Automatic Broadcast News Speech Summarization

Sameer Raj Maskey

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2008

ABSTRACT

Automatic Broadcast News Speech Summarization

Sameer Raj Maskey

As the numbers of speech and video documents available on the web and on handheld devices soar to new levels, it becomes increasingly important to enable users to find relevant, significant and interesting parts of the documents automatically. In this dissertation, we present a system for summarizing Broadcast News (BN), ConciseSpeech, that identifies important segments of speech using lexical, acoustic/prosodic, and structural information, and combines them, optimizing significance, length and redundancy of the summary. There are many obstacles particular to speech such as word errors, disfluencies and the lack of segmentation that make speech summarization challenging. We present methods to address these problems. We show the use of Automatic Speech Recognition (ASR) confidence scores to compensate for word errors; present a phrase-level machine translation approach using weighted finite state transducers for detecting disfluency; and present the possibility of using intonational phrase segments for summarization. We also describe structural properties of BN used in determining which segments should be selected for a summary, including speaker roles, soundbites and commercials. We present Information Extraction (IE) techniques based on statistical methods such as conditional random fields and deci-

sion trees to automatically identify such structural properties. ConciseSpeech was built for handling single spoken documents, but we have extended it to handle user queries that can summarize multiple documents. For the query-focused version of ConciseSpeech we also built a knowledge resource (NE-NET) that can find related named entities to significantly improve the document retrieval task of query-focused summarization. We show how all these techniques improve speech summarization when compared to traditional text-based methods applied to speech transcripts.

Contents

List of Figures

List of Tables

Acknowledgement

In the last five and half years that I have spent at Columbia, I have come across many individuals of various background, race, and knowledge. Many of them have touched some part of my life in one way or other, and I thank them all for doing so. Among them there are a few individuals who made the biggest difference on my growth, on my knowledge and on my dissertation.

First among them is my advisor Prof. Julia Hirschberg. Coming out of the first meeting I knew Julia was going to be a great advisor and that has come out to be very true. She was always there to help, from making me understand the concepts of Natural Language Processing to pointing out my errors in writing. She was the only advisor as far as I knew whom one could go and talk to anytime, even in her busiest times she would listen to my problems. I am grateful to Julia for giving her time and her guidance, and for helping me fight the challenges I faced in my life and dissertation.

My previous advisors from my undergraduate years Prof. Alan W. Black and Prof. John Rhodes are other two individuals who have made a difference in the education route I have taken. Without the inspiration from John during the Math course I probably wouldn't have pursued mathematics as one of my majors, and if I hadn't met Alan I probably would never have pursued a PhD in computer science nor would I have enjoyed research as much as I do today.

I would like to thank my thesis committee members Prof. Kathy McKeown, Prof. Luis Gravano, Dr. Michiel Bacchiani, Prof. Regina Barzilay and Prof. Julia Hirschberg for taking their time to guide me through my thesis, helping me see the big picture and figuring out how to fit the pieces of work into one coherent thesis. They helped me to understand the scope and the problem of my thesis better.

I would not have been able to accomplish any of these if I had not had the support of my mother Gyanu Maskey, who having gone through tough times in life, has always been there for me. She is responsible for me being able to reach my goals in life and I thank her for that. I would like to thank my father Madhav Maskey. My sister Anju Shrestha, my brother Dr. Sabin Maskey and my uncle Mukunda Joshi have always been there for me. I thank you all. And thank you to Rajaram Shrestha and my lovely nephew and niece Aranab Shrestha and Simran Shrestha.

I was also very fortunate to have the best officemates one can find. Thank you Wisam for all your help. It's been fun 5 years of being officemates. Thank you Arezu for adding that high spirit of energy to the office. Thank you to all my friends in NLP and Speech research group at Columbia University.

Special thanks to my friends who have supported me and with whom I have shared some of the best moments of my life. Thanks to (in alphabetical order) Ani Rudra Silwal, Rashmi Shrestha, Rushil Shakya and Sanjay Shrestha. Also thanks to all my friends from St. Xavier's school, Budhanilkantha school and Birendra Sainik school. Also thanks to all my friends in New York whom I have gotten to know better in the last few years.

Lastly, I would like to thank one individual without whom I would have been just a mindless person wandering the walks of life. My wife, Prerana Shrestha, has been immensely patient with me, has given me everything she has without asking anything in return. I thank her for everything.

To my mother, Gyanu Maskey

Chapter 1

Introduction

Imagine yourself coming back from a hard day at work and you start your television to find a half hour show that has summarized all Broadcast News (BN) shows of your interest in all the available news channels. It has kept track of the news you have already seen on various topics so the show only contains new information. If you have missed the news shows of the last four days, you can reset the program to create a summary of news of all those days. Such a program is not available in reality today. We have seen developments of some information sharing and storage hardware that work with the current news channels; but none of them are "intelligent" enough to automatically create programs that summarize news shows according to users' preferences. In this thesis, we present our automatic BN speech summarization system that would take us a step closer to building such a technology.

BN is one of the most common media in which people obtain news besides

newswire. BN contains one or more speakers presenting, discussing or analyzing current events that are deemed important. A team of producers, screenwriters, audio and video editors, reporters and anchors are involved in the production of BN and they generally follow a standard format of news reporting. Most BN shows contain a sequence of reports on significant current events followed by some commercials, weather, sports and entertainment news. This standard formatting of BN can be useful for automatic processing of BN.

Automatic summarization of BN is a very difficult problem. We have seen publicly accessible automatic summarization systems for newswire such as NewsBlaster [McKeown et al., 2003] and MEAD [Radev et al., 2004]; but we do not have a publicly available BN summarizer. This lack of a BN summarization system could be due to inherent additional challenges that a BN summarizer faces which corresponding text summarization systems do not. Text documents have word, sentence and paragraph boundaries defined which makes it easier to choose the desired processing unit reliably. Speech, however, is one long stream of audio signal with none of these boundaries. Such lack of segmentation makes it difficult to process speech in meaningful semantic units. This problem is typically addressed by employing speech segmentation algorithms.

In order to process speech documents we need to convert speech signals into a sequence of words that is meaningful for users. Automatic Speech Recognition (ASR) engines that convert speech to text have only fair accuracy, even though they have improved in recent years. Poor accuracy hurts speech summarizers because word errors degrade the overall performance of a system that assumes well-formed sentences

as input.

Another problem faced by speech summarization systems is disfluency. Even though humans write well-formed grammatical sentences, when they speak they repeat or repair phrases, insert filled pauses such as 'uh', and 'oh'. Text-trained NLP tools such as parsers and taggers suffer with reduced accuracy on speech documents because of such disfluency. Not having adequate NLP tools that work well with speech, added to other problems of processing speech, makes summarization of speech more challenging than text summarization.

Even though such problems make BN speech summarization harder, there is extra information available in speech that does not exist in text documents. Speech has acoustic information that may help in identifying topic shifts or acoustically significant segments. Spoken documents such as news broadcasts tend to have multiple speakers who play different roles in the broadcasts. Identifying these roles may provide cues to the structure of a broadcast, and can be exploited to deduce the significance of segments for extractive summarization. Also, a speaker's emphasis of particular segments of speech may indicate the significance he or she attaches to that segment.

If we are able to address the additional challenges of speech summarization and extract additional information in spoken documents we may be able to build a robust BN summarization system that would be useful for many applications and devices. For example, handheld devices could stream summarized news instead of full length BN shows where data transmission costs are expensive. A BN summarization system could be tailored to users such as traders and investors who need to be constantly updated on all the "breaking news." Instead of having to watch many different

channels, they could watch automatically generated summaries of breaking news as it occurs. A system that summarizes news from many different news feeds keeping track of news that users have already watched can be useful for everyday BN news viewers as well. If we extended our speech summarizer from BN to other sources of speech such as telephone conversation, voicemail, meeting and chat, we would find even more uses of a speech summarizer.

A successful BN summarization system must perform three functions: computing the significance of segments in a story, finding redundant information, and computing the "right length" for the summary. A summary should not contain irrelevant information, so finding the optimal set of significant segments is very important. On the other hand if we just compile all the most relevant segments together, the summary may contain a lot of redundant information. Such redundancy will decrease the value of the summary to the user. Even if we are able to find significant segments without redundancy, a summary will be of little use if its length exceeds the limit desired by the user. Hence, an ideal summarizer will optimize all of the above functions to produce a *significant, non-redundant* summary of the *desired length*.

In this thesis we present our system, ConciseSpeech, which not only optimizes all the three functions we mentioned in the last paragraph, but also utilizes additional information available in spoken documents. We first present algorithms to handle the additional challenges speech summarization presents, particularly those of segmentation, disfluency and word errors. We then address how we can obtain information from the speech signal that is independent of transcripts. We then show how the relevance, redundancy and length of a summary can be optimized using an algorithm

we have developed. We extend our system to produce user-focused summaries.

When we built ConciseSpeech, we had to address several research problems, the solutions of which we believe make contributions in the advancement of speech summarization research. The main research contributions of the thesis are as follows:

- ConciseSpeech: One of the main contribution of the thesis is the BN summarization system, ConciseSpeech. The system is built in a highly modular fashion that can be extended easily by other developers. The system is fully trainable and includes innovative methods for building a summary, optimizing redundancy, significance and length.

- IE for Spoken Documents: We present methods to extract entities and relations from speech that are useful for summarization. Some relations such as soundbites and soundbite-speakers have not been explored before to our knowledge. The methods show that reasonably accurate IE engines can be developed for spoken documents even though the task involves processing ASR transcripts with errors and unsegmented speech.

- Use of Acoustic Information: We show in this thesis that acoustic information such as pitch, amplitude and speaking rate can be effectively modeled to compute the significance of segments for summarization. In fact, our experiments also show that a summarizer based only on prosodic information is better than a standard lead-summary baseline.[1]

[1]The lead summary baseline is created by extracting the first N sentences of a story for a fixed value of N.

- Disfluency Removal: We also present a novel method for removing disfluency from speech. We use a phrase-based machine translation method based on Weighted Finite State Transducers (WFSTs). Since the algorithm is based on WFSTs it can be easily integrated into many speech processing systems that are also based on WFSTs with a simple composition operation.

- NE-NET: We also built a novel knowledge base, NE-NET, that can relate NEs. NE-NET can be queried with any NE and returns a set of related NEs within a user-specified degree. We show that NE-NET can be used to improve information retrieval.

1.1 Related Work

There has been considerable work on IE and summarization for text documents. Some of the techniques used in text summarization are relevant for speech summarization because one possible way to summarize speech is by using text summarization techniques on transcribed speech. Hence we describe some relevant text summarization methods first.

Text summarization techniques can be broadly categorized by purpose (indicative, informative), type (single or multiple documents), output method (extraction, generation), level of processing (words, phrases, sentences), and approach (corpus, discourse, knowledge). Among these techniques, let us first describe some of the text summarization approaches that are relevant for the speech summarization techniques that we have proposed in this thesis.

The corpus-based approach uses a corpus of manually built summaries to train a feature-based model that predicts which segment should be included in the summary. Some examples of corpus-based approaches are [Kupiec et al., 1995, Hovy and Lin, 1997, Witbrock and Mittal, 1999]. [Kupiec et al., 1995] marks the development towards a trainable corpus-based summarizer. In this paper, [Kupiec et al., 1995] propose a Bayesian classifier that computes the probability of a sentence being included in the summary using a set of features that are trained on 188 full-text/summary pairs. On the other hand, discourse-based approaches exploit the discourse structure of a document for generating a summary. [Marcu, 1997a] proposes one such discourse-based method. Marcu's method [Marcu, 1997a] differs from corpus-based methods because he did not use a corpus of any kind. Instead, he built a Rhetorical Structure Theory (RST) trees using a rhetorical parser for unrestricted text that exploits cue phrases. Rhetorical parsers are prone to word errors in the transcript, which has led us to explore corpus-based methods.

Similar to text summarization output, speech summaries can be a concatenation of significant sentences extracted from the document or they can be sentences generated using text generation techniques. If we examine the text summarization literature we note that the extraction method has been successfully used in many domains and applications. But there are also some systems that successfully generate sentences for the summary. [McKeown et al., 2001] and [Witbrock and Mittal, 1999] both generate their sentences in their summary. [Witbrock and Mittal, 1999] use a naive sentence generator which does surface realization simply using bigram probabilities. [McKeown et al., 2001] use a more sophisticated text generation strategy that takes

account of sentence ordering and intersection. [McKeown et al., 2001]'s method of clustering themes and finding intersecting phrases to be selected by content planner allows them to select phrase-level structures and combine them for the summary. But using generative techniques similar to [McKeown et al., 2001] poses many challenges in the speech domain. First, we will lose the originality of the speaker's voice if we generate a summary. Second, text generation methods that rely on extensive syntactic information may perform worse for speech transcripts. These reasons are motivating factors for choosing extraction-based methods for summarizing spoken documents. Other methods, such as summarization based on lexical chains [Barzilay and Elhadad, 1997], can potentially be useful for speech transcripts as well. But the possible degradation of lexical chain computation due to errorful transcript is again a potential problem.

Some text-summarization techniques, especially extractive corpus-based methods, have been used for summarizing spoken documents of different genres. Speech summarization research has mainly focused on three different genres: BN [Hori, 2002, Maskey and Hirschberg, 2005, Christensen et al., 2004], voicemail [Koumpis and Renals, 2000, Zechner and Waibel, 2000] and meetings [Galley et al., 2004, Murray et al., 2005a].

Most of the proposed methods extract segments (phrases, sentences) based on acoustic and lexical features, and combine the selected segments to generate a summary. [Koumpis and Renals, 2000, Christensen et al., 2004] report encouraging results using lexical and prosodic features to extract relevant sentences. [Maskey and Hirschberg, 2003, Maskey and Hirschberg, 2005] have shown that sentence extraction

for summarization based on additional structural and discourse information can be useful. One of the problems with sentence extraction based methods is that sentence segmentation of speech such as meetings can be very poor. [Galley et al., 2003] addresses this problem by segmenting meetings into topics and extracting the topics using lexical cohesion techniques.

A few methods compress the sentences or utterances instead of extracting segments. [Hori et al., 2002] remove redundant, irrelevant words and phrases in BN sentences using four types of scores (word, linguistic, confidence and concatenation). Recent work by [Hori et al., 2003] also removes irrelevant terms but uses a Finite State Transducer to translate irrelevant terms to null. The voicemail summarization system of [Zechner and Waibel, 2000] removes disfluencies and uses the Maximal Marginal Relevance algorithm [Carbonell and Goldstein, 1998] to generate a summary. [Koumpis and Renals, 2000] use a filtering approach by removing ASR errors and disfluencies to compress sentences for voicemail summarization.

Instead of viewing summarization as a two step process of transcribing speech and then summarizing it, [Banerjee and Rudnicky, 2006] use an interactive method for summarizing meetings. They use 'smart notes,' which meeting participants enter. These notes are collected and presented as a summary. Another user-interaction based method is proposed by [He et al., 1999] which asks users to identify important segments and stores user preferences. As more users use the system, preference histories of users are combined to produce summaries for new users. One of the main disadvantages of such an approach is that it is heavily dependent on the number of users that view the same document, which is not viable for standalone application

like ours.

Most of the speech summarization algorithms mentioned above use statistical techniques. Also, the methods described above assume the availability of boundaries that divide speech into smaller segments. Even though it is fair to assume that word segmentation is available for speech because most ASR systems provide word boundaries, it is important for the speech to be segmented into semantically meaningful chunks such as phrases, sentences, topics or stories. [Shriberg et al., 2000] chunk speech input into sentences using prosodic features. [Hirschberg and Nakatani, 1998] have shown that acoustic information can be used for identifying topic boundaries and [Rosenberg and Hirschberg, 2006] used acoustic and lexical information to detect story boundaries in BN. We will use [Rosenberg and Hirschberg, 2006]'s story segmentation method for our BN.

Information Extraction (IE) methods are relevant to our task of speech summarization as we extract entities and relations from BN. IE methods have been explored widely for use in both text and speech domains. IE methods have obtained significant attention due to DARPA-funded projects such as GALE, EARS, TIDES, TREC and TDT [LDC, 2002].[2] The tasks defined in these various projects include generating a rich transcription that contains capitalization, punctuation and speaker diarization, and detecting disfluency and named entities (person, people, organization). Many of these tasks have been successfully addressed by [Snover et al., 2004a, Honal and Schultz, 2003, Shriberg et al., 1997] for disfluency, [Miller et al., 1999] for named entities and [Allan et al., 1998] for topics. Most of the techniques have employed

[2]More information on these projects are available in NIST website http://www.nist.gov/speech/

ML techniques using ASR transcripts and acoustic features. The research on these topics suggests that ML techniques are robust for speech but degrade with poor ASR performance. For example, [Miller et al., 1999] have found a significant correlation between ASR Word Error Rate (WER) and the accuracy of named-entity detection.

The work described above on text and speech summarization and on IE can be used to assess the potential advantages and disadvantages of various ways we can summarize speech. As we mentioned previously, ASR transcripts are errorful and this degrades the accuracy of many NLP tools. Also, text generation tools may have difficulty processing disfluency and word errors in transcripts and generating sentences that capture the original intent of the source. Thus, extraction-based approaches may offer a better solution for speech summarization than generation-based summarization.

1.2 Structure of the Thesis

The thesis is divided into nine chapters including this chapter. In this chapter we introduced the concept of BN speech summarization and explained the related work on summarization and IE of spoken data. There is no standard speech summarization corpus provided by LDC or any other organization so we have created our own BN speech summarization corpus. We discuss our data annotation process and the analysis of corpora in Chapter 2. After we prepared our data we built a collection of feature extraction modules which we describe in Chapter 3.

With the data prepared and features extracted, we describe our first approach to

speech summarization. One of the first questions that arises when addressing speech summarization is "How is speech summarization different from text summarization, i.e., is speech summarization just a summarization of ASR transcripts?" We explore answers to this question in Chapter 4. We next build modules to address the challenges particular to speech summarization and describe them in Chapter 5. We then build IE engines to extract useful entities and relations in BN spoken document and discuss them in Chapter 6.

After implementing algorithms to address challenges and extract information particular to BN, we combine all of these modules to build our BN speech summarization system, which we present in Chapter 7. In this chapter we also present an algorithm to optimize significance, redundancy and length of segments to generate a summary. ConciseSpeech is extended to handle user queries as described in Chapter 8. We also present a knowledge resource we have designed and built, NE-NET, which improves the information retrieval task for users queries. Finally, we conclude in Chapter 9 where we discuss the contributions and limitations of our thesis.

Chapter 2

Corpus, Annotation and Data Analysis

We conducted our experiments on a corpus of BN shows. Our corpus consists of a subset of shows in the Topic Detection and Tracking corpus(TDT) included in the Global Autonomous Language Exploitation (GALE)[1] corpora provided by the Language Data Consortium [LDC, 2006]. There is no standard corpus available for BN summarization experiments. So, we created our own BN summarization corpus by annotating select shows from the TDT and GALE corpora.[2]

[1]GALE is a large multi-institution large project undertaken by DARPA for automatically finding relevant answers for a given set of question types. The GALE evaluation task consists of recognition, translation and distillation. More information is available at http://www.nist.gov/speech/tests/gale/index.htm.

[2]Our annotation is publicly available at http://www.cs.columbia.edu/~smaskey/thesis.

2.1 Corpora

We have created two corpora, which we call Columbia BN Speech Summarization Corpus I (CBNSCI) and Columbia BN Speech Summarization Corpus II (CBNSCII). Both corpora consist of CNN Headline news shows collected over the period of 1998 to 2003 with annotations for summarization and IE experiments.

2.1.1 Columbia BN Speech Summarization Corpus I

CBNSCI consists of 216 stories from 20 CNN Headlines News shows from TDT-2 [LDC, 2002] corpus. Each news story has a corresponding summary. This summary was created by an annotator extracting significant sentences. We describe the annotation process in detail in Section 2.1.1.1. The corresponding audio files for these 20 shows consist of 10 hours of speech data. The corpus contains manually transcribed and (some) close captioned transcripts for each of the BN shows. The corpus also contains ASR transcripts provided by the Dragon ASR engine. [3] We did not have access to ASR lattices so we use the best ASR hypothesis for our experiments. Manual transcripts provided by LDC for these shows contain story segmentation and topic labels while the ASR transcripts include word boundaries and pauses. Besides summarization annotation, 65 shows were further manually annotated with named entities, headlines, anchors, reporters, interviewees, soundbites and soundbite-speakers following a labeling manual we prepared.[4] We will explain our definition of soundbites,

[3]The Dragon ASR engine is a large vocabulary ASR system now owned by Nuance.

[4]The labeling manual can be downloaded from www.cs.columbia.edu/~smaskey/thesis. The labeling manual has been used by other research groups for identifying anchors and reporters.

headlines, interviewees in Section 3.3.

2.1.1.1 Annotating Extractive Summaries and Relations

We annotated CBNSCI for extractive summarization. One annotator extracted summaries for the stories in CBNSCI. The annotator was not given explicit instructions on the desired length of the summaries.

