998. Generating hypermedia documents from transcriptions of television programs using parallel text alignment. In B. Furht, editor, *Handbook of Internet and Multimedia Systems and Applications*. CRCPress.

J. Hirschberg, M. Bacchiani, D. Hindle, P. Isenhour, A. Rosenberg, L. Stark, L. Stead, S. Whittaker, and G. Zamchick. 2001. Scanmail: Browsing and searching speech data by content. In *Proceedings of the European Conference on Speech Communication and Technology (Eurospeech)*, Aalborg, Denmark.

A. K. Jain and R. C. Dubes. 1988. *Algorithms for Clustering Data*. Prentice-Hall.

A. Ljolje, M. Saraclar, M. Bacchiani, M. Collins, and B. Roark. 2002. The AT&T RT-02 STT system. In *Proc. RT02 Workshop*, Vienna, Virginia.

M. Mohri, F. C. N. Pereira, and M. Riley. 1997. AT&T FSM Library – Finite-State Machine Library. http://www.research.att.com/sw/tools/fsm.

M. F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

A. Rosenberg, A. Gorin, Z. Liu, and S. Parthasarathy. 2002. Unsupervised speaker segmentation of telephone conversations. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, Denver, Colorado, USA.

M. Saraclar and R. Sproat. 2004. Lattice-based search for spoken utterance retrieval. In *Proc. HLT-NAACL*.

M. Saraclar, M. Riley, E. Bocchieri, and V. Goffin. 2002. Towards automatic closed captioning: Low latency real time broadcast news transcription. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, Denver, Colorado, USA.

L. Stark, S. Whittaker, and J. Hirschberg. 2000. ASR satisficing: the effects of ASR accuracy on speech retrieval. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*.