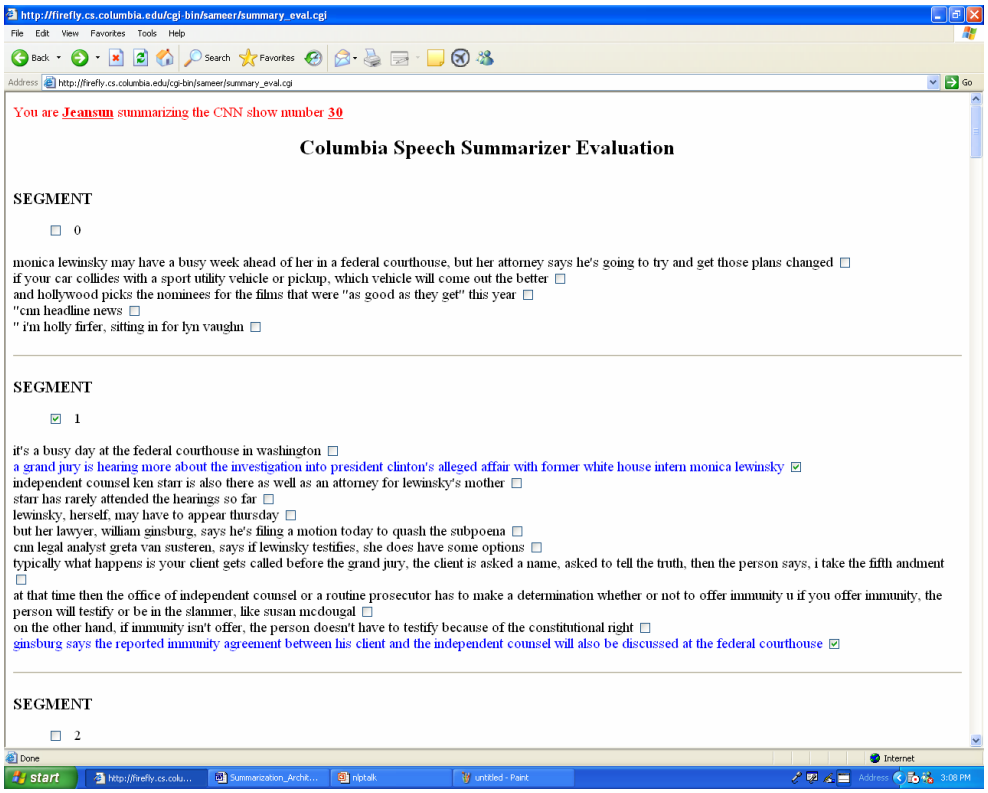


Figure 2.1: Web Interface for Summarization Annotation

A sample of the web-interface provided to summarization annotators is shown in Figure 2.1. We note that the annotators can choose which story they want to summarize by selecting the SEGMENT (story) number and then selecting the sentences of the story by clicking on the buttons at the end of each sentence. Whenever an

annotator clicks the radio button at the end of the sentence, the text of the sentence is highlighted to ensure that the annotator has actually selected the sentence he/she believes has been selected. After the annotator has finished selecting sentences to be extracted, the annotator is sent to a confirmation page where he/she can read both the story and summary on the same page. The annotator can change or confirm the summary. A sample of this step of the interface is shown in Appendix A.2. All the summaries are stored in a MySQL database.

We also annotated the CBNSCI corpus for relations and entities such as Speaker Roles, Soundbites, NEs, Headlines, and Sign-Ons. Soundbites are statements in BN that can be directly attributable to a person who is not an anchor or a reporter. We call the speaker of a soundbite a soundbite-speaker. Headlines are the introductory statements made by an anchor in BN at the beginning of the show. Signoffs are the final comments made by anchors at the end of the show. Annotators were provided with manual transcripts, a labeling manual and a labeling tool, dLabel.[5]

dLabel is a Java based labeling tool that can be used to annotate sgml, xml or text files. A screenshot of the tool can be seen in Figure 2.2. The tool lets an annotator highlight a segment of BN transcript and click one of the buttons to put start and end tags around the selected text. The button that was clicked is used as reference to decide on what kind of tags are to be used to surround the selected text. dLabel can also be used to parse and update xml files and has many customizable functions from buttons to fonts.

[5]dLabel is available through a link at http://www.cs.columbia.edu/~smaskey/thesis.

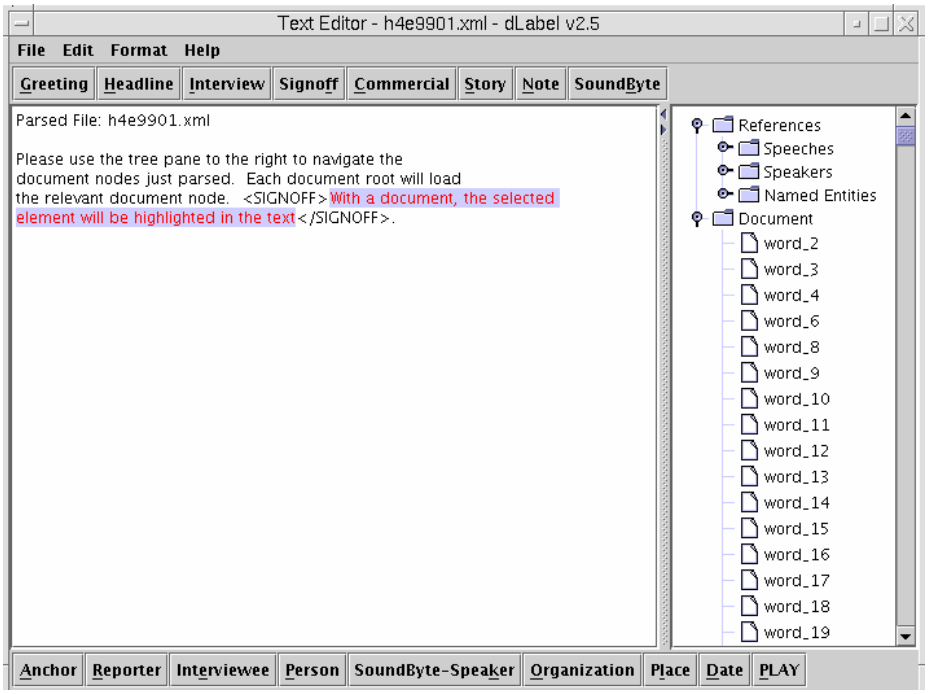


Figure 2.2: dLabel Ver 2.5, Annotation Tool for Entities and Relations in BN

2.1.2 Columbia BN Speech Summarization Corpus II

CBNSCII consists of 20 CNN Headline News Shows from the TDT4 corpora. Besides the manual and ASR transcripts already made available for the TDT4 corpora, we also had access to ASR transcripts provided by SRI as part of the GALE project. These transcripts contained sentence boundaries and phone outputs for words. These were further enriched with story boundaries using a segmentation tool provided by [Rosenberg and Hirschberg, 2006]. We also had manual transcripts for these shows that contained manual annotations for turn boundaries, topic labels, and story boundaries. Manual transcripts were annotated for speaker roles as well. Speaker roles are the roles of the speakers in BN show such as anchors and reporters. 447 stories of these shows were also manually summarized by annotators, who produced both ab-

stractive and extractive summaries. On average each show is 216 sentences long with slightly fewer than 10 sentences in each story, and a wide variation of story lengths (ranging from a few sentences to more than 50 sentences).

2.1.2.1 Annotation for Abstractive, Extractive Summaries

The CBNSCII corpus was annotated for abstract (manually generated summaries), and extractive summaries. One of the main differences between CBNSCI and CBNSCII is the method used for generating summaries. Three native English-speaking annotators, were asked to write a summary for each story in their own words but using words from BN whenever possible. Unlike CBNSCI, where we did not provide any length requirements, for CBNSCII we asked the annotators to generate a summary approximately 30% of the length of the original story. When they finished writing abstracts, we asked them to extract sentences from the story that supported their abstract. We provided them with the updated dLabel having *INSUMMARY* buttons to highlight the sentences to be extracted for the summary.

Of the three annotators who generated abstracts, two of them also generated extractive summaries. Even though we had asked annotators to produce summaries with a summarization ratio of 30%, they produced summaries that were of higher summarization ratio for stories that were very short (4 to 5) sentences. This could be explained by the difficulty of choosing sentences for a very short story. If a sentence has 5 sentences choosing only 2 sentences amounts to a summary of 40%. One of the annotators was also asked to annotate the stories with speaker roles, soundbites, interviewees, soundbite-speakers and commercials.

We computed the overall agreement between two annotators using Kappa statistics. It is difficult to compute the agreement on abstractive summaries so we compared the labels provided by the annotators for the extractive summaries. We obtained a Kappa score of 0.4126. This shows that the annotators agree well on the sentences they extract for summaries but they don't produce exactly same summaries.

2.2 Pre-processing of Corpora

After the completion of the annotation process, we had to perform a sequence of pre-processing steps on the corpora to make it usable for our summarization and IE experiments. We describe these pre-processing steps in the following sections.

2.2.1 Aligner

Our annotation of turn boundaries, speaker roles, summary, commercials and soundbites was done on manual transcripts. In order to build an automatic summarizer based on ASR transcripts, we need to train our model on ASR transcripts which do not have these annotations. We can easily run some of the automatic annotators on ASR transcripts such as POS taggers but we cannot obtain sentence summary labels without asking human labelers to re-label ASR transcripts. We did not want to ask human labelers to annotate ASR transcripts because ASR accuracy changes, thus making our models and annotation obsolete whenever there is an improvement of the ASR engine. In order to obtain manual annotation for ASR transcripts, we aligned manual and ASR transcripts at the word level using the Minimum Edit Dis-

tance (MED) algorithm, which has been explained in detail in [Navarro, 2001]. We built an aligner that would align manual and ASR transcripts. We tested different combinations of insertion, substitution, match and deletion costs. We used word level match cost of 0, substitution cost of 4 and insertion and deletion cost of 3 in our MED algorithm which gave us the best alignment.[6]

Using the aligner we were able to transfer the annotations made on manual transcripts to the ASR transcripts. One of the problems we face in this transfer process is themis -match of boundaries. Manually or automatically labeled boundaries such as sentence, turn and story boundaries may occur at different points in manual and ASR transcripts. When we transfer a label for sentence M_i in the manual transcript to sentence A_k in the ASR transcript, sentence boundaries may not match as i may not equal k. In such cases we assign the labels to ASR sentences with the closest matching manual sentence. An example of aligned manual and ASR transcripts with the annotation is shown in Appendix A.1.

Another problem that we face in building an automatic summarizer is using manually generated summaries to train an extractive summarizer. Our statistical model for the summarizer is trained to extract segments by either classifying or ranking sentences (segments) by significance. In order to train such a model we need a summary label for each of the sentences in the story. Abstractive summaries have no notion of segments from the original story, as the sentences generated in the summary by humans are in their own words. Recall that in our instructions to the annotators, we had asked the annotators to use the words from the original story whenever possible.

[6]The alignment software is available at www.cs.columbia.edu/~smaskey/thesis.

Such words allow us to associate the words in the summary with the words in the original story by aligning the human generated summary sentences with the sentences of the story. We used the aligner to align the summaries with the stories. Then we computed the labels for the sentences in the story by computing the degree of match between the words of the sentence in the story with the words in the summary. Thus we can label all ASR sentences with summary label scores.

The use of the aligner and transfer of manual annotations to ASR transcripts allow us to explore the effects of various combinations of manual and automatic annotation on ASR transcripts to identify the important annotations for summarization.

Chapter 3

Feature Analysis and Extraction

Many of our summarization modules are trained using Machine Learning (ML) methods. In this chapter we explain some of the features we use to train our ML algorithms for our summarization and IE experiments. Although different problems may require different sets of features, some features are useful for most of our experiments. We extract features that can be roughly categorized into four broad classes namely, *Lexical, Acoustic/Prosodic, Structural* and *Discourse* features. We describe the most important of these feature types in this chapter.

3.1 Lexical Features

The lexical features we use have been motivated by features developed in previous research for text and speech summarization [Schiffman et al., 2002, Radev and McKeown, 1998, Christensen et al., 2004]. Some of the lexical features we use are listed in Table 3.1.

Sub-Feature Type	Feature
Named Entity	Person Name
	Organization Name
	Location Name
Frequency	Term Frequency (TF)
	Inverse Document Frequency (IDF)
	TF·IDF
Text Similarity	Cosine Similarity
	Centroid Similarity
	Simfinder similarity
Word Types	Is this a Function Word?
	Is this a Content Word?
Part of Speech Tag (POS)	Type of POS

Table 3.1: Lexical Features

- *Named Entity*: We have seen in our corpus that content-laden sentences have a high percentage of NEs. For user focused summarization, NEs help to weight the sentences according to the number of NEs that users may be interested in. In order to extract NE based features, we first identify NEs. We identify *person, organization* and *location* names. We compute *Total NEs, Total sub-types of NEs*, and binary features such as *"Does the sentence contain NE on topic?"*.

- *Frequency*: Many summarizers available today use the frequency of words as one of the features to compute the significance of a sentence. The word frequencies

provide clues to their significance and generality. Words such as "Kathmandu" may occur less often than "apple" and can potentially be more important if the story is about Nepal. A frequency measure of words known as TF·IDF proposed by authors [Robertston and Jones, 1976] has been widely used for summarization. We use this measure as one of our features. TF·IDF (Term Frequency x Inverse Document Frequency) proposes that a term is significant if it rarely occurs in a corpus but occurs many times in a given document. TF·IDF is computed using the following equation.

$$TF \cdot IDF = TermFreq * InvDocumentFreq \tag{3.1}$$

where

$$TermFreq = \sum n_i \tag{3.2}$$

where n_i the number of times the i_{th} term occurred in document d

$$InvDocumentFreq = log(\frac{N}{\sum n_{ij}}) \tag{3.3}$$

where N is total number of documents in the corpus and $\sum n_{ij}$ is the total number of documents that contain the term i. Besides the word frequency we can also obtain frequency counts of other terms in the story such as frequency of pronouns, verbs which have been shown to be useful as well.

- *Text Similarity* Text similarity scores are important features for summarization because these scores help to identify redundant sentences in the story.

Various text similarity metrics are available to identify similar sentences. We compute the similarity between sentences by computing cosine similarity, and Longest Common Subsequence (LCS). Cosine similarity between two documents or spans of text is defined as follows:

$$Similarity(X, Y) = \frac{\vec{X} \cdot \vec{Y}}{||\vec{X}|| \cdot ||\vec{Y}||} \tag{3.4}$$

where $\vec{X}$ and $\vec{Y}$ are word vectors for sentences X and Y, and $\vec{X} \cdot \vec{Y}$ is the dot product between them.

- *Part of Speech (POS)* POS tags are useful because they can be used to identify adverbial and prepositional phrases. POS can also be used to identify pronouns. Some of the POS features that were useful for our purpose were count of various pronouns in the sentence, and counts of noun phrases. In order to obtain POS, we used a POS tagger provided by Ratnaparkhi [Ratnaparkhi, 1996].

3.2 Acoustic/Prosodic Features

Sub-Feature Type	Feature
Intensity	meanDB, maxDB, averageDB
Pitch	F0 max, min, average, slope
Duration	segment duration (secs)
Speaking Rate	number of voiced frames/total frames

Table 3.2: Acoustic/Prosodic Features

Acoustic/Prosodic features have been examined in summarization [Inoue et al., 2004] and information extraction tasks from speech documents [Shriberg et al., 2000]. The motivation for using prosodic/acoustic features for these tasks is based on speech prosody research [Hirschberg, 2002] that suggests that humans use intonational variation such as expanded pitch range, phrasing or intonational prominence to mark the importance of particular segments in their speech. Our acoustic features include features mentioned in [Inoue et al., 2004, Christensen et al., 2004] as well as some new features that have not been explored before. Some of the acoustic/prosodic features we use are shown in Table 3.2.

- *Intensity*: We extract the intensity (amplitude) of the speech signal with the motivation that speakers may speak louder if they want to emphasize significant segments of speech. We extract *maximum and minimum intensity* of the signal for a given sentence. A large difference between maximum and minimum intensity may signify significant information. We also extract *average and slope of the intensity*. Averge intensity can be used to normalize the intensity variation between speakers so that we can take account of variation between loudness due to change in speakers but not due to the signifiance of the segment being spoken. Slope of intensity means the difference between loudness in the beginning of speech to the end of speech. A high value of slope in intensity will signify a sudden change in the speakers voice potentially signifying an important transition in speaker's discourse.

- *Pitch*: There is considerable evidence that topic shifts may be marked by

changes in pitch [Hirschberg and Nakatani, 1996] and new topics or stories in BN are often introduced with content-laden sentences which, in turn, are often included in a summary. Inclusion of sentences at the beginning of a topic in a story occurs often in BN stories because BN stories are shorter in length than newswire stories. We extract *maximum, minimum, mean* and *slope* of pitch. Mean pitch can be used for normalizing pitch differences between speakers of different genders while a maximum and minimum pitch and their difference can provide important clues on the effective emphasis a person may be putting while he/she is speaking.

- *Duration*: Sentence duration represents the length of the sentence in seconds. Our motivation for including this feature is two-fold: Very short segments are not likely to contain important information. On the other hand, very long segments may not be useful to include in a summary, simply so as not to produce too long summaries. This length feature can accommodate both types of information. We obtain the length of a sentence by subtracting the end from the start time for the sentence.

- *Speaking Rate*: We extract speaking rate (the ratio of voiced/total frames) with the motivation that a sudden change in speaker rate may signify a change in emphasis on the segment of speech. We can also roughly estimate speaking rate by dividing the duration of a sentence by the number of words in the sentence. Some people inherently speak fast while some speak slow, hence we need to normalize speaking rate accordingly when we use speaking rate for identifying

significant segments of speech.

We extract the acoustic/prosodic features using the Praat tool [Boersma, 2001]. Praat allows us to provide sentence boundary timestamps as one of the data tiers as shown in Figure 3.1. We obtain the segment boundaries from the ASR transcripts. We can see the features being extracted in Figure 3.1, where the top tier shows the speech signal, the middle tier shows the extracted features and the bottom tier shows the sentence boundaries.

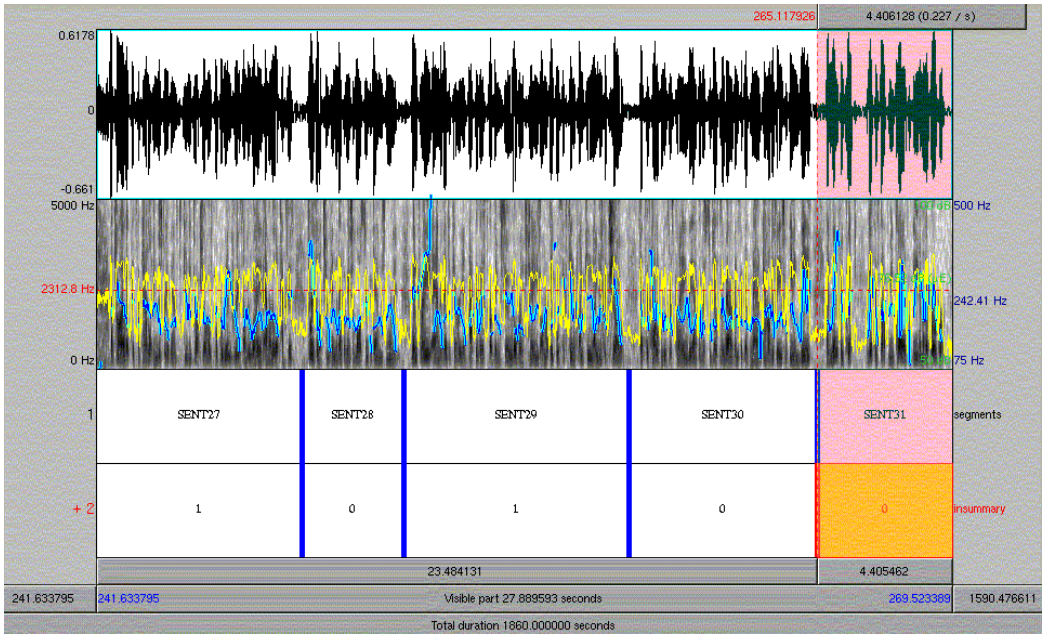


Figure 3.1: Acoustic Feature Extraction with Praat

Sub-Feature Type	Feature
Speaker Role	Anchor, Reporter, Other
Position	Sentence, Speaker position
	Relative positions of sentences and speakers
Turns	Turn length, Turn position
Boundaries	Sentences
	Turn
	Story
Segment Type	Headline
	Sign-on
	Sign-off

Table 3.3: Structural Features

3.3 Structural Features

BN programs exhibit similar structure for shows from the same news channel. Each show usually begins with an anchor or anchors reporting the headlines, followed by the actual presentation of those stories by the anchor and reporters. Programs are usually concluded in the same conventional manner. We call the features which rely upon aspects of this patterning and from the overall structure of the BN *structural features* [Maskey and Hirschberg, 2003], comparable to [Christensen et al., 2004]'s *style features*. The structural features we investigated for our study are listed in Table 3.3.

- *Speaker Role*: Speaker role describes the role of the speaker in BN - *anchor,*

reporter or *others*. Anchors tend to say content-laden sentences in the beginning of the turn when they introduce a story. If we know which turns are spoken by anchors this can provide useful clues for a summarizer for selecting sentences. In order to extract the speaker's role as a feature, we first build a classifier that classifies turns by speaker roles. We explain this classification process in Section 6.1.

- *Sentence, Turn and Story Positions*: Position information is vital for summarization because sometimes story highlights are presented at the beginning of the story. We extract *position of sentence, turn and story* in BN. We also extractive *relative position* of sentences within a turn, relative position of turns within a story and relative position of story within the show.

- *Previous and Next Turn Lengths*: The features *length of previous segment* and *length of next segment* help us differentiate between planned, unplanned, formal and informal speech in news broadcasts. We compute the length of the segment by counting the number of words in the segment and also by computing the total time (seconds). For example, dialogs between anchor and reporter or reporters' interviews are generally signaled by a sequence of short segments. These features thus allow us to give less weight to such dialogs, since they seem less likely to contain vital information for summarization.

- *Segment Type*: Some segments of BN are characterized by the role they play in BN. These segments are *Headlines, Teasers, Commercials*. Headlines are the first few turns by anchors where they present the summary of the headline

stories to be presented in the BN show. Teasers are highlights of the new story that is about to follow after the commercial breaks. We identify these segments by building separate specialized detection modules which we explain in detail in Chapter 6.

3.4 Discourse Features

Sub-Feature Type	Feature
Given-New	Given-Newness Score

Table 3.4: Discourse Features

Discourse features, such as [Marcu, 1997b]'s discourse trees, which model the rhetorical structure of a text to identify important segments for extraction, have been explored for summarization. We have explored a different discourse feature, by computing a measure of 'givenness' in our stories. Following [Prince, 1992], we identify 'discourse given' information as information which has previously been evoked in a discourse, either by explicit reference or by indirect (in our case, stem) similarity to other mentioned items. Our intuition is that given information is less likely to be included in a summary, since it represents redundant information. Our *given\new* feature represents a very simple implementation of this intuition and proves to be a useful predictor of whether a sentence will be included in a summary. This feature is a score that ranges between -1 and 1, with a sentence containing only new information receiving a score of '1', and a sentence containing only 'given' information receiving '-1'. We calculate this score for each sentence by the following equation:

$$S(i) = \frac{n_i}{d} - \frac{s_i}{t-d} \tag{3.5}$$

Here, n_i is the number of 'new' noun stems in sentence i, d is the total number of unique noun stems in the story; s_i is the number of noun stems in the sentence i that have already been seen in the story; and t is the total number of noun stems in the story.

The intuition behind this feature is that, if a sentence contains more new noun stems, it is likely that more 'new' information is included in the sentence. The term n_i/d in the equation 3.5 takes account of this 'newness'. On the other hand, a very long sentence may have many new nouns but still include other references to items that have already been mentioned. In such cases, we would want to reduce the given-new score by the 'givenness' in the sentence; this givenness reduction is taken into account by $\frac{s_i}{t-d}$.

Chapter 4

From Text to Speech Summarization

When we built our speech summarizer, we asked ourselves a few questions that would provide important clues for speech summarization. Is it sufficient to use or extend text summarization techniques for use on speech transcripts? What makes speech summarization different from text summarization? Should the summary output be in speech or text or both? Should speech summarization be based only on the speech signal? Can we build a speech summarizer just based on prosody? In this chapter we present some of the answers to these questions that we have found in our research. We performed a set of experiments that would help us illustrate the answers to these questions. We first experimented with using text summarization technique on speech transcripts. Then we used additional information available in the speech signal such as acoustic and structural properties. These experiments provided important information regarding the usefulness of information that cannot be exploited in text sum-

marization. We further explored the challenges particular to speech summarization that text summarization does not have to address.

4.1 Extractive Speech Summarization Using Transcripts

The goal of single document text or speech summarization is to identify information from a text or spoken document that summarizes, or conveys the essence of a document. EXTRACTIVE SUMMARIZATION identifies important portions of an original document and concatenates these segments to form a summary. How these segments are selected is thus critical to summarization adequacy.

We use ML technique to identify sentences to be extracted for the speech summary, similar to the approach of [Kupiec et al., 1995] for text summarization. The classifier is trained using Summarization Corpus I which has binary summary labels (inclusion or exclusion) for each sentence in a story. The trained model classifies whether each sentence should be included or not in the summary. We concatenate the sentences predicted by the model labeled for inclusion to produce the final summary.

Since we are building a speech summarizer based on manual transcripts, we extracted lexical features from the transcripts to train our ML model. We used a subset of the lexical features described in Section 3.1 which includes counts of **person names**, **organization names**, and **place names** for each sentence. We also included the **total number of named entities** in a sentence as a feature, in addition to the **number of words in the current, previous and following sentence**. Our

feature selection algorithm selected the total number of NEs and the number of words in the sentence as particularly useful features for predicting sentences to be included in a summary. We trained our model with these features and binary summary labels using a Bayesian Network learning algorithm. We explain the details of the Bayesian Network algorithm in Section 7.1.2.2.

We then compared the summaries generated by our model with baseline summaries. We generated baseline summaries by taking the first 23% of the sentences of each story in BN. We chose the length of the baseline summary by computing the average length of the annotated summaries in the corpus. It is standard to take the first few sentences as a baseline for summarization tasks. For our purpose this is a very strict baseline because the stories are short compared to newswire documents, and headlines (significant sentences for summaries) are introduced in the beginning of each story. Our summarizer based on lexical information only obtained an f-measure of 0.487, as shown in Figure 4.1

We see in Figure 4.1 that we obtain an improvement of 5.9% over the strict baseline by using text summarization techniques on the transcripts. This is an encouraging result and shows the promise for text summarization techniques for speech summarization. To confirm that our results are unaffected by the choice of classifier, we also computed ROC curves for the classifiers we tried. The area under the curve (AOC) of the ROC curve computes the 'goodness' of a classifier; the best classifier would have an AOC of 1. For the classifiers we examined, we obtained an AOC of 0.771 for Bayesian Networks, 0.647 for C4.5 Decision Trees, 0.643 for Ripper showing that Bayesian Networks are indeed the right choice for training the model.

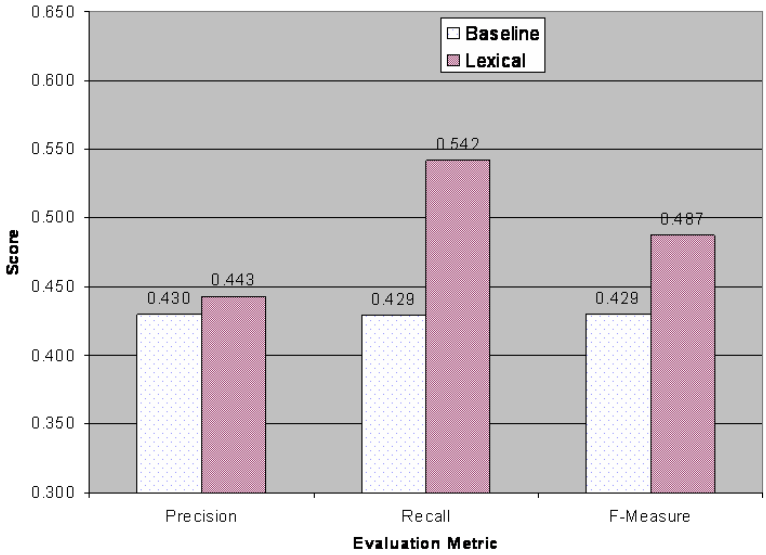


Figure 4.1: Summarizing Speech with Transcripts Only

We have to note that we are using manual transcripts for the above experiment, and the same experiment on ASR transcripts will produce lower accuracy. The use of manual transcripts allows us to estimate the ceiling of performance we can obtain in the above experimental conditions. We explain our experiments with ASR transcripts in later chapters. Also, we have not used the acoustic/prosodic information available in speech signal. Acoustic/prosodic information should be useful for finding significant segments of significant speech for summarization, and we show this is the case in the following section.

4.2 Using Additional Information Available in Spoken Documents

4.2.1 Adding Acoustic/Prosodic Information

We mentioned in Section 3.2 that humans use intonational variation for emphasizing different parts of speech [Hirschberg, 2002]. In BN, we note that change in pitch, amplitude or speaking rate may signal differences in the relative importance of the speech segments produced by the speakers of BN. In order to test the effectiveness of using acoustic/prosodic information, we extracted acoustic features for the same set of sentences in CBNSCI that were used in the previous section 4.1. The feature set includes **speaking rate** (the ratio of voiced/total frames); **F0 minimum**, **maximum**, and **mean**; **F0 range** and **slope**; **minimum, maximum**, and **mean RMS energy** (minDB, maxDB, meanDB); **RMS slope** (slopeDB); **sentence duration** (timeLen = endtime - starttime) as described in detail in Section 3.2.

After extracting the acoustic/prosodic information we re-trained our previous Bayesian Network model. The new model includes the additional acoustic information plus the lexical information. We tested our new model using 10-fold cross validation on the same data set. Figure 4.2 shows that there is additional information in the speech signal which is not captured by the words. We see an improvement of 3.7% in f-measure for using this additional acoustic information. This is a very encouraging result as it shows that a speech summarizer is not just an extension of a text summarizer of transcripts; using additional intonational information as well

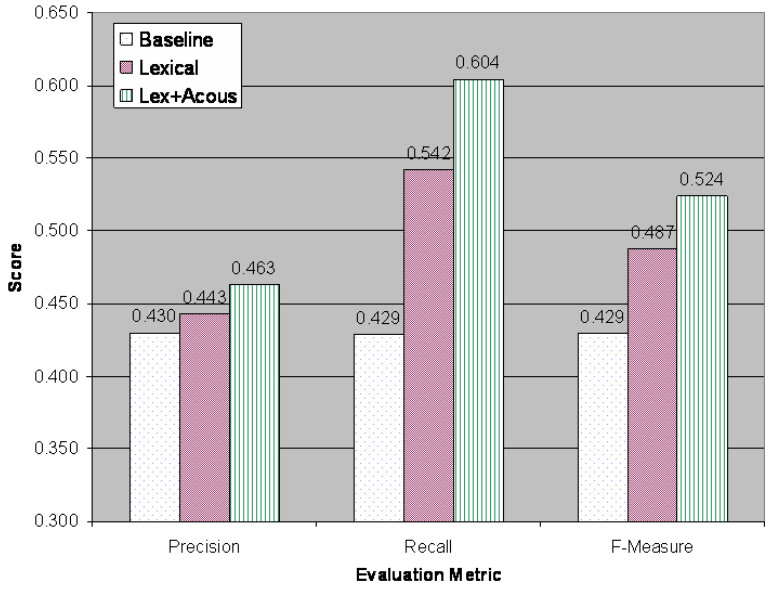


Figure 4.2: Adding Acoustic/Prosodic Information to Transcript Based Summarizer improves the summarizer.

4.2.1.1 Speech Summarization Based on Acoustic Information Only

We saw in the section above that the use of acoustic information improves the transcript-based speech summarizer; but can we summarize speech without using the transcript at all, that is, *knowing absolutely nothing about what is being said, can we identify significant segments?* In order to explore this somewhat radical question, we performed the following experiment — we re-trained our ML model with the same settings using only acoustic/prosodic described in Section 4.2. This model essentially predicts if a sentence should be in the summary or not by simply using pitch, intensity, duration and speaking rate as features for the model. The results in Table 4.1 suggest that it is possible to *summarize speech without transcripts almost as well as*

Type	Precision	Recall	Fmeasure
Baseline	0.430	0.429	0.429
Acoustic Only	0.443	0.495	0.468
Lexical Only	0.443	0.542	0.487

Table 4.1: Summarizing Speech Without Transcripts

using transcripts alone. The significant improvement of 3.9% in f-measure over the baseline with the use of acoustic/prosodic features shows that speech can indeed be summarized fairly well by modeling only such information – information not available to text summarization algorithms. The acoustic-only model does only slightly worse than summarization based on lexical only information.

Intuitively, one might imagine that speakers change their amplitude and pitch when they believe their utterances are important, to convey that importance to the hearer. So a hypothesis that "the importance of **what** is said correlates with **how** it is said" should be true. Otherwise we would expect the sentences that our lexical features included in a summary to be different from those predicted for inclusion by our acoustic/prosodic features. We computed the correlation coefficient between the predictions of these two different feature-sets. The correlation of 0.74 supports our hypothesis.

4.2.2 Exploiting the Structure of Broadcast News

We saw in previous sections that using acoustic information improved a text based summarizer. Can we do more to improve upon the lexical and acoustic/prosodic

summarizer? BN programs, especially the same series, show similarity in 'structure'. We would like to exploit this structure to improve our summarizer for BN. In order to show that using structural features further improves summarization, we performed the following experiment.

For the same set of sentences from Summarization Corpus I, which we used for our previous experiments, we extracted a subset of structural features described in Section 3.3 such as **normalized /sentence position in turn**, **speaker role**, **next-speaker role**, **previous-speaker role**, **speaker change**, **turn position in the show** and **sentence position in the show**. We added these structural features to the feature vector that already contained lexical and acoustic/prosodic features and re-trained our Bayesian Network model. We tested on the same dataset that was used for previous experiments. We obtained a further improvement of 1.6% in f-measure, as shown in Figure 4.3 with f-measure of 0.54.

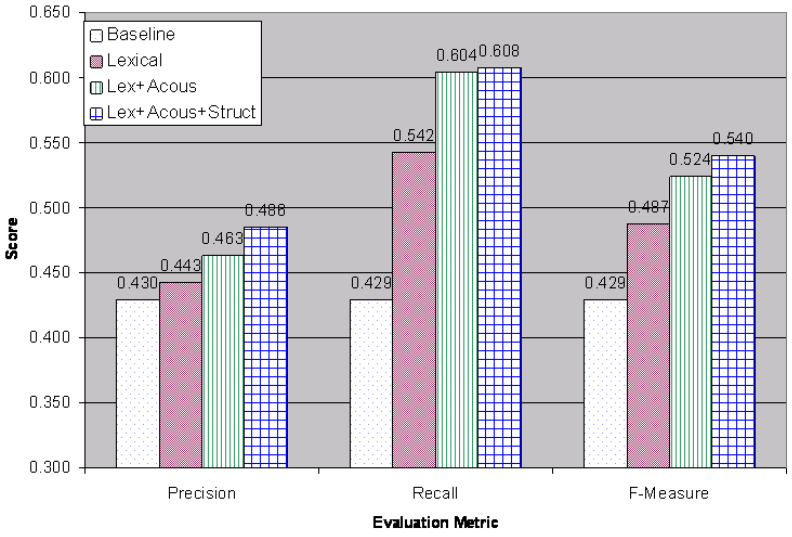


Figure 4.3: Improvement With the Addition of Structural Information

One of the issues that we have not addressed in this experiment is that structural cues can be heavily dependent on the news program and we may lose the improvement gained by the use of structural information if we test the system on broadcasts from different channels. We propose to investigate this issue further as a part of future work.

In the beginning of this chapter we asked the question, "Is it sufficient to use text summarization techniques on ASR transcripts for summarizing spoken document?" We have experimentally shown that it is not sufficient to summarize speech just by employing text summarization techniques on ASR transcripts. Rather, a speech summarizer should exploit acoustic/prosodic features of the speech signal and structural information of the document to perform better. The difference of 5.3% in f-measure due to this additional acoustic and structural information over just using lexical information supports this conclusion.

Our findings also suggest it may be possible to do effective speech summarization without the use of transcription at all, whether manual (as employed here) or from speech recognition. Two of our feature-sets, acoustic/prosodic and structural, are independent of lexical transcription, except for sentence-level and speaker segmentation and classification, which have been shown to be automatically extractable using only acoustic/prosodic information [Shriberg et al., 2000]. The accuracy of our acoustic/prosodic features alone ($F = 0.47$), and of our combined acoustic/prosodic and structural features ($F = 0.50$) compares favorably to that of our combined feature-sets ($F = 0.54$). So, even if transcription is not available, it is possible to summarize broadcast news effectively.

4.2.3 Combining All Available Information for Summarization

After experimenting with various combinations of feature sets we created all possible combinations of our 4 features classes —- Lexical (L), Acoustic/Prosodic (A), Structural (S) and Discourse (D) information, producing 15 different feature sets to experiment with. Then we ran a 10-fold cross validation experiment on these combinations with the same data and model settings. The results are shown in Figure 4.4 which confirms our belief that speech summarization should use additional information to produce a better summary.

From Figure 4.4 we can see that the best performing experiments combine all the feature-sets $L + A + S + D$. This gives an f-measure of 0.54, recall of 0.61, precision of 0.49 and an accuracy of 73.8% on our dataset with 10-fold cross validation. Our system thus has an f-measure which is 11% higher than the baseline. The f-measures for the individual feature-sets when tested alone are: discourse (0.13), structural (0.33), acoustic/prosodic (0.47) and lexical (0.49). So we see that the lexical and acoustic/prosodic feature-sets perform best alone, both surpassing the baseline.

When we look at the performance of discourse features alone, compared to that of discourse plus structural features, we see that the f-measure improves from 0.33 to 0.39. Discourse features added to lexical improve the f-measure from 0.49 to 0.50, and added to acoustic/prosodic features, improve the f-measure from 0.47 to 0.48. So, there appears to be more redundancy of our discourse features with acoustic/prosodic and lexical features than with the structural feature-set.

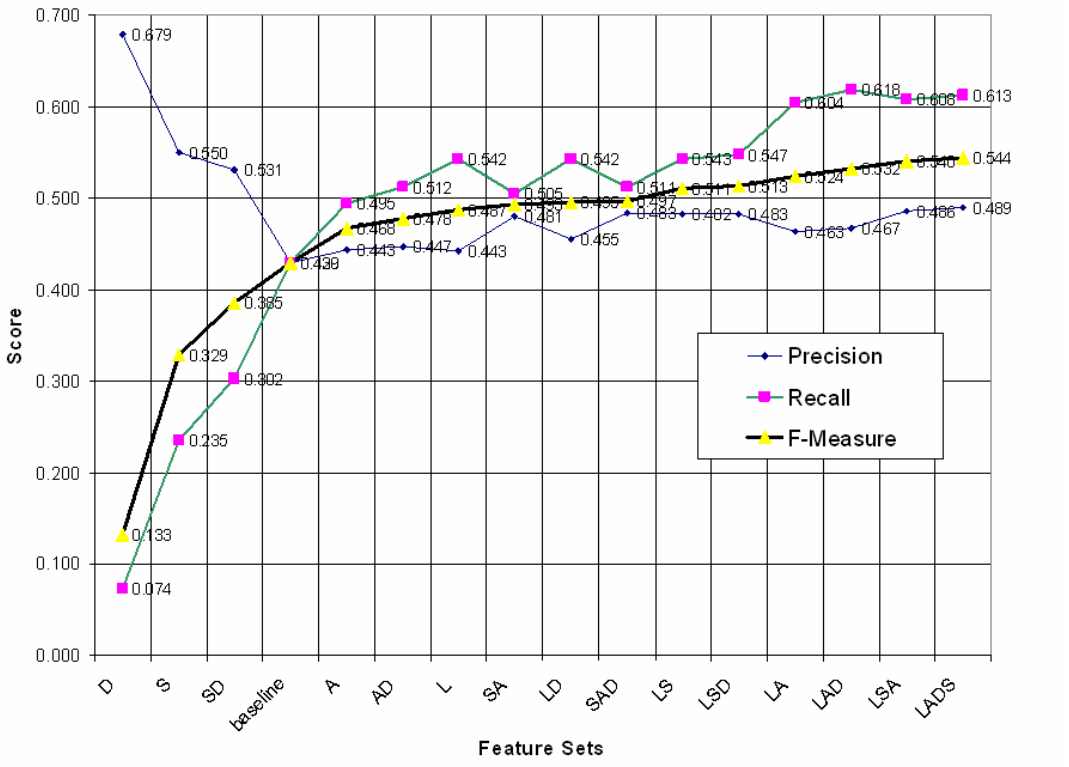


Figure 4.4: F-Measure: Various Combinations of Lexical (L), Acoustic (A), Structural (S) and Discourse (D) Information

To look more specifically at which of the features in our feature-sets are most useful for predicting summary sentence selection, we performed feature selection on our entire set of features using a selection algorithm that computed individual predictive power of each feature and the redundancies between features. The five most useful features are shown in Table 4.2. Note that the best performing individual features include features from all four of our feature-sets with two from the lexical set and one from each of the others. Interestingly, this set of five was also selected as the optimal set of features by the feature selection algorithm. Our f-measure with just

Table 4.2: Best Features for Predicting Summary Sentences

Rank	Type	Feature
1	A	Time Length in sec.
2	L	Num. of words
3	L	Tot Named Entities
4	S	Normalized SentPos
5	D	Given-New Score

these features is 0.53 which is only 1% lower than the highest f-measure shown in Figure 4.4.

The results mentioned above assume an exact match of a predicted summary sentence to a labeled summary sentence. For summarization purposes, this measure is generally considered too strict, since a sentence classified incorrectly as a summary sentence may be very close in semantic content to another sentence which was included in the gold standard summary. Another metric standardly used in summary evaluation, which takes this synonymy into account, is ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [Lin, 2004]. ROUGE measures overlap units between automatic and manual summaries. Units measured can be n-gram, word sequences or word pairs. For ROUGE-N, ROUGE-L, ROUGE-S and ROUGE-SU, N indicates (the size of) the n-grams computed, L indicates the longest common subsequence, and S and SU stand for skip bigram co-occurrence statistics with and without optional unigram counting. ROUGE-N is computed using the following equation.

$$ROUGE - N = \frac{\sum_{S \in Ref.Sum} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in Ref.Sum} \sum_{gram_n \in S} Count(gram_n)}$$

Figure 4.5 presents results of evaluating our feature-sets using the ROUGE metric, with $N = 1$ to 4 and all of the variants described above.

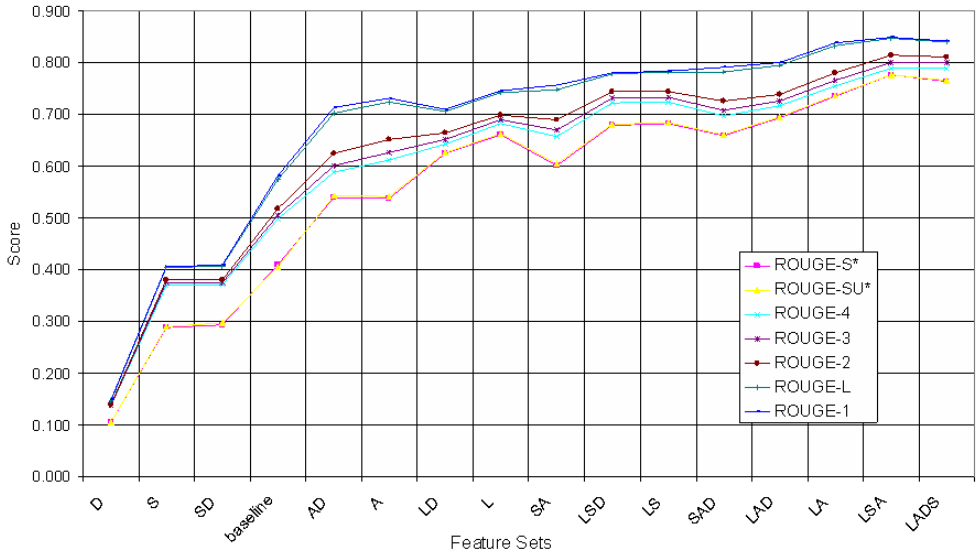


Figure 4.5: Rouge: Various Combinations of Lexical (L), Acoustic (A), Structural (S) and Discourse (D) Information

The results shown in Figure 4.5 are similar to those shown in Figure 4.4 using the f-measure metric. However, the difference between the baseline and our best combined feature-set is even greater using ROUGE. We obtained our highest ROUGE score of 0.85 with ROUGE1 and ROUGE-L which is 27% higher than the baseline. If we take an average of the performance of different ROUGE systems, we get a mean score of

0.80, which is 30.3% above the baseline. This mean score may be a more reasonable measure than ROUGE1, since it includes the performance of versions which look for more than mere unigram overlap. Note that the combined acoustic/prosodic and structural features alone obtain a ROUGE1 score of 0.76 and an average ROUGE score of 0.68.

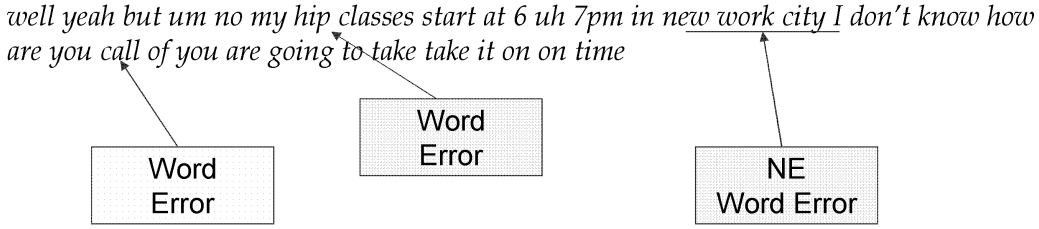
4.3 Additional Problems in Speech Summarization

We saw in the previous sections that the use of additional information improved our speech summarizer; but there are additional problems that speech summarization has to address. We saw in Section 4.2 that two of the best five features were based on lexical information. Lexical information is highly dependent on the quality of transcripts which, in turn, depends heavily on the errors introduced by ASR engines. ASR introduces word errors and boundary errors, while speaking style introduces disfluency errors. We explain these errors in the following sections and provide details on how to address them in Chapter 5.

4.3.1 Word Errors

The experiments described in previous sections were tested on manual and close captioned transcripts; but a fully automatic summarizer has access to only ASR transcripts. There are many problems that a speech summarization algorithm has to handle that a text summarizer does not have to address in using automatically generated transcripts. Although the general performance of ASR engines has improved

significantly in the last decade, ASR still produces transcripts with a significant number of word errors. Let us look at an example of an errorful ASR transcript shown in Figure 4.6.



*well yeah but um no my **step** classes start at 6 uh 7pm in **New York City***
*I don't know how are you **all** of you are going to take make it on on time*

Figure 4.6: Word Errors

The transcript shown above contains three word errors. "Step", "York" and "all" have been misrecognized as "hip", "work" and "call". These word errors not only make it very hard to understand the meaning of the text but also result in significant challenges for downstream processing of the transcript, such as Named Entity (NE) detection. For example, misrecognition of "york" may cause to fail NE detector to label 'New York City' as a location entity. Since NEs are important features for summarization [Maskey and Hirschberg, 2005], lower accuracy in the detection of NEs will result in poor summarization accuracy. Word errors can also degrade the performance of many other syntax-based NLP tools, such as chunking and parsing. Summarization of a spoken document by text summarization methods alone may thus yield poor summarization results if we do not address the word errors. We present an experiment showing the use of confidence scores to take account of word errors in

Section 5.1.

4.3.2 Disfluency Errors

One of the major differences between a text and a speech document is the lack of disfluency in text documents. Humans generally write well-constructed sentences but speak disfluently by repeating or correcting words and phrases, and adding words in the sentences that break their grammatical construction.

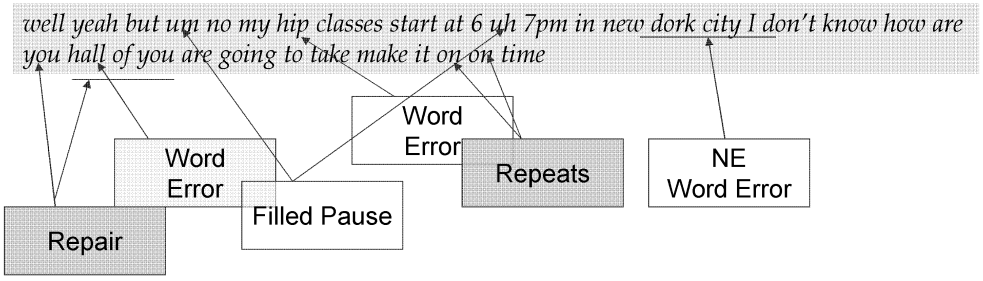


Figure 4.7: Disfluency Errors

In the example shown above there are three types of disfluencies — 'uh, um' are fillers, 'you, all of you' is a repair and 'on on' is a repeat. We notice from the example that disfluencies make it harder to understand the meaning of the text. Disfluency such as 'on, on' can result in parsing errors because most of the NLP parsers expect grammatically correct sentences. Increased parsing errors may result in reduced summarization accuracy, especially for text summarization algorithms that depend on accurate syntactic analysis. We propose a disfluency detection algorithm in Section 5.2.

4.3.3 Boundary Errors

Most extractive text summarization algorithms depend on sentence and paragraph boundaries. Transcripts generated by ASR engines have no notion of sentence or paragraph boundaries. ASR transcripts not only lack sentence boundaries but also lack commas and capitalization that may help in detecting these boundaries. Text summarization methods based on sentence extraction face a serious challenge without sentence boundaries. The speech segment in the example below lacks two sentence boundaries, resulting in a difficult situation for extracting sentences as the system does not know where a sentence begins and ends. Instead of sentences we extract intonational phrases as units for summarization as described in Chapter 5.

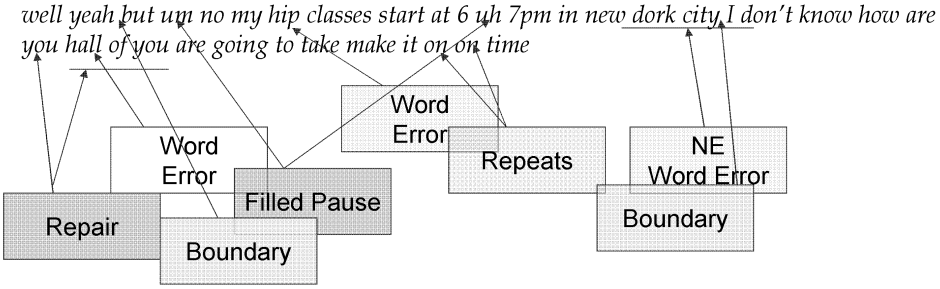


Figure 4.8: Boundary Errors

The problems we have mentioned above such as lack of boundaries, word errors and disfluency in ASR transcripts can significantly degrade the performance of summarization algorithms. We may be able to export some text summarization techniques and features for speech summarization but we would need additional methods to handle such errors. We present such techniques in the next chapter.

Chapter 5

Speech Summarization Challenges: Word Errors, Disfluency and Segmentation

In this chapter, we present methods to handle the *additional challenges* that we described previously — word errors, disfluency and speech segmentation.

5.1 Overcoming Word Errors

One of the important tasks of a summarizer is to find redundant information. Finding redundant information entails finding segments of text or speech that are similar in meaning. One of the known methods to find similarity between sentences is Cosine Similarity (CS). CS has been used to cluster sentences together to automatically identify topics/sub-topics of a document or a collection of documents. Word errors in

ASR transcripts reduce the performance of the CS score, which in turn, reduces the accuracy of a clustering algorithm, thus making it harder to find redundant sentences for summarization. We present a method to reduce the impact of word errors when computing CS between sentences, allowing us to improve the summarization module that finds redundant information.

5.1.1 Using Confidence Scores

ID	Sentence	Word Confidence Scores
1	this is headline news	1, 1,1,0.9, 1
2	dan rather reporting	0.8, 1, 0.9

Table 5.1: Example: Sentences in Paragraph 1

In order to compute similarity between text spans (sentences, paragraphs or documents) using the CS measure, we first create word vectors from the text spans. Word vectors contain the counts of all the unique words in the documents. Then, we compute the cosine angle between these word vectors. A cosine value of 1 represents no difference between word vectors (the sentences are completely redundant) and cosine value of 0 represents no similarity. Let us look at an example: let the sentences ID1 and ID2 be a paragraph I of a document D, as shown in Table 5.1. Let sentences ID1' and ID2' be sentences from paragraph I' of document D as shown in Table 5.2'. Let us compute the similarity between the sentences ID2 and ID2'.

If we build the word vectors for these sentences in 2 paragraphs we get the word vectors shown in Table 5.3, where dimension for these word vectors are defined by

ID'	Sentence	Confidence Scores
1'	This is Headline News,	1,0.8, 1, 1
2'	Dan Radar reporting CNN	0.8, 0.3, 1, 1, 1

Table 5.2: Example: Sentences in Paragraph 2

[headline,is, this, dan, news, rather, reporting, radar, cnn]

ID	Word Vectors
ParaI-sentID2	[0,0,0,0,1,1,1,0,0]
ParaII-sentID2	[0,0,0,0,1,0,1,1,1]

Table 5.3: Word Vectors for Similar Sentences

If the word vectors obtained in Table 5.3 are from written text, then computing CS between them is not a problem, as we would have known for sure that "Radar" and "Rather" are two different words. But if the sentences are from ASR transcripts, assuming that these words are correct can give a false sense of dissimilarity between sentences. If we look at the confidence score of "Radar" in Table 5.2 we note a low confidence of 0.3. This low score should affect the similarity metric accordingly. How to take account of low confidence in a similarity metric is the question we are seeking to solve. We take an approach that is simple yet effective. We weight each dimension of the word vector with its confidence score. Hence, for the dimension of the word "Radar," instead of getting values of 0 and 1 as shown in the 8^{th} column (2^{nd} to the last) of the word vectors in Table 5.3, we get values 0 and 0.3 as shown in Table 5.4. If there are many occurrences of the same word we take the average of the confidence scores and weight the dimension with that average.

ID	ASR Confidence-Weighted Word Vectors
ParaI-sentID2	[0,0,0,0,0.8,1,0.9,0,0]
ParaII-sentID2	[0,0,0,0,0.8,0,1,0.3,1]

Table 5.4: ASR Confidence-Weighted Word Vectors for Similar Sentences

In this vector based model, word errors in the spoken document transcriptions are effectively addressed by introducing weights for each dimension and changing the weights according to the ASR confidence score for each word. The confidence level on the word prediction will change the similarity score for documents. This would enable us to have a low similarity score if we believe there is a probability of many words being misrecognized.

We test the above confidence-weighted similarity metric by clustering documents on the same topic with the intuition that a better clustering of these documents should translate to a better module for identifying documents with redundant information. We explain our experiment in the following section.

5.1.2 Clustering Experiment

We clustered 1841 TDT2 [LDC, 2002] documents into 69 different clusters where cluster sizes ranged from 1 to 301 documents.[1] All of the documents were annotated by human labelers with a topic-label describing the cluster they belong to. We removed foreign documents because we were primarily testing clustering module for English. We also removed multiple-topic documents to better test the effect of our

[1]This experiment was done jointly with Sasha Blair-Goldensohn.

model on a set of documents for each topic. This resulted in 62 clusters for this experiment. We performed the experiments in three different conditions to distinguish the improvement in clustering using the confidence-weighted CS metric that we described in previous section. We first clustered *baseline ASR word vectors* where ASR transcripts were treated as regular text. Then we clustered *confidence-weighted ASR word vectors* where we weighted the vectors with confidence scores. Then we clustered *close-captioned word vectors* to estimate the improvement ceiling. The results are shown in Table 5.5.

The metric used to compute results in Table 5.5 is proposed by [Rosenberg and Hirschberg, 2007] which introduces "homogeneity" as a measure akin to precision, measuring the degree a given cluster cluster contains elements from a single true cluster; "completeness" as a measure akin to recall measuring the degree to which the elements of the same cluster are grouped together; and "validity" as a measure akin to f-measure, combining homogeneity and completeness in a harmonic mean. The measures scale from 0 to 1 with 1 being a perfect score. A perfect clustering method will have all three scores equal to 1.

Exp. Type	Homogeneity	Completeness	Validity
ASR Baseline	0.78	0.668	0.72
Confidence-Weighted	0.758	0.705	0.731
Manual Transcript	0.928	0.952	0.928

Table 5.5: Overcoming Word Errors

The drop in the validity score in Table 5.5 when ASR transcripts are used in-

stead of manual transcripts shows that word errors can significantly degrade overall performance of a clustering algorithm based on CS. But we also see that we gain about 1% in the validity score when we use the confidence-weighted CS metric for clustering. The improvement of 3.7% in completeness shows that more correct documents of each cluster are being identified but a 2.2% drop in homogeneity could mean that some documents are assigned to incorrect topics. This could happen because for confidence weighted clustering algorithm to fully utilize the advantage of word confidences, a cluster needs to have a large enough set of correct words in the cluster which happens only after the first few iterations in clustering. The improvement of 1% in the clustering experiment shows that the use of ASR confidence scores to weigh the word vectors has reduced, though by a small amount, the impact of word errors in computing similarity between text segments.

ASR word errors are one of the challenges speech summarization faces and we have presented a method to deal with such errors. Improved clustering due to our use of an improved similarity metric provides a better chance of finding redundant information, and thus a chance to develop a better speech summarizer. Another method to use confidence weights for summarization is to assign a lower summary score for sentences with many low confidence words. Too many low confidence words in a sentence would mean we cannot rely on linguistic and discourse features dependent on the identify of these words. In the following section we address the next challenge speech summarizers face, disfluency.

5.2 Disfluency Removal

Disfluency is common in speech and can also significantly degrade the overall performance of a speech summarizer. Disfluency detection can be useful for increasing the readability of a speech transcript, thus possibly improving the accuracy of a summarizer. In this section we propose a phrase-level machine translation approach for disfluency detection using Weighted Finite State Transducers (WFSTs).[2]

Repeats	**I want to buy three glasses** *** three glasses of tea**
Repairs	**I want to buy three glasses** *** no five cups of tea**
Fillers	**I want to buy three glasses** *** umm four glasses please**

Table 5.6: Example of Disfluencies

5.2.1 Definition and Related Work

We follow the definition of [Stolcke and Shriberg, 1996] and divide disfluencies into three main components: **reparandum** (the words that are repaired), **interregnum** (filler words or filled pauses) and **resumption** (the new set of words that repair the

[2]The work presented in the following Section was jointly done with Bowen Zhou and Yuqing Gao at IBM. The algorithms and techniques presented here are owned by IBM Corporation with a patent filed by IBM with title "Disfluency Detection for a Speech-to-Speech Translation System Using Phrase-Level Machine Translation Approach with Weighted Finite State Transducers',' IBM docket no. YOR920060506US1.

reparandum). In this thesis we detect three types of disfluencies: **repeats** (reparandum edited with the same sequence of words), **repairs** (reparandum edited with different sequence of words) and **filled pauses** (words in interregnum region).

Fillers are placed at the interruption point of the speaker's turn. Fillers include filled pauses such as 'um', 'uh', 'well'; discourse markers such as 'well', 'then', 'you know' and editing terms. Filled pauses may serve to signal hesitation or confusion in the speaker or to signify change in a given topic of conversation depending on the type of filled pause a speaker uses. In Table 5.6 'umm' is a filled pause placed at the interruption point '*'.

Repeats are one of the most common types of disfluencies. In the above example in Table 5.6, 'three glasses' is a repeat. Any such occurrences of verbatim repetition of a portion of a spoken utterance are 'repeats'. The phrase 'three glasses' is reparandum, which is repaired with 'five cups' after the interruption point.

There has been a significant amount of work in disfluency detection [Kim et al., 2004, Nakatani and Hirschberg, 1994, Heeman et al., 1996, Honal and Schultz, 2005, Johnson et al., 2004, Snover et al., 2004b, Spiker et al., 2000]. Some disfluency detection systems have been built in the DARPA EARS Rich Transcription program. Most systems that have been proposed use combinations of prosodic and lexical features though some systems are lexically driven [Snover et al., 2004b] without any use of acoustic features. Snover [Snover et al., 2004b] rely exclusively on word based information and have shown that reasonable performance can be obtained without using acoustic features. Johnson and Charniak [Johnson and Charniak, 2004] use a Tree Adjoining Grammar (TAG) noisy channel model and [Heeman et al., 1996] focus on

similarity and the length of repeated sequence of words in the ASR transcripts.

On the other hand Nakatani and Hirschberg [Nakatani and Hirschberg, 1994] have shown the advantages of using acoustic/prosodic features. They successfully detected interruption points (IP) by building a decision tree with acoustic features. [Shriberg et al., 1997] also proposed a method of detecting IPs by building a decision tree model based on prosodic features only. [Stolcke et al., 1998] improved this system with the addition of a hidden event language model (LM) to detect boundaries and various types of disfluencies.

The addition of prosodic features to word based features has some clear advantages. For example, usually the intonation of a speaker is disrupted at the interruption point indicating some form of restart. Such information is useful and has been shown to be significant [Liu et al., 2003]. [Liu et al., 2003] successfully combined lexical and prosodic features and detected the onset of reparandum. Even though the use of combined lexical and prosodic features has some clear advantages, it should be noted that the prosodic features may not always be easily available.

5.2.2 Approach

We view the disfluency removal problem as a process that transforms the "noisy" disfluent transcript into a "clean" one [Honal and Schultz, 2003, Honal and Schultz, 2005]. Such a transformation can be described using statistical machine translation models. Particularly, Zhou and Gao [Zhou and Gao, 2006] have formulated a fast phrase-based statistical translation using FST's for a speech-to-speech (S2S) translation. We are motivated to apply a similar framework as [Zhou and Gao, 2006] to

address disfluency detection.

There are several advantages of the proposed scheme. First, this approach enables the disfluency detection component to be easily integrated with other FST-based components by a simple FST composition operation, providing a unified search space. Second, the proposed approach obtains very high speed and memory efficiency using an optimized decoder.

5.2.2.1 Translation Model

Based on a source channel model approach to statistical machine translation [Brown et al., 1993], translating a foreign token sequence n_1^J to a target token sequence c_1^I can be viewed as a stochastic process of maximizing the joint probability of $p(n, c)$ as stated in Equation 5.1:

$$\hat{c} = argmax_{c_1^I} Pr(n_1^J, c_1^I) \tag{5.1}$$

The joint probability can be obtained by summing over all the hidden variables that can be approximated by maximization. For machine translation purposes these random variables take account of alignment, permutation, and fertility models as described in [Brown et al., 1993]. For our purpose we view disfluency detection as translation from a noisy token sequence $n_1^J := n_1, n_2, ..., n_J$ to a clean token sequence $c_1^I := c_1, c_2, ..., c_I$. Since the removal of disfluency will entail removal of words from n_1^J we would still require alignment and fertility models as $I < J$.

We simplify the training of our translation model by retokenizing the c_1^I sequence. Instead of a clean speech transcript without any disfluent words, we append a tag that signifies the type of disfluency for each disfluent word in n_1^J. This retokenization

produces c_1^I with the same number of words as n_1^J such that $I = J$ as shown in the following example.

- Noisy Data: I want to buy three glasses no five cups of tea
- Clean Data: I want to buy REPAIR0 REPAIR1 FP0 five cups of tea

These modifications to the standard machine translation model simplify our model in the following ways: i) We do not require a fertility model since the number of words in clean and disfluent speech are equal and words in the noisy speech transcript are preserved as disfluent tags. ii) With disfluent words retokenized ($I = J$) we have a perfect alignment between the noisy and clean transcripts in the parallel corpora, removing the need for an alignment model.

5.2.2.2 Phrase Level Translation

Repeats and repairs are difficult to detect because the reparandum of these disfluencies can include more than one word. In our example in Table 5.6 the reparandum is "three glasses," a two word phrase. The reparandum phrase can be of any length though phrases longer than five words are very rare. Word-based disfluency detection algorithms have difficulty in detecting such disfluent phrases because the classifier not only has to classify words as disfluent or not but also has to detect the start and end boundaries of the reparandum. This added complexity in repeat and repair detection can be addressed if we assume that disfluency occurs in phrases. Since we define a phrase as a sequence of one or more words, single word disfluencies are also addressed with such phrase assumption.

Our phrase level translation model is built in a process that is similar to the one described in [Zhou and Gao, 2006]. Since our bi-text alignment is a perfect one-to-one mapping, the phrase pair extraction procedure is simple, and the only variable to consider is the phrase length limit. The length limit on the phrase size makes a significant difference in the size of the dictionary. We chose a maximum phrase size of five, as 99.9% of the disfluent phrases in our training corpus were smaller than five words. Our algorithm include techniques for phrase pair extraction, the phrase translation model estimation, and the WFST implementations. The process also includes techniques to determinize and minimize the transducers to optimize the search procedure.

We denote the phrase segmentation by introducing a hidden variable p_1^K to the Eq. 5.2 summing over the joint probability. In addition, we can approximate the sum over the hidden variables using a maximum operator.

$$\hat{c} = argmax_{c_1^J} \sum_{p_1^K} Pr(p_1^K, n_1^J, c_1^J) \tag{5.2}$$

$$\approx argmax_{c_1^J} max_{p_1^K} Pr(p_1^K, n_1^J, c_1^J) \tag{5.3}$$

5.2.2.3 Weighted Finite State Transducer Implementation

We can implement Equation 5.2 using weighted finite state transducers. Using the chain rule we can easily decompose the joint probability into a chain of conditional probabilities as follows, in a similar way to [Zhou et al., 2005, Zhou and Gao, 2006]:

$$Pr(p_1^K, n_1^I, c_1^I) = P(c_1^J). \tag{5.4}$$

$$P(p_1^K|c_1^J). \tag{5.5}$$

$$P(n_1^I|p_1^K, c_1^J) \tag{5.6}$$

We can compute the conditional probabilities of Equations 5.4, 5.5 and 5.6 by using the parallel corpus and the phrase dictionary. Furthermore, we can build WFST for each probability distribution - L, N and P where L is a language model, N is the translation model and P is the phrase segmentation model respectively.

The arc probabilities for the translation model N are computed by computing the relative frequencies from the collected phrase pairs.

$$P(c|n) = \frac{N(c,n)}{N(c)} \tag{5.7}$$

where $N(c,n)$ is the number of times a clean phrase c is translated by a noisy phrase n. The above equation overestimates the probabilities of rare phrases. In order to take account of such overestimation we smoothen our translation probability by performing a delta smoothing. We add a small numerical quantity δ on the numerator of Equation 5.7 and add $\delta \cdot |V|$ on the denominator where V is the size of the translation vocabulary for a given phrase.

The language model plays an important role in a source channel model like ours. Our language model L is a standard trigram language model with the n-gram probabilities computed from the clean corpus that has disfluent words tagged as REPEAT, REPAIR and FP (filled pauses). In other words, we used the annotated side of the parallel corpus as the language model training data. We built a back-off 3-gram language model, and encoded it as a weighted acceptor as described in [Zhou and Gao, 2006] to be employed by our translation decoder.

After building all three types of WFSTs we can perform a *cascaded composition* of these finite state transducers as shown in the equation below to obtain a translation lattice that translates sequence of noisy words to clean phrases.

$$T = P \cdot N \cdot L \tag{5.8}$$

For decoding, we employed the decoder that is developed in [Zhou and Gao, 2006] using a multilayer search algorithm. Specifically, we have one layer for each of the input FSMs: the noisy input/acceptor, the translation model lattice, and the language model lattice. At each layer, the search process is performed via a state traversal procedure starting from the start state, and consuming an input word in each step in a left-to-right manner. This can be viewed as an optimized version of on-the-fly or dynamic composition. However, specialized versions of composition have the advantage of not only having the potential to be many times faster than general composition implementations found in FSM toolkits, but also in incorporating information sources that cannot be easily or compactly represented using WFSTs. For example, the decoder can allow us to apply translation length penalties and phrase penalties to score partial translation candidates during search. In addition, they can incorporate new parameter values (e.g., language model weight) at runtime without the need for any modification of the input WFSTs.

One significant disadvantage of using WFST based translation method is a lack of simple feature based probabilistic training algorithm for WFSTs. Even though Eisner [Eisner, 2002] has proposed a training algorithm for "probabilistic WFST", it is not an easy task to add features for parameterization of WFSTs. For example, if

we want to add a feature that defines "if the previous word is the same as the current one," in the WFST we would need to add self looping arcs for all the words in the vocabulary, adding significant amount of confusability. It is obvious that any more complicated rule for parameterization would add many arcs to the translation lattice. We propose a simple method to add such features in our WFST framework.

We limit ourselves to nominal features. We extract nominal features for each word and convert them to binary strings. We append each binary string to the end of each noisy word. For example we can have features such as "if the next word is the same as the current one," and "if the part of speech of the next word is the same as current one." If f_1, f_2 were the binary features mentioned above each noisy word n is appended with $f_1 f_2$ producing $n_{f_1 f_2}$. Adding too many features can increase data sparsity problems, so we have to be careful about the number and type of features we use.

5.2.3 Disfluency Detection Experiment

We tested our method on the Penn III Switchboard Tree Bank corpus. We split our data into a training set of 1,221,502 words and a held-out test set of 199,520 words. We performed all of our experiments on manual transcripts. We do not assume that we have sentence boundaries or interruption point information, but we do assume that we have turn boundaries, so that the decoder can handle sizable chunks of speech without severe memory problems.

For training, we used the annotation for repeat, repair and filled pauses that were provided with the corpus. We converted the Switchboard annotation to produce a

parallel text corpus of clean and noisy speech transcripts where all the disfluent words in a noisy transcript align with disfluent token IDs. These disfluent token IDs signify the type of disfluency for the corresponding word in the noisy transcript as shown in the example in Section 5.2.2.1. Let us define C as the annotated clean corpus and N as the corresponding noisy parallel corpus.

We first extracted two lexical features for all the 1.22 million words in corpus C and appended these feature values to each word, thus obtaining a corpus CF. The two features we extracted were "*are part of speech tags of any of the next two words the same as the current one?*" and "*are any of the next two words the same as the current word?.*" In order to obtain part-of-speech feature values, we use automatic part-of-speech tagger [Ratnaparkhi, 1996]. The training corpus had 13,287 repair phrases with only 48 phrases longer than 5 words. Most of the disfluent phrases are composed of one or two words. 11.8% of all the phrases consist of more than 2 words.

Using the phrase extraction algorithm employed in [Zhou and Gao, 2006] we extracted all the parallel phrases from the corpora N and CF. We set the limit of phrase size as five words. We obtained a phrase dictionary with 2.85 million entries. Each entry consists of a noisy phrase, a corresponding clean phrase and a probability for the translation, computed as described in Section 5.2.2.3. We smoothed the transition probability with a δ of 0.01.

We built the WFSTs as described Section 5.2.2.3 and composed them to produce a noisy to clean speech translation lattice. We built the language model using the IBM language model toolkit. The size of the final translation lattice is listed in Table 5.7.

When we tested our model on our held out test set we obtained the results listed in Table 5.8. We tested our method by using the standard precision, recall and f-measure. The scores were computed at the word level. The train and test data are heavily skewed with very few positive examples of disfluency. In our test set of 199520, only 6.93%, of words were disfluent, so f-measure is a reasonable metric for the system evaluation.

Type	# of states	# of transitions
Translation	1,635,398	3,426,379
LM	234,118	1,073,678

Table 5.7: The Size of the Translation Lattice and LM

We built two different types of translation lattices to examine the effects of the language model. When we added the language model (LM) to the translation model, the f-measure improved for repeats by 4.9%, precision by 4.8% and recall by 5.1%. Similarly for repairs, f-measure improved by 6.7%, precision by 3% and recall by 7.5%. The addition of the LM only slightly improved filled pause results signifying that the LM is more critical for detecting repeats and repairs than filled pauses. We also note that the improvement in f-measure and recall with the addition of LM for repairs is significantly higher than it is for repeats, possibly showing that surrounding word context is more important for repair detection. The f-measure for repeat detection is 39.6% higher than the f-measure for repair detection. We obtain a very high recall of 0.86 and f-measure of 0.797 for repeat detection.

We are able to detect filled pauses with a greater accuracy than repeats and

	Disfluency	Precision	Recall	F-measure
without LM	REPEAT	0.695	0.809	0.747
	REPAIR	0.479	0.256	0.334
	FILLED PAUSE	0.953	0.998	0.975
with LM	REPEAT	0.743	0.860	0.797
	REPAIR	0.509	0.331	0.401
	FILLED PAUSE	0.955	0.998	0.976

Table 5.8: Disfluency Detection Results on Held-out Test Set

repairs. We obtain an f-measure of 0.976 for filled pause detection. One of the reasons that the same translation model does very well for filled pauses but not for repeat and repairs is because most of the filled pauses are unigrams and a few set of words constitutes most of the filled pauses. The most common filled pause in our test corpus was "uh" constituting 82.7% of all the filled pauses. The least occurring filled pauses were "ah", "anyway" that occurred only once.

One of the key advantages of using our method for disfluency detection is speed. We are able to detect disfluency in one thousand sentences in less than a second using a memory optimized WFST decoder developed for phrase-based translation [Zhou and Gao, 2006].

5.2.4 Disfluency Removal and Summarization

We described a novel approach that is based on a phrase-level translation framework for detecting repeats, repairs and filled pauses. Our method requires a parallel corpora

of noisy and clean speech transcripts and not many other natural language resources.

We removed disfluencies in our CBNSCII corpus using a method similar to that described above.[3] We performed a 10-fold cross validation using the same features and ML techniques as we described in Chapter 4. We gained less than 1% improvement in overall summarization score. We did not see much gain possibly because of the following reasons. First, the model was trained on telephone conversations but tested on BN — the speaking style between BN and telephone conversation differ significantly. Second, BN speech is less disfluent than other spoken conversations because anchors read from scripts.

So far, we have addressed two challenges that speech summarizer face including word errors and disfluency. Next, we look at the problem of speech segmentation.

5.3 Speech Segmentation: What Unit to Extract?

We have mentioned previously that the extraction unit chosen, whether a sentence, syntactic constituent, or other segment, has a significant impact on the overall quality and fluency of the summary. Sentences may not always be an ideal unit to extract, as they are hard to detect. What unit to extract and how to detect the boundaries for such a unit is another major challenge a speech summarizer has to address. In this section, we compare sentences, intonational phrases and pause-based acoustic 'chunks' as candidate units of extraction, and show that extraction of automatically

[3]We used a slightly different disfluency removal system than the one described above. The algorithm described above could not be used outside IBM for legal reasons.

segmented intonational phrases may produce better speech summaries.[4]

Generally, speech segments extracted for summarization should be semantically meaningful and coherent stretches of speech. Segmentations currently used or proposed for extractive summarization include words, phrases, sentences, and speaker turns. Choice of segmentation unit greatly influences the length and quality of the resulting summary. If speaker turns are extracted, the shortest summary will be a single turn, which may contain many sentences, not all of which may be important. If there are only ten turns in a document, a compression ratio of less than 10% is impossible — a significant limitation for many summarization tasks. We have the most control over the length of the summary if we extract individual words. However, by sacrificing higher-level structural, semantic and syntactic information from the source document, this approach is likely to be limited to a set of key words. Sentences might be a better choice of segmentation for extraction, since they are shorter than turns, affording finer control over the length of a summary. Moreover, sentences are also semantically and syntactically meaningful units. However, longer sentences may also include modifiers, phrases and clauses which are not essential for the summary. Phrase extraction is a promising alternative to sentence extraction, but identifying phrases in speech transcripts using current Natural Language Processing (NLP) tools is errorful, for the reasons we have described before. It is however possible to identify phrasal units using acoustic and prosodic information from the speech source.

[4]This work was done jointly with Andrew Rosenberg at Columbia University.

5.3.1 Speech Segmentation

In order to determine which segmentation type is best for extractive summarization of spoken documents, we must first produce candidate segments at different levels of granularity. We describe these segmentations and the techniques used to generate them next.

Segmentation	number per story	avg. length (words)
250ms Pause	43.2	11.47
500ms Pause	19.1	25.97
Sentence	26.9	18.46
Intonational Phrase	71.2	6.96

Table 5.9: Speech Segmentation Statistics

5.3.1.1 Pause-Based Segmentation

To generate pause-based segments, we calculated the pause duration between each pair of ASR-hypothesized words. We inserted a segmentation boundary at every pause that exceeds a pre-determined threshold. For these experiments, we constructed two input segmentations — one using a 250ms threshold, another with a threshold of 500ms. Obviously, the set of boundaries selected with a 250ms threshold is a superset of those selected with a 500ms threshold. We hesitate to use a threshold below 250ms due to the potential confusion of stop gaps with phrasal boundaries [Santen, 1992].

5.3.1.2 Automatic Sentence Segmentation

We used an automatic sentence boundary detector from ICSI [Liu et al., 2006] to produce sentence segmentation together with confidence scores for each hypothesized boundary. This module comprises three components: a language model (LM), a prosodic model (PM), and a strategy for combining the models. The PM is implemented as a decision tree which predicts whether or not a given word occurs at a SU boundary based on acoustic features of the word. The SU detection module is implemented as a majority voting combination of a Hidden Markov model (HMM), Conditional Random Field (CRF) and Maximum Entropy (ME) model. Each of these accepts a word sequence, the hidden event sequence produced by the LM, and output from the PM as input, and produces hypothesized SU boundaries. The training data for this model was manually annotated human transcriptions of BN. On automatically recognized speech, it operates with an error rate of 57.23%.

5.3.1.3 Intonational Phrase Segmentation

The third kind of speech segmentation we looked at was Intonational Phrases (IPs). To produce training material for a ToBI-based intonational phrase (IP) model, we asked an expert ToBI labeler to manually annotate one half hour TDT4 show, an ABC "World News Tonight" broadcast for (binary) pitch accent presence on each word and for level 4 phrase boundaries. The annotator annotated the ASR transcript of this show, marking a hypothesized word as ending an intonational phrase if the phrase ended after or within the ASR-hypothesized word boundaries. After omitting regions of ASR error, silence and music, the training material included approximately

20 minutes of annotated speech and 3326 hypothesized words.

Using the J48 java implementation in weka [Witten and Frank, 2000] of Quinlan's C4.5 algorithm [Salzberg, 1994] to train a model on this single annotated show, we classified each ASR word in our summarization material as either preceding an intonational boundary or not. This decision tree is trained using feature vector containing only acoustic information: pitch, duration and intensity features. These features are extracted from raw and speaker normalized pitch and intensity tracks calculated using Praat [Boersma, 2001]. The features that were the most indicative of the presence of a phrase boundary were: a long pause following the word, a descending change of energy over the final quarter of the word, lower minimum energy relative to the two preceding words, and decreased standard deviation of pitch. On the training material, based on ten-fold cross-validation experiments, intonational phrase boundaries are predicted with 89.1% accuracy, and an f-measure of 0.665 (precision: 0.683, recall: 0.647).

5.3.2 Segment Comparison for Summarization

We next built summarizers that extract segments based on these four segmentation units: pauses of 250ms and 500ms, sentences, and IPs. We extracted features for each segment type and assigned summary labels to each unit, using the forced alignments with manual summaries described in Section 2.1. We constructed each of our four summarizers as a binary Bayesian Network classifier [Heckerman, 1995] where the summarizer's task is to determine whether a given segment should be included in the summary or not. All the experiments were performed on the documents of

Summarization Corpus I.

5.3.2.1 Features

We used only acoustic and structural features that were independent of the word identities hypothesized in the ASR transcripts. Using only acoustic/prosodic and structural features allows us to avoid the potential effects of word errors in the segments we extract.

We extracted aggregated acoustic features over the candidate segments. We extracted the minimum, maximum, standard deviation, mean of f0, Δ f0, RMS intensity (I) and ΔI over each segment. We also extracted the z-score ($\frac{value-mean}{std.dev.}$) of the maximum and minimum within the segment over these four acoustic information streams. Using the pitch (f0) tracks only, we extracted three pitch reset features. These were calculated by taking the difference between the average of the last 10, 5 or 1 pitch points (calculated using a 10ms frame) in the current segment, and the average of the first 10, 5 or 1 pitch points in the following segment. We normalized the f0 and intensity tracks using z-score normalization, and calculated speaker normalized versions of the previously described features. We also included in the feature vector the average word length within the segment. The raw pitch tracks were extracted using Praat's [Boersma, 2001] 'To Pitch (ac)...' function, intensity tracks using 'To Intensity...'. These features totaled 87, most based on duration, energy, and pitch.

Using hypothesized story boundaries [Rosenberg and Hirschberg, 2006] and speaker turn boundaries provided by ICSI [Liu et al., 2006], we extracted a set of structural features for each segmentation. For each segment, we identified its length, absolute

and relative start time, relative position in the current speaker turn and relative position in the story. Additionally, based on the speaker identifications produced by the diarization module, we calculated the relative position of the current segment based on all of the material spoken by its speaker. We used these features to build four different summarizers each based on different segment length that we have described above.

5.3.2.2 Intonational Phrases for Speech Summarization

The results of 10-fold stratified cross validation experiments are shown in Table 5.10. We find that the best summarization (using f-measure) is obtained from the sum-

Segmentation	Precision	Recall	F-measure
250ms Pause	0.333	0.622	0.432
500ms Pause	0.255	0.756	0.381
Sentence	0.362	0.540	0.434
Intonational Phrase	0.428	0.650	0.516

Table 5.10: Information Retrieval-based Summarization Results

marizer which uses IPs as its input segmentation unit. The summaries produced represent a significant improvement of 8.2% in f-measure over sentence-based summaries. Note that, on average, there are about 2.75 intonational phrases for every sentence. This enables the summarizer to extract smaller segments for inclusion in summaries. However, the superior performance of the IP-based summarizer improvement is not simply due to its ability to extract smaller segments. When we compare

the IP-based summarizer to the summarizer trained on 250ms pause-based segments, we see a considerable difference in f-measure. Note that, while this pause-based segmentation does operate on more units than the sentence-based summarizer (there are 1.6 250ms-pause-based segments for each sentence), the sentence-based results are slightly better. Thus, merely extracting shorter segments does not necessarily improve summarization performance. Using more linguistically meaningful units, even when they are somewhat errorful, provides the best summarizer performance.

Note that, due to potential error in automatic story segmentation, there may be a different number of automatic and manual stories for each show. In fact, there are more than 4 times as many manually annotated stories as those derived from automatic segmentation. Therefore, in order to evaluate the performance of the summarizer against human summaries, we compared summaries of entire shows against one another, rather than performing a story-by-story comparison. Thus, the target summaries for each of the twelve training shows were constructed by concatenating human summaries of manually-transcribed and manually-segmented stories. The automatic summaries of each show are concatenated automatic summaries of each automatically-segmented story.

We also compared the results using ROUGE measure. We note that the improvement in summarization is even more pronounced when we use ROUGE for evaluation. IP-based summarization ROUGE scores are higher than those obtained by extracting 250ms pause-based segments – 13.5% using ROUGE-1, 8%, using ROUGE-2 and 14.4% using ROUGE-L. Moreover, IP-based summaries are dramatically better than sentence-based summaries. The improvements seen in summarization scores,

f-measure, ROUGE-1, ROUGE-2 and ROUGE-L confirm that hypothesized IPs are an excellent unit for extractive summarization of BN.

Segmentation	ROUGE-1	ROUGE-2	ROUGE-L
250ms Pause	0.437	0.103	0.415
500ms Pause	0.440	0.128	0.412
Sentence	0.394	0.096	0.377
Intonational Phrase	0.572	0.183	0.559

Table 5.11: Speech Segments and ROUGE scores

Based upon a comparison of four types of input segmentation, sentences, two pause-based segmentations, and IP segmentation, and using summarizers trained on the same corpus and using the same features, we found that IPs are the best candidates for extractive summarization, improving over the second highest-performing approach, sentence-based summarization, by 8.2% f-measure. We attribute the superior performance of the IP-based summarizer to the fact that IPs are shorter than sentences in our corpus but are more linguistically and semantically meaningful units than simple pause-based segments.

5.4 Additional Challenges

In our discussion in Sections 5.1, 5.2, and 5.3, we addressed the 3 basic challenges a speech summarizer faces compared to text-based summarizers: word errors, disfluency and speech segmentation. One should note that this is not an exhaustive list of all the obstacles a speech summarizer faces, but these are three additional problems

that are essential to be addressed by any successful speech summarizer that text summarization system do not have to deal with. We saw that addressing the word errors using ASR confidence scores improved the clustering of documents in same topics which should help summarizer. We presented a phrase-based machine translation approach for disfluency detection, which improved speech summarizer, though only slightly. Then we saw that the choice of speech segmentation makes a considerable difference in speech summary, and we presented the results that showed that speech summarizer based on intonational phrases is the best. We next show how we can extract additional information available in spoken documents.

Chapter 6

Information Extraction for Speech Summarization

There are some instances of entities and relationship types such as anchors, reporters, commercials that are found only in BN. These relations that do not exist in text documents can provide an added benefit in computing sentence significance for summaries. Sentences spoken by an anchor have higher likelihood of being included in a summary. Commercials should be excluded from the summaries. Soundbites spoken by important speakers potentially are useful for summarization. We would like to use information provided by these relations in our summarization system; but in order to do so we must first identify these relations. In this chapter we explain these relations and our methods to automatically identify them in BN.

6.1 Speaker Role Detection

The primary speakers in Broadcast News (BN) are news ANCHORS. Anchors introduce stories which are generally presented by REPORTERS. The roles they fulfill are reflected in their speaking style and content. Anchors read prepared transcripts while reporters speak more informally. We would like to automatically detect anchors and reporters in BN to improve our BN summarization system.

There has been some previous work done in speaker role detection. Barzilay et. al. [Barzilay et al., 2000] built a maximum entropy model and a Boostexter model to perform a three-way classification of speakers in English BN. They used key words, context, duration features and explicit speaker introductions to distinguish among speaker types, obtaining classification accuracy of about 80%. Liu [Liu, 2006] constructed a maximum entropy model for distinguishing among the same speaker types in Mandarin BN, reporting comparable accuracy by combining language model scores trained for each speaker type.

There is also considerable research on speaker diarization, the segmentation of spoken corpora into distinct speakers and the clustering of such segments into 'same speaker' clusters; but this work does not in general attempt to identify individual speakers or their roles.

6.1.1 Approach, Experiment and Results

We viewed our speaker role detection as a classification problem where we need to classify each sentence into one of the classes - anchors, reporters or others. We had

Class	Precision	Recall	F-Measure
ANCHOR	0.912	0.901	0.906
REPORTER	0.69	0.756	0.721
OTHER	0.668	0.552	0.604

Table 6.1: Speaker Role Detection Results Using C4.5 Decision Trees

manual labels for speaker roles for all the sentences in CBNSCII corpus. In order to identify the best training algorithm for speaker role classification we performed 10-fold cross validation experiments using a set of training algorithms including Bayesian Networks, Decision Trees and Support Vector machines. We found that for the task of speaker role identification decision trees based on C4.5 training procedure performed the best. One of the reasons decision trees may have done better than other algorithms such as Bayesian Network is that many of the acoustic features are real values, and real values need to be discretized for training Bayesian Networks to compute conditional probability tables (CPTs) for the nodes, thus losing some information on the process.

We trained a C4.5 decision tree model from our labeled data. We allowed pruning of the trees to take account of over-fitting as our dataset is not very large and we did not restrict binary splitting of nominal attributes. The results shown in Table 6.1 are for 10-fold cross validation results. We obtain a very high accuracy of 83.24% with a high f-measure for anchor detection of 0.90. We believe this is one of the best results reported for anchor detection with the use of ASR transcripts in the speaker role detection literature. We performed feature selection with information gain criteria for training the model. Automatically extracted features such as intensity, speaker

Pred Anchor	Pred Reporter	Pred Other	
2561	239	43	Anchor
198	817	66	Reporter
50	128	219	Other

Table 6.2: Confusion Matrix for Speaker Role Detection Experiment

cluster information, turn position and named entities were some of the most useful features.

We should note that f-measure for reporter and other classes are significantly lower. The data is skewed with a higher percentage of anchors than any other classes, so availability of more training samples for anchor classes is probably the reason anchor detection does the best. This is evident if we look at the confusion matrix in Table 6.2. The high distribution of Anchor classes is expected as most of the news are reported by anchors.

The speaker role detection system we have built will be used to automatically identify the roles for each sentence of the story we are summarizing. We explain the results on using speaker roles for summarization in Section 7.3.

6.2 Soundbite Detection

Frequently, anchors and reporters introduce segments of speech by others, which support a news story. These speakers may be interviewed, or clips of their speech (e.g. a speech or interview quotations) may be included in the newscast. These clips, when they can be identified by speaker, are of considerable value in news corpora, since

physicist <SOUNDBITE-SPEAKER> richard seed < /SOUNDBITE-SPEAKER>wants to open a clinic in chicago unlike any other in the world to clone humans. but there is a huge cost. <SOUNDBITE> i'm talking to some people, and i have enough to get started, but not enough to finish the project. <SOUNDBITE>

Figure 6.1: Soundbite and Soundbite-Speaker Example

they contain material representing views that are clearly and directly attributable to the speaker, rather than third party commentary. We term such segments as soundbites; we term the speakers of such material SOUNDBITE-SPEAKERS. In Figure 6.1 the turn within the tags <SOUNDBITE> and < /SOUNDBITE> is a soundbite.

To our knowledge, no research has yet been done on soundbite detection in BN. In the literature, speaker role detection is perhaps the most relevant to our task [Barzilay et al., 2000, Liu, 2006]; such work attempts to classify speech segments as to the **type** of speaker (anchor, reporter or other) producing the segment. A large proportion of the category 'other' is comprised of our soundbite-speakers. Anchors tend to speak often in a broadcast and any individual soundbite-speaker will tend to occur very infrequently in a single newscast.

We performed our experiments on a subset of the TDT2 BN corpus [LDC, 2002]. All of the features we extract from the corpus are extracted from these ASR transcripts, except for the turn segmentation. This is based on the manual segmentation of the human transcriptions, automatically aligned with the ASR transcripts. Note that we use the automatic cluster ids generated for TDT2 in computing features such

as the distribution of speaker turns in broadcasts.

6.2.0.1 Approach

We trained a statistical model that would classify each turn as to whether it is a soundbite or not. We built a Conditional Random Field (CRF) based model. We define our CRF with the following parametrization. Let $\mathbf{o} =< o_1, o_2, ..., o_T >$ be the observation sequence of turns in each broadcast show. Let $\mathbf{s} =< s_1, s_2, ..., s_T >$ be the sequence of states. The values on these T output nodes are limited to 0 or 1, with 0 signifying 'not a soundbite' and 1 signifying 'a soundbite'. The conditional probability of a state sequence s given the input sequence of turns is defined as

$$p_\wedge(\mathbf{s}|\mathbf{o}) = \frac{1}{Z_o} exp\left(\sum_{t=1}^{T}\sum_{k}\lambda_k f_k(s_{t-1}, s_t, \mathbf{o}, t)\right)$$

where Z_o is a normalization factor over all state sequences and $f_k(s_{t-1}, s_t, \mathbf{o}, t)$ is an arbitrary feature function. λ_k is a weight for each feature function. The normalization factor Z_o is obtained by summing over the scores of all possible state sequences:

$$Z_o = \sum_{s \in S^T} exp\left(\sum_{t=1}^{T}\sum_{k}\lambda_k f_k(s_{t-1}, s_t, \mathbf{o}, t)\right)$$

This can be computed efficiently in our case using dynamic programming, since our CRF is a linear chain of states as shown in Figure 6.2.

We used Prosodic/Acoustic, Structural and Lexical features to identify soundbites in our corpus. Prosodic/Acoustic features are useful for detecting soundbites in BN, since change in pitch, amplitude or speaking rate often differentiate between speech segments produced by various speakers.

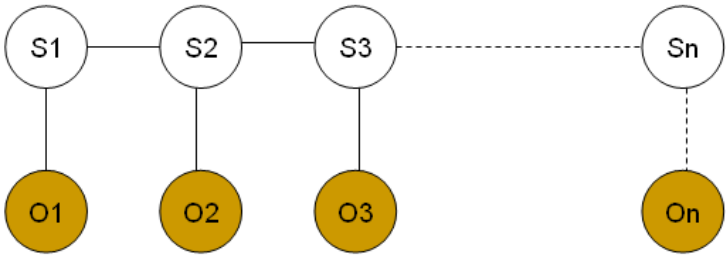


Figure 6.2: Linear chain CRF for Soundbite Detection

We further hypothesize that a turn uttered by a soundbite-speaker may exhibit different acoustic features from a turn spoken by an anchor or reporter, due not only to changes in speaking style but also variation in signal quality and the background noise of the soundbite recording. While anchors and reporters are often recorded in the studio, soundbites are generally recorded in the field (and often spliced in for an interview) or are cut from other recorded events. So, recording conditions vary considerable for soundbites, and acoustic features may capture some of this variation and thus aid in prediction. Our Prosodic/Acoustic feature-set includes features described in Section 3.2. We hypothesize that our generally non-professional soundbite-speakers may have a different speaking rates from the trained speech of anchors and reporters, based on Bolinger's description of newscaster speech [Bolinger, 1982]. Similarly, our pitch and energy features are motivated by the possibility that these may vary differently for trained anchors and reporters. On average, 'turn duration' for soundbites was 26 seconds shorter than non-soundbite turns so we use duration feature as well.

The structural features we investigated for our current study include **normalized position of turn** in the broadcast; **turn position** in the show; **speaker distribution**; **previous and next speaker cluster id**; and **top-ranking speakers** in the

broadcast.[1] The positional feature encodes where the current turn is in the broadcast; soundbites, for example, rarely occur at the beginning or end of a broadcast. 'Speaker change' is a binary feature, indicating whether the current speaker is different from the previous one.

The 'speaker distribution' feature captures the percentage of turns belonging to a given speaker in the broadcast, as calculated from the automatic speaker clustering information (i.e., identification of individual speakers by unique identifier for each segment) provided in TDT2. We hypothesize that the overall percentage of turns for any soundbite-speaker should be very low compared to the anchors and reporters; note however that, with automatic speaker clustering, we are adding a degree of noise here. The information on identity of previous and next speaker ids should also help in identifying soundbites, as there is a very low probability of two soundbites occurring together. Soundbites are usually followed by anchor or reporter comments. 'Top-ranking speakers' indicate whether the current speaker is among the top 3 speakers in the broadcast in the number of turns produced; this feature is intended to rule out anchors in particular as speakers of a soundbite.

We extract all of our lexical features from the ASR transcript, with no capitalization or punctuation. Our lexical features include **number of words in the turn**, **cue phrases**, and **distribution of cue words**. Cue phrases are currently identified by inspection of the soundbite turns as well as the turns that precede and follow them. These can be important cues for a turn transition. Since we train and test on

[1]We have already described some of these features in Chapter 3. In this section we only explain features particular to soundbites that have not be explained before.

shows that are primarily CNN Headline News, our cues are dependent on the type of cue phrases used in these broadcasts. Cue words and phrases such as anchor names, "headline news", "reporting from" are all useful in indicating an upcoming soundbite.

6.2.0.2 Experiment, Results and Discussion

To classify segments as soundbite segments or not, we built CRF models using the Mallet tool [McCallum, 2002]. This tool allows us to build CRF models with varying degrees of Markov order. In order to test the effect of previous context, we built CRF models with a Markov order of 0, 1 and 2 and compared them to MEMM models. Figure 6.3 compares the performance of the different models.

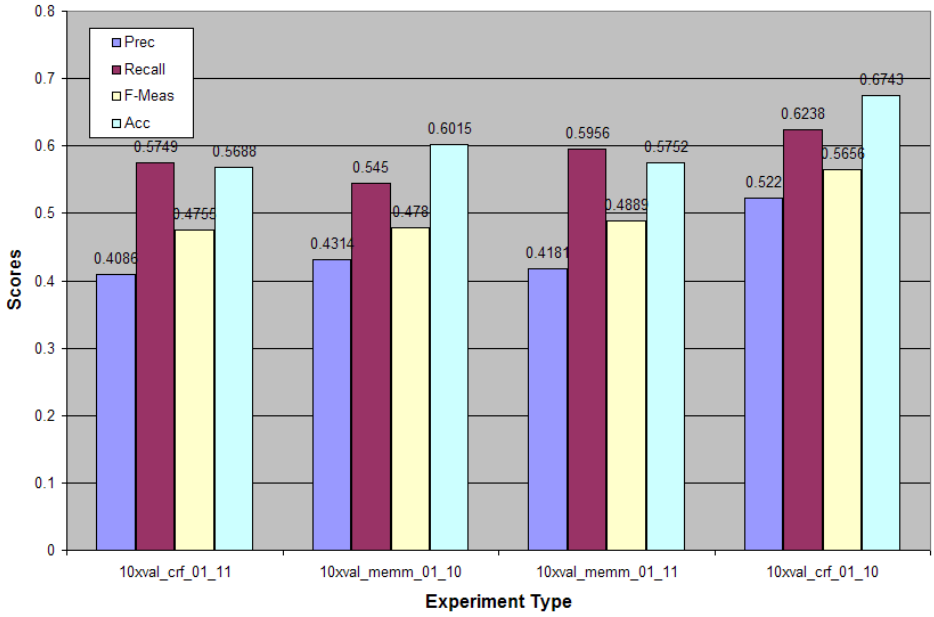


Figure 6.3: 10-fold Cross Validation Results for Soundbite Detection

We note that our model significantly outperforms the baseline as shown in Table

Type	Precision	Recall	Fmeasure	Accuracy
Baseline	0.18	0.18	0.18	0.465
MEMM	0.431	0.545	0.478	0.602
CRF	0.522	0.624	0.566	0.674

Table 6.3: Soundbite Detection Results

6.3. Our baseline model is randomly assigning labels to turns such that distribution of soundbite labels closely matches the distribution of training data. We do not use a "majority class baseline" because it would result in a baseline with a recall and an f-measure of 0, since the data has a very skewed distribution with very few soundbites. The CRF 10-fold cross-validation results are significantly higher than the baseline. This f-measure is 38.56% higher than baseline and the best performing iteration has an f-measure 64.7% higher than the baseline. Similarly, recall and precision for the CRF model is 45.38% and 34.2% higher than the baseline respectively.

The first model on the left in Figure 6.3 is a CRF model with a markov order of 1, with the observation conditioned both on the parent state and the previous parent. The second model is maximum entropy model where the observation is conditioned on the parent state only and the current state is dependent on the previous state. The best model, shown on the right of the figure, is a 1-order CRF model with the current state depending only on the previous state. Intuitively, we would assume that for a task such as soundbite detection, a higher order model would do better. However, our experiments showed that a 2-order degrades overall performance.

With the 1-order model, we obtained 67.43% accuracy in soundbite prediction

with precision of 0.522, recall of 0.624 and an f-measure of 0.566. For this model, the maximum accuracy we obtained in a single iteration was 85.9%. For the same iteration we obtained a precision of 0.816, recall of 0.838 and f-measure of 0.827, which were also the highest among all the iterations. The lowest accuracy and f-measure for this model were 55.1% and 0.394. The difference in results between best and worst iterations was thus quite high, with a difference of 30.8% in accuracy and 43.3% in f-measure.

The significant difference in the best and the worst iteration of the cross-validation experiment for the soundbite detection suggests either that the variability in the soundbites is quite high or that our annotation is somewhat noisy. We suspect, anecdotally, that both these possibilities are true. Our annotators reported some difficulty in labeling the data.

In order to determine whether conditioning the observations on more context would improve performance, we built a CRF model in which the observations were conditioned on previous states. The model generated with such conditioning did worse than CRF models that conditioned only on the parent state. The f-measure on 10-fold cross validation f-measure was lower by 9.01%, recall was lower by 4.89% and precision was lower by 11.34%. Such a difference in performance shows that either the model is overfitting the data or that our features are not dependent across turns. We think it likely that some of our features — in particular, the acoustic features — may not be dependent on prior context, since soundbites are often recorded in completely different contexts from the rest of the broadcast, even for interviews, and later spliced in to the show.

We also built Maximum Entropy Markov (MEMM) models for soundbite detection to compare to our CRF models. CRFs are similar to MEMMs except MEMMs suffer from a label bias problem due to normalization over local features rather than over the entire sequence. The results presented in Table 6.3 show that the MEMM model does slightly worst than the CRF models. For the same Markov order and similar conditioning of features over states, the CRF model does better than MEMM by 7.28% on accuracy, 8.76% on f-measure, 7.88% on recall and 9.06% on precision.

Our experiments on soundbite detection suggest to us that this task is more difficult than the related task of speaker role labeling. To explore this hypothesis on our corpus, we built a reporter detection model with the same set of features used for soundbite detection, but we used a Bayesian Network model for training purposes. We also built a Bayesian Network model for soundbite detection and compared the results. For reporter detection, the Bayesian Net model could classify reporter vs. non-reporter segments with an accuracy of 72%, an f-measure of 0.665, precision of 0.719 and recall of 0.618. However, a similar Bayesian Net model built on the same set of features classified soundbites more poorly, with an accuracy of 67.6%, an f-measure of 0.522, precision of 0.477 and recall of 0.577. These results are considerably lower than results for reporter detection.

6.3 Commercial Detection

Besides identifying the roles of the speakers and soundbites, we also extracted commercials from BN. There has been a lot of research on commercial detection in the

video research community in which video cues are used for identifying commercials. One of the main differences between such systems and ours is that we use only speech-based cues. With the availability of the speech signal and an ASR transcript we use lexical and acoustic information to determine which segments are commercials.

We detect commercials by training a ML model to classify each sentence as belonging to a commercial or not, then obtaining the posterior probability of the classes and using this in a post-processing step we have developed that tunes the start and end of commercial boundaries according to the probability mass distribution of commercials over a given window size. Let us first describe the trained ML model and then the details of window-tuning algorithm.

We first compared training algorithms that differed significantly in their training criterion, i.e. some training methods try to maximize information gain for each feature (decision trees) while some try to create a hyperplane to minimize errors (SVMs). Our best performing algorithm was again C4.5 decision trees, so we built a decision tree model for commercial detection. After we predicted whether each sentence was part of a commercial or not using this decision tree, we used the posterior probability obtained from the decision tree to further tune the predictions.

The motivation for performing such post-processing is that the likelihood of a sentence being a part of a commercial is high if all the sentences before and after it are labeled as commercial, since commercials generally contain contiguous sentences. Hence we post-process the predictions provided by our trained model to improve our system. We call our post-processing a window tuning algorithm because we take a running window of size N over all the predictions and update the predictions for the

Class	Precision	Recall	F-Measure
NON-COMMERCIAL	0.96	0.971	0.965
COMMERCIAL	0.555	0.471	0.51

Table 6.4: Commercial Detection Results Using C4.5 Decision Trees

sentence in the middle of the window taking the surrounding sentence predictions into account.

The running window algorithm takes the sentences of window size N then sums the posterior probability provided by our statistical model. If the total average probability is higher than a threshold and all sentences before and after the current sentence in the given window belong to NON-COMMERCIAL sentences we update the class of the sentence to NON-COMMERCIAL, i.e. if there is only one sentence surrounded by non-commercial sentences then that sentence is less likely to be commercial. For our purposes we chose a window size of three.

The 10-fold cross validation results tested on ASR transcripts are reported in Table 6.4. The total accuracy of the system is 93.39%. However, we get an f-measure of 0.51 which is lower than the speaker role detection f-measure. The possible reasons for this lower performance are the fact that the data is heavily skewed, with more than 90% of the sentence in BN belonging to the NON-COMMERCIAL class. Also, the acoustic features extracted for commercials will be much noisier because of the background music that typically occurs in commercials. Another possible reason for our lower performance is that the ASR may be performing worse for commercials, thus producing noisy transcripts. A noisy transcript lowers the accuracy of our lexical

features, further reducing the quality of features used in training of the model. Hence the overall higher degree of noise in the pre-processing step is the likely cause for lower accuracy on commercial detection.

In Chapter 5 we addressed the challenges we face in speech summarization and in this chapter we showed how to automatically extract additional information only found in BN. In the next chapter we show how to combine this information to build our BN speech summarizer ConciseSpeech.

Chapter 7

ConciseSpeech: Single Document Broadcast News Speech Summarizer

Our BN speech summarizer, ConciseSpeech, summarizes a single BN document by identifying significant segments of the show, extracting them and concatenating them together to generate a summary of user-desired length. The system uses a statistically trained model to identify the importance of segments, and uses an algorithm we have developed to optimize significance, redundancy and length of summary. We describe the details of ConciseSpeech in this chapter.

7.1 System Architecture

Our fully trainable system consists of seven main components. Each of the components may include sub-modules. The first component is a pre-processing engine that processes speech signal and transcripts to extract features. The extracted features are stored in an rttmx file format. An rttmx file is a special format file which consists of a matrix where rows represent words and columns represent annotation or features. An example rttmx file is shown in Appendix A. The component router forwards the information stored in the rttmx file to the feature extraction module that extracts lexical, acoustic, discourse and structural information. The router updates the rttmx file with these features. The rttmx file is passed on to the IE module that extracts information on anchors, reporters, soundbites and commercials. After the pre-processing, feature extraction and IE stages, the rttmx file is passed on to two different modules - a clustering module which finds redundant information and a statistical estimation module which computes the significance of segments of BN. The redundancy and significance scores produced by these modules are passed on to the summary compilation module, which combines these scores to produce a final summary. We look at the details of individual components in the following sections.

The pre-processing component cleans the data, extracts relevant information and converts the data from xml and sgml formats to the common rttmx format. This pre-processing component contains many sub-modules which include XML-TO-RTTMX CONVERTER, SGML-TO-RTTMX CONVERTER, SPEECH SEGMENTATION MODULE, DISFLUENCY REMOVAL, SUMMARY LABEL EXTRACTOR, RTTMX ALIGNER. The rttmx aligner aligns the manual and ASR transcripts using the Minimum Edit Distance

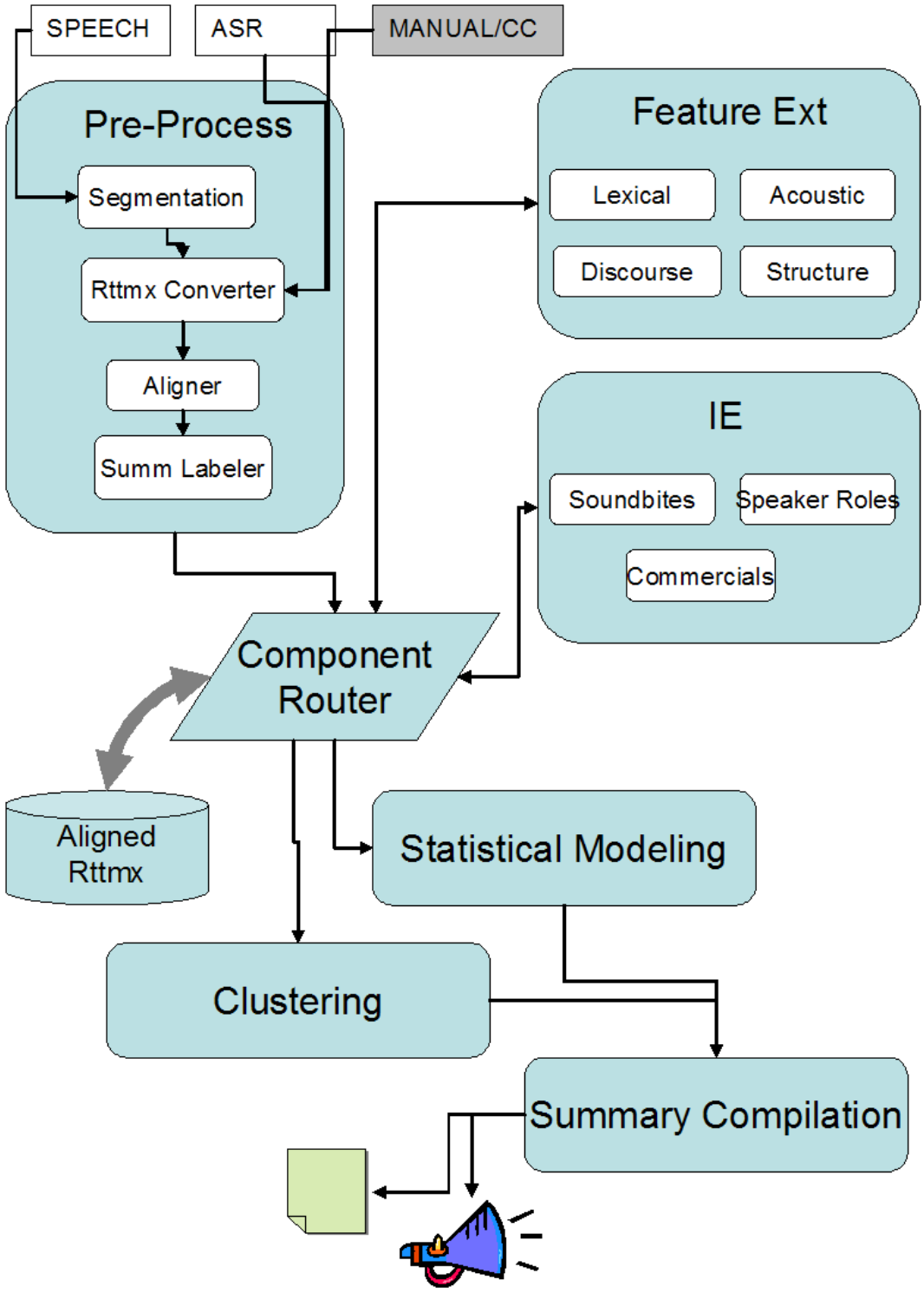


Figure 7.1: ConciseSpeech System Architecture

method that we have described in Section 2.2.1. At the end of the pre-processing phase, we have an rttmx file that contains all boundary information including sentence, turn and story boundaries.

The feature extraction component consists of four main modules which extract lexical, acoustic, structural and discourse features. This component depends on the boundary information provided by the pre-processing components, so poor speech segmentation can significantly degrade the accuracy of features.

The IE component extracts information on anchors, reporters, soundbites and commercials. We described details of each of these modules of this component in Chapter 6. After extracting information on previously mentioned relations and entities, the IE component updates the rttmx file by passing this information to the component router.

7.1.1 Component Router

In Figure 7.1 we note that the Component Router (CR) is one of the central components of the system. Instead of building a standard pipeline based architecture, we opted to build a system with a CR because we wanted our system to be easily trainable in a new data environment. We also wanted the system to be flexible enough so that many of the components can be easily modified if required. Since we are dealing with BN, the data environment can vary significantly depending on what kind of ASR engine and segmentation modules are available. Instead of directly passing information from component to component we pass information through the CR, so that it can check for inconsistencies and inform the data receiving module. Such an

information routing system also allows us to easily create multiple data paths that can be used for training and testing, which helps us to re-train our summarizer for new news channels or new data formats. For example, if we want to retrain our model with ABCWorldNews Tonight shows and we believe the headline detection system may not work well, we can easily create a path with CR that bypasses the headline detection module in the training of the system.

7.1.2 Statistical Modeling: Segment Significance

After data has passed through pre-processing, feature extraction and IE components, we have all the information we want to estimate the significance of segments for a summary. In Chapter 4 we have seen that we were able to successfully build a summarizer based on Bayesian Networks to identify the significant sentences for a summary. In this section we will provide more detail on the statistical modeling techniques that we have experimented with for summarization.

There are various statistical techniques that rank or classify segments based upon their significance, such as Decision Trees, Bayesian Networks, Ripper, Support Vector Machines (SVMs), Hidden Markov Models (HMMs) and Conditional Random Fields (CRFs), to name a few. Each of these statistical techniques has its advantages and disadvantages. Decision Trees can make very complex decisions with a fairly limited data and it is easy to interpret the graphical representation of decision trees, but the decisions are strict as the thresholds on nodes are not probability estimates. Bayesian Networks provide a mechanism to strictly define the parent-child relationship between various attributes of the data, allowing the system to realize the

relationships within data. On the other hand, Bayesian Networks cannot encode time dependent information on the nodes. SVMs have gained wide attention due to their successful application to various NLP and Spoken Language Processing problems. The training process is quite slow for multi-class SVM and it needs hand tuning of a key parameter. None of the previously mentioned classifiers take account of time dependent information. BN data has a temporal nature, i.e., the turn that is being spoken by a reporter can be affected by what an anchor said in the previous turn. Taking account of such temporality in the data should be very useful for summarization as we do not want to include any redundant information in a summary. There are a few sequence modeling techniques that have been successfully applied to NLP such as Maximum Entropy Markov Models (MEMMs), CRFs and Hidden Markov Models (HMMs). Since we have labeled data, training HMMs is straightforward with computation of relative frequencies. Hence, we explore the use of HMMs for summarization in the next section.

7.1.2.1 Using Continuous HMMs for Speech Summarization

Since BN shows are critically temporal in nature — news shows exhibit clear patterns as they unfold — we want to take advantage of various types of such patterns in our classification. Some of these arise from the temporal sequence of speaker turns or from the repeated occurrence of particular phrases before or after turns. Hidden Markov Models (HMM) have been used successfully for modeling such temporal data.

While HMMs are used in many language processing tasks, they have not been employed frequently in summarization. A significant exception is the work of Conroy

and O'Leary (2001), which employs an HMM model with pivoted QR decomposition for text summarization. However, the structure of their model is constrained by identifying a fixed number of 'lead' sentences to be extracted for a summary. We introduce a new HMM approach to extractive summarization which addresses some of the deficiencies of work done to date.

We define our HMM by the following parameters: $\Omega = 1..N$: the state space, representing a set of states where N is the total number of states in the model; $O = o_{1k}, o_{2k}, o_{3k}, ...o_{Mk}$: the set of observation vectors, where each vector is of size k; $A = \{a_{ij}\}$: the transition probability matrix, where a_{ij} is the probability of transition from state i to state j; $b_j(o_{jk})$: the observation probability density function, estimated by $\Sigma_{k=1}^{M} c_{jk} N(o_{jk}, \mu_{jk}, \Sigma_{jk})$, where o_{jk} denotes the feature vector; $N(o_{jk}, \mu_{jk}, \Sigma_{jk})$ denotes a single Gaussian density function with mean of μ_{jk} and covariance matrix Σ_{jk} for the state j, with M the number of mixture components and c_{jk} the weight of the k^{th} mixture component; $\Pi = \pi_i$: the initial state probability distribution. For convenience, we define the parameters for our HMM by a set λ that represents A, B and Π. We can use the parameter set λ to evaluate $P(O|\lambda)$, i.e. to measure the maximum likelihood performance of the output observables O. In order to evaluate $P(O|\lambda)$, however, we first need to compute the probabilities in the matrices in the parameter set λ.

The Markov assumption that state durations have a geometric distribution defined by the probability of self transitions makes it difficult to model durations in an HMM. If we introduce an explicit duration probability to replace self transition probabilities, the Markov assumption no longer holds. Yet, HMMs have been extended by defining

state duration distributions called Hidden Semi-Markov Model (HSMM) that has been successfully used [Tweed et al., 2005]. Similar to [Tweed et al., 2005]'s use of HSMMs, we want to model the position of a sentence in the source document explicitly. But instead of building an HSMM, we model this positional information by building our position-sensitive HMM in the following way:

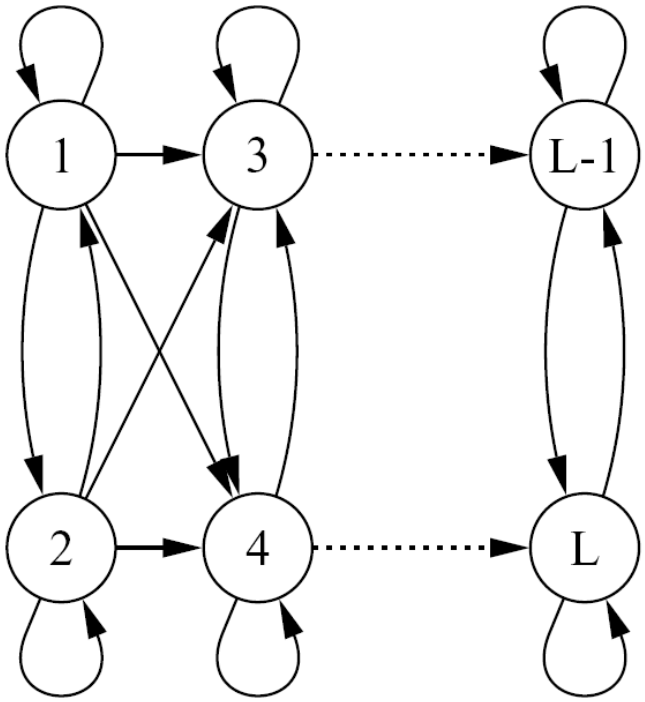


Figure 7.2: Topology of HMM

We first discretize the position feature into L bins, where the number of sentences in each bin is proportional to the length of the document. We build 2 states for each bin where the second state models the probability of the sentence being included in the document's summary and the other models the exclusion probability.

Hence, for L bins we have $2L$ states. For any bin lth where $2l$ and $2l-1$ are the corresponding states, we remove all transitions from these states to other states

except $2(l+1)$ and $2(l+1)-1$. This converts our ergodic L state HMM to an almost Left-to-Right HMM though l states can go back to $l-1$. This models sentence position in that decisions at the lth state can be arrived at only after decisions at the $(l-1)th$ state have been made. For example, if we discretize sentence position in document into 10 bins, such that 10% of sentences in the document fall into each bin, then states 13 and 14, corresponding to the seventh bin (i.e. all positions between 0.6 to 0.7 of the text) can be reached only from states 11, 12, 13 and 14. The topology of our HMM is shown in Figure 7.2.

We trained a continuous HMM model described above using acoustic features extracted from CBNSCI corpus. We obtained encouraging results with f-measure of 0.41 with 8-state HMM.[1]

ModelType	Precision	Recall	F-Meas
HMM-8state	0.26	0.95	0.41
HMM-2state	0.27	0.79	0.40

Table 7.1: Speech Summarization Using HMM with Acoustic Features

Even though we were able to use a continuous HMM for summarization with acoustic features it is challenging to include lexical and structural features in our HMM framework. Our HMM which is based based on Gaussian mixtures assumes that observation features can be parametrized using a normal distribution. Such restrictions can be limiting for summarization purposes, where lexical and structural features can be on nominal and binomial distributions. In order to overcome this

[1]The details of the experiments and results are available in [Maskey and Hirschberg, 2005].

limitation on the use of features by the type of modeling, we used a modeling technique that can easily integrate various kinds of features such as Bayesian Networks.

7.1.2.2 Bayesian Network for Summarization

Given a set of features the Bayesian Network identifies the significant segments by computing a joint probability over all the features and the class. We have a big feature set with lexical, acoustic and structural features. In order to make the computation of joint probability tractable Bayesian Network decomposes the joint probability over all features into conditional probabilities according to the structure of the Bayesian Network.

Given a Bayesian Network $G = \{X, E\}$ where $X = \{x_1, x_2, ..., x_n\}$ are nodes and $E = \{(x_i, x_j) : i \neq j\}$ are arcs, we can factorize the joint distribution over all the variables into the product of local terms as follows:

$$P(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} P(x_i|\pi_{x_i}) \tag{7.1}$$

where π_{x_i} are parents of node x_i and $P(x_i|\pi_{x_i})$ is the conditional distribution of x_i, given its parents π_{x_i}. Here the x_i's represent our acoustic, lexical and structural features.

One of the reasons a Bayesian Network is suitable for our purpose is because it allows us to encode the relationship between features using the network structure. For example the structural feature's relationships can be encoded with a network structure as shown in Figure 7.3. In this network, ASRSENTID (sentence position) is dependent on the role of the speaker and the ASRENTIDNORMPOS (normalized

sentence position) is dependent on unnormalized sentence position.[2]

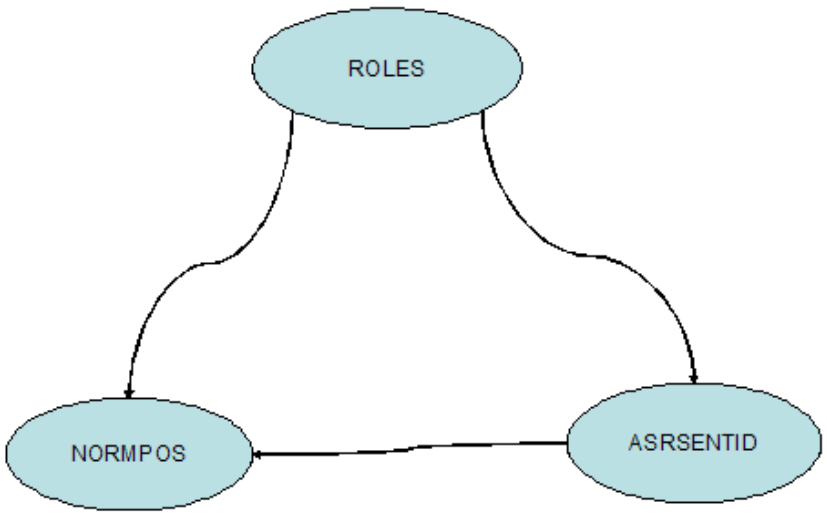


Figure 7.3: A Subgraph of Bayesian Network Graph for ConciseSpeech

After we find the topology of the graph represented in Figure 7.3, we only need to find values of the parameters of the Conditional Probability Tables (CPTs) at each node. We can estimate the parameters of the CPT by finding Maximum Likelihood Estimates (MLE). If we let θ_i be a parameter vector for the CPT of node x_i and define $\theta(x_i, \pi_{x_i}) = P(x_i|\pi_{x_i}, \theta_i)$ then we can find parameter values using MLE as follows:

$$\theta(x_i, \pi_{x_i}) = \frac{m(x_i, \pi_{x_i})}{m(\pi_{x_i})} \tag{7.2}$$

where $m(x_i, \pi_{x_i})$ represents the number of times the node and its parents are jointly in a particular configuration and $m(\pi_{x_i})$ represents the number of times the parents are in that configuration.

[2]The figure only shows a small portion of the graph, since the original graph with all nodes is too large to display here.

For our experiments, since we have continuous features mixed with nominal features, we first converted all the continuous features into discrete features so that we could compute CPTs for each node where node is a feature. We then performed 10 fold cross validation on all our data of CBNSCII corpus. We obtained f-measure of 0.456, precision of 0.328 and recall of 0.749 using lexical, acoustic and structural features. The predicted labels and their posterior probability were then passed on to our optimization algorithm to generate the final summaries. We describe the optimization algorithm in the following section, and further discuss our results in Section 7.3.

7.2 Optimizing Redundancy, Significance and Length for Summary Generation

A summarizer should be able to provide a summary in user's target summarization ratio and should remove all redundant information. In order to build a summarizer we believe we need to optimize the computation of three functions: i) significance, ii) redundancy, and iii) length. Our summary generation algorithm include three modules corresponding to these functions, which we combine to produce an optimal summary. The algorithm uses a statistical method to compute significance scores, which are interpolated with centroid scores. These scores are used to extract significant but non-redundant segments until a user-defined summarization ratio is reached. We have already described our Bayesian Network based statistical model that computes segment significance. Here we describe our modules that compute centroid of a

document, cluster the segments to find redundant information and the compiles the segments to produce a summary respectively.

7.2.1 Computing the Centroid of a Document

Most BN news stories have one central topic and various subtopics related to the main topic. Finding the main topic and subtopics is important because it allows the system to keep the summary focused on a central theme without losing relevant information on subtopics. This concept has been explored previously by finding the centroid of a document or of a set of documents. Various methods have been proposed to compute the centroid of a document. [Radev et al., 2000] assumes that the centroid is a set of words in the document that has TF·IDF scores above a certain threshold. They hypothesize that the sentences that contain the words from a centroid are indicative of the topic of the document. We compute the centroid of a document by this method. We first create a document vector where each dimension is a unique word. We next compute the TF·IDF measure for all the words in the document. Using a threshold t we remove all the words below a given threshold estimated empirically. The remaining document vector is a centroid vector of the document. We next compute a centroid score for each sentence in the document by computing the similarity between the centroid vector and the TF·IDF vector of each sentence using a cosine similarity measure we described in Section 3.1. We represent the sentence of a document as a TF·IDF vector, i.e.

$S_i = t_{f1}log(N/d_{f1}), t_{f1}log(N/d_{f1}), ..., t_{f_n}log(N/d_{f_n})$

where t_{fi} is the term frequency of ith term in the sentence, N is the total number

of documents in the corpus and d_{fi} is the inverse document frequency, i.e. the number of documents the term appears in.

7.2.2 Combining Significance and Centroid Scores

After computing the centroid of the document we need to rank the significant sentences for the summary based on the proximity to centroid and the significance rankings obtained from our trained model. We first obtain the significance score $S_p$for each sentence based on the predictions of our statistically trained model. We obtain $S_p = S_{p1}, S_{p2}, ..., S_{pn}$ for all the sentences. We then obtain the centroid scores for all n sentences $S_c = S_{c1}, S_{c2}, ..., S_{cn}$. We combine the scores S_{pi} and S_{ci} for all sentences as shown in Equation 7.3.

$$S_i = w_c * S_c + w_p * S_p \tag{7.3}$$

We choose the weights w_c and w_p manually. For our experiments we chose w_c of 0.5 and w_p of 0.5. The scores S_i for each sentence can be used to extract the best sentences for the summary.

7.2.3 Compilation

Compilation is the last step in the algorithm. Compilation 'compiles' the significant sentences taking account of following constraints i) sentence order, ii) redundancy removal, and iii) user-defined summary length limit.

One of the disadvantages of using a centroid based method is that two redundant sentences can get equally high significance scores if both are closely related to the

central topic/theme of the document. In order to compile a summary without redundancy but still include the relevant significant sentences, we built an algorithm which we describe next.

Let us define m_{ij} to be the Similarity score between sentence i and j. Similarity score is computed by computing word overlap between two sentences. If we have n sentences in the story we get an n by n square matrix of similarity scores. A high similarity score m_{ij} would signify redundant information between two sentences i and j. We should note that we compute similarity between sentences based on lexical information only. We do not use any acoustic information in computing similarity because it is not evident on how to use acoustic information such as pitch and intonation in computing segments similar in their content. Given a summarization ratio R we need to extract L sentences such that L/N is approximately R where N is the total number of sentences. If we simply choose a threshold t and extract the sentences with significance scores higher than the threshold then we will not be able to control the length of the summary. If we sort the sentences by the scores and extract the top sentences to match the summarization ratio R we we may have redundant sentences. Instead we use the algorithm to compile the sentences for the summary as shown in Figure 7.4. Let $SimSentSet$ be a set of sentences (i, j) that are similar to each other with a $m_{ij} > T_m$ where T_m is a threshold of similarity.

The algorithm described in Figure 7.4 compiles the sentences with highest significance scores until the summarization ratio is reached. It does not include any sentence that is similar (redundant) enough to any other sentence already in the summary set $SUMM$. It then reorders the compiled sentences to the original document position

```
S= set of segments in document;
N = total number of sentences;
R= summarization ratio
SUMM = {} //set of segments in summary
sort(S) by significance score
L = 0; i=0
while ( (L/N) < R) )
  S_to_Add = Highest(SimSentSet that contain S_i)
    if (! (SUMM contains S_to_Add) )
           SUMM = SUMM.add(S_to_Add)
       L = L + Length(S_to_Add)
    i++;
end
sort(S) by position
```

Figure 7.4: Algorithm for Summary Compilation

to generate the final summary.

7.3 Evaluation and Discussion

We evaluated our summarizer, ConciseSpeech, using automatic evaluation tools and manual methods. For automatic evaluation, we used standard IR measures (Precision, Recall, F-measure) and ROUGE [Lin, 2004]. For manual evaluation, we followed

the guidelines provided for human evaluators for the Document Understanding Conference (DUC) 2007 and asked human judges to rate our summaries on a scale of 1 to 5 to evaluate the responsiveness of the summary to the topic of the story.[3]

We previously asked human annotators to produce summaries that were 30% of the original length, so for all our experiments we set the summarization ratio to 0.3. We first produced summaries using our baseline summarizer. The baseline summarizer extracted the first 30% of the sentences and concatenated them to produce a summary.[4] We computed ROUGE scores for these summaries by comparing the baseline summaries with human generated summaries of annotator 1 (ANN1), annotator 2 (ANN2) and both annotators. The results are shown in Table 7.2. We obtain ROUGE-2 scores between 0.38 to 0.40 and ROUGE-L scores between 0.46 to 0.5. A ROUGE score in the range of 0.4 is rather high for a baseline. One possible explanation for this high baseline ROUGE score is that the anchors in BN tend to introduce stories by providing important information in the beginning of the story, and our stories are relatively short compared to newswire stories.

After we computed our baseline summaries, we built ConciseSpeech summaries. First we split CBNSCII corpus into a 90% training set and a 10% test set. For the sentences in the training set, we extracted the lexical and acoustic features described in Chapter 3. We had the manual labels from our annotators for all the sentences as to whether they should be included in the summary or not. We used the manual

[3]More information on the DUC 2007 guidelines can be obtained from http://duc.nist.gov/duc2007/tasks.html.

[4]The tested story\summary pairs are available on the web at http://www.cs.columbia.edu/~smaskey/thesis.

Type	Annotator	Rouge-2	Rouge-L
Baseline System	Annotator 1	0.402	0.505
	Annotator 2	0.385	0.469
	Annotator Both	0.402	0.498
ASR Based System	Annotator 1	0.487	0.629
	Annotator 2	0.431	0.555
	Annotator Both	0.470	0.608
ASR with Automatically Extracted Relations	Annotator 1	0.493	0.634
	Annotator 2	0.437	0.561
	Annotator Both	0.474	0.612
Manual Transcript Based System	Annotator 1	0.664	0.720
	Annotator 2	0.524	0.580
	Annotator Both	0.607	0.666

Table 7.2: Rouge Results for ConciseSpeech

labels provided by ANN1 for training our summarizer. Using the manual labels of ANN1 and features extracted, we trained a Bayesian Network classifier as described in Section 7.1.2.2. The classifier predicted whether the given sentence with its lexical and acoustic properties should be included in a summary with a probability S_p. We took the label and the posterior probability of the classifier S_p and passed them to our optimization algorithm with a summarization ratio of 0.3 and centroid scores S_c. The optimization algorithm selected the most significant but non-redundant sentences until it reached the summarization ratio of 0.3. The final summaries generated by the

summarizer were then evaluated using ROUGE. We again compared the automatically generated summaries with the human generated summaries of ANN1, ANN2 and both of the annotators. The results are shown in Table 7.2. We obtained ROUGE-2 and ROUGE-L scores of 0.487 and 0.629 respectively. These are significantly higher scores than the baseline summarizer with an improvement of 8.5% in ROUGE-2 and 12.4% improvement in ROUGE-L scores for ANN1. These are very promising results considering that **our model is based on ASR transcripts with no manual annotations**. As we noted above, the baseline summaries are already very competitive in this domain. Obtaining such improvements over such a high baseline is very promising. If we look at the ROUGE scores in comparison with the human summaries generated by ANN2 we see that scores degrade by 5.6% for ROUGE-2 and 7.5% for ROUGE-L respectively. Even though the scores are lower than those obtained by comparing with ANN1, they are still much higher than the baseline scores, showing that our system is quite robust. This conclusion is further strengthened when we compared the automatic summaries with both annotators. ROUGE scores when evaluated with both annotators were 0.47 for ROUGE-2 and 0.608 for ROUGE-L slightly higher than our ANN2 comparison and slightly lower than our comparison with ANN1, as we had expected.

In the summarizer described above, we did not include information such as speaker role, commercials and soundbites. We described our automatic IE systems for identifying this information in Chapter 6. We automatically extracted this information for all of our sentences in training data set and added the new features to our feature set. We re-built the summarization system described above and again evaluated it in

the same setting. The results are shown in Table 7.2. The new model does slightly better than the previous one, with an improvement of 0.6% on ROUGE-1 and 0.5% on ROUGE-L for ANN1. The overall improvement on ROUGE for ANN2 is also around 0.5%. We had hoped for better results using this additional information. One of the reasons we probably did not see more improvement could be because the extraction system is also based on ASR transcripts which are noisy to start with. The predictions of these relations based on noisy input may not be good enough to add more value to the overall feature set that have already been added previously. Even though we see only slight improvement over our previous summarizer, we again beat the baseline summarizer with a wide margin of 9.1% on ROUGE-1 and 12.4% on ROUGE-2 metric.

Our fully automatic summarizer, ConciseSpeech, takes only ASR transcripts and the speech signal, and produces summaries that are promising according to the evaluation we described above. Next, we wanted to see if we could get further improvement if we used manual transcripts and manual annotation. We should note that obtaining manual transcripts and manual annotation for many of our relations is not practical for real-time systems. We built a manual-transcript based summarizer to find the highest accuracy that can be reached with our framework. We took all the manual transcripts for our stories and built our system following the method we have described above. We obtained results as shown in Table 7.2. We see a improvement from our previous summarizer that used ASR transcripts. We obtain ROUGE scores of 0.664 and 0.720 for ROUGE-2 and ROUGE-L which is an improvement of 17.1% and 8.6% respectively. We examine the story summary pairs produced by the

system based on ASR transcript and manual transcript, respectively. First, manual transcripts have no word errors unlike ASR transcripts. Second, we are comparing ROUGE scores with the baseline human summaries that were generated from manual transcripts. There are instances in the summary when the ASR based summarizer and manual transcript based summarizer both selected the same sentences but the automatic summaries generated from manual transcripts had slightly higher ROUGE scores because the manual summaries had numbers written in digits while ASR transcripts produced normalized text such as "one nine zero zero zero" for the number 19000. So the same sentence was written differently. Also, for manual transcript based summaries we used manually annotated roles, commercials and soundbites. These annotations can be very useful when they are correct. For our ASR based summarizer we used automatically extracted relations which are not as accurate as the manual annotations.

Our automatic evaluation of ConciseSpeech showed promising results. Our summarizer performed better than the baseline summarizer. We would like to see if we obtain similar improvement when evaluated by humans. We performed manual evaluation of our summaries. We used Amazon Mechanical Turk to obtain human evaluators to rate our automatically generated summaries. Amazon Mechanical Turk gives an experimenter access to a pool of human annotators to perform various tasks over the web.[5] The experimenter can give an assessment test to the annotators to exclude annotators who may not fit the profile for the experiment. For our manual evaluation we randomly sampled 16 stories from our test set. We obtained baseline

[5]Wisam Dakka helped us to setup the evaluation experiment in mturk. www.mturk.com.

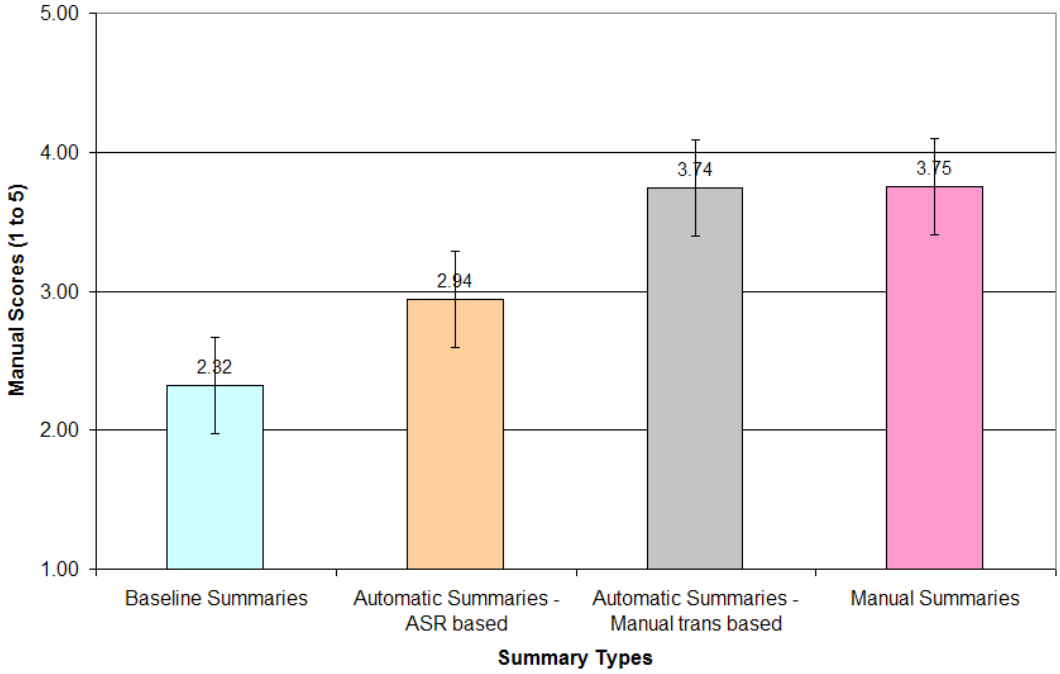


Figure 7.5: Manual Evaluation Scores for the Automatically Generated Summaries

summaries, automatic summaries based on ASR transcripts, automatic summaries based on manual transcripts and the summaries prepared by human annotators. We then asked the humans to rate the summaries on a scale of 1 to 5 with the guidelines similar to DUC 2007.[6] Basically, humans had to assess how well the summary responds to the main topic of the story. The human evaluators would see a story and then a corresponding summary which they would read and score. We had 5 or more human evaluators perform this task, so we had multiple scores ranging from 1 to 5 for each summary of each story. For each summary set we computed the average

[6]The guidelines we provided to our annotators are available at www.cs.columbia.edu/~smaskey/thesis along with the data.

human score, as shown in Figure 7.5. We obtained an average score of 2.32 for baseline summaries. If we compare this average with automatic summaries generated by ConciseSpeech using ASR transcript, we get an improvement of 12.44% for a score of 2.94. The improvement using human evaluation is consistent with our automatic evaluation results that showed similar improvement. Our summarizer based on manual transcripts performed better than the one based on ASR transcripts. We obtained an improvement of 16%. We had seen a similar improvement of 17.1% and 8.6% using ROUGE-2 and ROUGE-L metrics, making our results consistent across automatic and human evaluation. For our automatic evaluation, ROUGE scores were computed by using ANN1 summaries as our gold standard. Even though the ROUGE scores between different systems are not directly comparable due to the lack of common training and testing set, when we compare our ROUGE scores they compare favorably to ROUGE scores of other systems. For example, ROUGE scores for systems for conversational speech [Zhu and Penn, 2006] and meetings [Murray et al., 2005b] also have similar score range of 0.4 to 0.65.

For our manual evaluation we also made the human summaries of ANN1 compete with our automatic summaries, i.e. we asked the evaluators to rate the human summaries without telling them that they are manually generated human summaries. We found that our automatically generated summaries were almost as good as human generated summaries in the ratings of evaluators. The human summaries were only 0.14% better than the summaries generated by our summarizer based on manual transcripts. This observation further strengthens the findings in summarization research that two humans may summarize the same story differently. Our human evaluators

did not have clear preference to human summaries over our automatic summaries even though our automatic summaries had only ROUGE scores of 0.63.

There are many directions that we can follow to improve the overall summarization accuracy of ConciseSpeech. One of them is exploring new sets of features based on acoustic signal and lexical information. Even though we have used many variations of intensity, pitch, speaking rate we have not explored many other features that have been used in speech recognition, particularly the cepstrum features. Also, lexical chains have been used in text summarization [Barzilay and Elhadad, 1997]. We can find lexical chains in noisy ASR transcript and use that information to improve the module for computing segment significance.

Another research avenue that we have not explored in this thesis is the use of syntactic information for summarization. Even though syntactic parsing is noisy for ASR transcripts, sentence compression techniques have been shown to be useful by [Galley and McKeown, 2007] for compressing text based sentences. Following [Galley and McKeown, 2007] method we can compress ASR sentences to shorten the sentences before we select the significant sentences. Compressing the sentences before we make the final selection allows us to better tune the length of the summaries providing better flexibility in the compression ratio of the summaries.

In this chapter we presented our BN summarizer, ConciseSpeech, and we evaluated the system using the ROUGE evaluation measure. Our fully automatic summarizer based on ASR transcripts and speech signal was able to improve over the baseline by a wide margin. We saw a further improvement with the use of manual transcripts. In the next chapter we show how we can extend ConciseSpeech to handle user queries.

Chapter 8

Extending ConciseSpeech: Query-Focused Speech Summarization

ConciseSpeech takes BN as input and produces a summary that users can browse. The general-purpose summary produced by ConciseSpeech does not take into account the user's preference on topics and stories. We would like to extend ConciseSpeech such that it can take a user's query as input and produce a summarized answer relevant to the query.

We need to perform some changes to ConciseSpeech to build such a query-focused speech summarizer. First, we need to add a module that analyzes a query to extract relevant information. Second, we need an information retrieval engine that can search a corpus of BN documents. Third, we need to change the selection criteria for computing the scores that are used to select significant sentences. Fourth, we need to identify

redundant information across multiple documents instead of one document. These additional problems make query-focused speech summarization a harder problem. In the following sections we describe the modules we added to extend ConciseSpeech to build a query-focused summarizer.

8.1 Query-Focused Speech Summarization

One of the motivations for extending ConciseSpeech to handle queries was our participation in the Global Autonomous Language Exploitation (GALE) project. GALE is a large-scale multi-institution project that allows the users of the system to search for answers to a given set of question types. GALE is an ideal test bed for testing query-focused speech summarization because the corpora for GALE include a large set of spoken documents as well as text documents. One of the goals of GALE is finding and distilling (summarizing) information in the corpus to produce answers for a given type of question. There were 14 question types for year 1 in GALE, and 17 question types for year 2. Question types varied from finding relations between two entities to producing biographies. We focused on producing answers for a template that seeks statements made by a person about a topic (Template 5).

Let us look at an example of a query in Figure 8.1 that our query-focused summarizer needs to handle. The query contains two main arguments: a person argument and a topic argument. The query also has a $< context >$ argument that allows us to narrow the search to a given source type. Users can modify the $< source-type >$ argument in the query to constrain the search in the subset of corpus that only contains

```
<query template-number="5" query-id="GNG012">
 <query-text>FIND STATEMENTS MADE BY OR ATTRIBUTED TO [Osama bin Laden]
  ON [the September 11, 2001 terrorist attacks on the United States]
 </query-text>
     - <query-arg arg-num="1" arg-type="Person">
          <arg-value>Osama bin Laden</arg-value>
  </query-arg>
    - <query-arg arg-num="2" arg-type="Topic">
          <arg-value>the September 11, 2001 terrorist attacks on
          the United States
          </arg-value>
  </query-arg>
    - <context>
        - <source-type>
          <source>AFA</source>
          <source>ALR</source>
          <source>ALJ</source>
       </source-type>
  </context>
</query>
```

Figure 8.1: Sample Template 5 GALE Query

English BN.

The overall architecture of our query-focused summarization system is shown in Figure 8.2. The query analyzer extracts relevant and important terms from the query and finds the NEs that are the main focus of the query. After finding the main NEs we estimate the number of documents to be retrieved. The Indri Document Retrieval engine [Strohman et al., 2005] is used to retrieve the relevant set of documents that may contain the statements we are interested in. After the documents are retrieved, we pass the sentences of each document through a filtering module that filters out malformed sentences. After the filtering stage, we extract the lexical features we described in Section 3.1. These features are passed on to a scoring module. The scoring module uses a set of rules to give weights to each sentence. The rules are generated manually by analyzing the data. (We could not induce the rules automatically because there was not enough data for statistical learning). The sentences are passed to a clustering module to find redundant and similar sentences. Finally the answer generation module uses the weights assigned by the scoring module and the similarity score provided by clustering module to produce an answer. We explain briefly each module below.

8.1.1 Query Analyzer

The query analyzer takes a query and extracts relevant information from the query and passes it to the IR engine. The query for finding statements has two arguments: i) the person who made the statement and ii) the topic of the statement. The query analyzer parses each of the arguments and extracts the person argument to extract

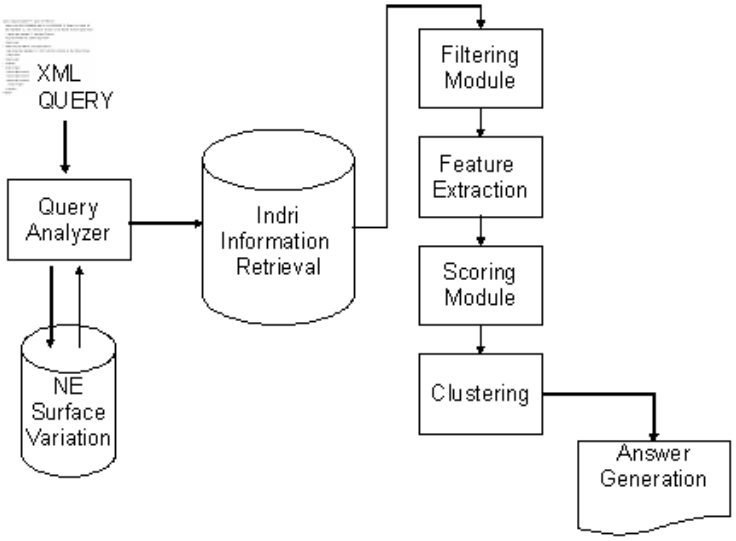


Figure 8.2: Query Focused ConciseSpeech

mainNE and subMainNE. MainNE is found by stripping of all prefixes and suffixes of the person argument. The extracted prefixes which are commonly the titles such as "president," "senator" are listed as subMainNE. We explicitly defined subMainNE because in our data analysis we found many instances of modifiers such as "president said, senator told" preceding statements.

Another important function of the query analyzer is to find the surface variations of the NEs in the query. For example, "Osama bin Laden" may be mentioned in the news as "Osama", "Bin Laden," and other surface variations. Knowing these various surface forms is important for information retrieval purposes as it has been shown in information retrieval literature that expanding queries with such additional surface forms usually improves the search results [Xu and Croft, 1996]. We have built a tool that provides surface variations of any given NE using the NE graph we describe in

said— has said— has told— told— announced— stated +
reported — clarify— quote— cite— reported +
asserted— replied — insisted— denied— speaking +
called— reiterated— spokesman— spokeswoman +
— spokesperson— accused— declared— charged +
address— message— call — appealed— announce +
reports— quoted— clarified— asserts— assert— insist +
— appeal— deny— accuse— addressed— criticized +
pointed out— statement issued by

Table 8.1: Common Say-Verbs

Section 8.2.3.

There are some common patterns in the style of introducing statements by news anchors or reporters. Some of these patterns are "Person said," "... was announced by Person," "He told others that ..." We gathered these patterns manually and came up with the verbs associated with these patterns such as "said, told, announced, spoke." These are called "verbs of saying." The verbs of saying that we extracted from our data analysis are shown in Table 8.1. We pass these verbs to our IR engine to expand query to improve document retrieval accuracy.

8.1.2 Information Retrieval

After the query is analyzed we pass it to the IR engine to retrieve information from the GALE corpus. The corpus consists of English, Arabic and Chinese Newswire,

Blogs, Chats and BN. We next explain the use of Indri for document retrieval.

8.1.2.1 Document Retrieval Using Indri

Indri is an information retrieval engine and its query language can handle ordered and unordered phrases. Indri supports a structured query language which can be represented in a xml format, allowing Indri to handle very complex GALE queries with phrase matching, weighted expression, boolean filtering. An Indri query is defined as shown in Equation 8.1.

$$query := (beliefOp)+ \tag{8.1}$$

where $beliefOp$ adds various belief weights by operators such as *#weight, #combine, #not* and term weights by operators such *#syn, #datebefore* allowing us to easily incorporate source and date restrictions as well. We use *#combine* operators to expand the query with the verbs of saying. We pass the expanded query to the Indri retrieval engine. Indri returns a set of relevant documents ranked according to the relevance with the question.

8.1.3 Filtering Module

After we receive the relevant documents, we filter out the sentences that are unlikely to be a part of the answer due to disfluencies, word errors or non-understandable tokens assigned by the ASR engine. The ASR engine we are using [Zheng et al., 2007] assigns @repeat@ tokens to segments of the speech signal whenever the signal is too noisy to decode. We cannot assign significance scores to a sequence of @repeat@

tokens so we remove sentences with too many @repeat@ tokens. The filtering module has a set of hand-crafted rules to remove such sentences.

8.1.4 Feature Extraction and Scoring

Our scoring mechanism for query-focused summarizer is significantly different than the scoring module of ConciseSpeech. In ConciseSpeech, we had no notion of user's queries so we only needed to compute significance relative to other sentences in the story. In query-focused summarizer, the arguments of the query defines the significance of sentences in the story. We extracted a set of lexical features based on the query arguments. The first set of features were based on the first argument of the query. Based on mainNE and subMainNE described in 8.1.1 we extract features such as the count of MainNEs and subMainNEs. We then extracted a feature that computes a distance between topic argument and the sentence using a word overlap measure. We further extracted features that are defined by a set of hand-crafted rules. Some examples of the rules for feature extraction are - Does the sentence contain mainNE and topic text? Does the sentence contain a verb of saying and subMainNE within a window of 5 words? We developed these rules manually by carefully looking at our annotated data.

After the feature extraction process was complete, we scored each sentence for its relevance to the query. Unlike the scoring module of ConciseSpeech, we could not build a statistical model for scoring for our query-focused summarizer due to lack of enough training data. Hence, we created a set of hand-prepared rules for scoring relevance of each sentence. We show some examples of scoring rules here.

1. If hasSubTopicText == true && hasSubMainNE == true && hasSaidVerb == true then SaidTopicNEScore = 1

2. If NE_SayVerb_TopicText_Pattern == true || SayVerb_TopicText_NE_Pattern == true then statementPatternScore = 1

3. If hasMainNE == true then mainNEScore = 1;

We linearly combined all of the scores with various weights to obtain a total score for each sentence. We compared the total score of the sentence with a threshold which was defined empirically. If the total score was higher than the threshold we appended the sentence to our answer. We performed many iterations of experiments where we compared the generated answer with the gold standard by varying the value of the threshold until we found the optimal threshold.

Sentences whose score surpasses the threshold are then clustered to identify redundancy. We use the clustering algorithm developed for GALE to eliminate redundancy.[1] The resulting set of sentences is the final answer to the question.

8.1.5 Evaluation

We evaluated our system on Template 5 GALE queries. Given a GALE query, the system fetched documents using the Indri search engine. The retrieved documents were automatically annotated for various entities and relations which were then passed on to our system. The system extracted the most significant sentences relevant to the query. We had the gold standard answers provided by LDC for each of these

[1]We are just end-users of the clustering module and treat it as black box.

Length	Precision	Recall	F-measure
3200	0.14	0.402	0.265
4800	0.112	0.420	0.258
6400	0.093	0.478	0.260
7200	0.082	0.478	0.251
Average	0.107	0.445	0.258

Table 8.2: Nuggeteer Results for Template 5 queries

queries. We evaluated the system by computing precision, recall and f-measure using the Nuggeteer evaluation tool [Marton and Radul, 2006]. The results are shown in Table 8.2. We obtain an f-measure of 0.258, precision of 0.107 and recall of 0.445. The reported numbers are the average for 10 queries of Template 5 with a thresholding for answer lengths of 3200, 4800, 6400 and 7200 characters.

We note that the precision is quite low for the system while recall is significantly better. One of the main reasons for this behavior is that template 5 question is about finding statements made by a person on a given topic; and in most cases there are only a few instances of statements made by a person on a particular topic. When there are only few instance of statements in the corpus and we retrieve many documents our precision drops significantly because most of the documents do not contain statements by the person. But we cannot retrieve very few documents for all queries because there are some queries which have many statements available in the corpus. Hence, in order to improve precision, we need a better way to find relevant documents. We address this issue in Section 8.2. Also, in the above evaluation, we are searching a

mixed corpus which also contains translated speech and text. The foreign spoken documents, when transcribed and translated, have many word errors especially for NEs. The NE detector frequently fails to detect NEs on these texts. For our question not being able to find the name of the person who made the statement significantly degrades the overall performance.

8.2 Improving Summarization with Query Expansion Using NE-NET

If we provide more documents relevant to the query it is likely that the query-focused summarizer will be able to find more relevant information to produce a better summary. We wanted to improve the document retrieval module to find more relevant documents. We hypothesize that finding related NEs for the name in the query argument and using these to enrich the query can improve the document retrieval module for query-focused speech summarization we described in last section. We describe a method to find the related NEs and experiments showing an improvement in document retrieval accuracy for GALE templates.

8.2.1 Related Named Entities

In the IR literature we see that knowing if the query terms are NEs can be effectively used for improving IR. If we know the query term is NE then a higher weights can be assigned for matching NE terms than matching other content words. If we search our corpus for "Orhan Pamuk" (a nobel laureate) then a document that contains men-

Orhan Pamuk

From Wikipedia, the free encyclopedia

Ferit Orhan Pamuk (born on June 7, 1952 in Istanbul) is a Nobel Prize-winning Turkish novelist. [1]

Pamuk is often regarded as a post-modern writer. His long-standing popularity in his home country was affected when he made a statement regarding the mass killings of Armenians and Kurds in Anatolia (see "Criminal case" below),[2] but his readership around the globe continues to grow. As one of Turkey's most prominent novelists,[3] his work has been translated into more than forty languages. He is the recipient of numerous national and international literary awards. He was awarded the Nobel Prize in Literature on October 12, 2006,[4] becoming the first Turkish person to receive a Nobel Prize.

Contents [hide]

1 Biography
2 Work
3 Nobel Prize
4 Criminal case
4.1 Pamuk's statements
4.2 Prosecution
4.3 International reaction
4.3.1 Some conflicting opinions from Western reviewers
4.4 Charges dropped
4.5 Aftermath

Ferit Orhan Pamuk

Born: June 7, 1952 (age 54)
Istanbul, Turkey
Occupation(s): Novelist
Nationality: Turkish
Writing period: 1974-present
Literary movement: post-modern literature

Figure 8.3: The Wikipedia page of the Turkish novelist *[Orhan Pamuk]*.

tions of "Orhan Pamuk" will have higher likelihood of being relevant than documents with no mention of Orhan Pamuk. For example, Indri estimates the ranks of the documents to be retrieved by computing the expectation over the posterior probability of $P(M|D)$ where M is the model and D is the set of documents as shown in Equation 8.2

$$P(r|D) = \int P(r|M)P(M|D)dM \tag{8.2}$$

Documents that contain "Orhan Pamuk" will have higher $P(r|D)$. Indri can incorporate new information using its #combine operator that can affect the computation of $P(r|D)$. In such an IR engine, if we provide information on related NEs to

"Orhan Pamuk" such as "The Black Book" (a book Orhan Pamuk wrote), besides the query term, $P(r|D)$ will may improve. In our experiments we show that providing related NEs with the query improves the relevance ranking of the document. But the important question remains, which is how to find these related NEs.

Also from IR literature in relevance feedback we know that using the most relevant documents from the first pass of search to generate new query terms and to re-issue the query can improve the overall accuracy of a document retrieval system. It is possible that finding relevant NEs related to the query may also improve the document retrieval performance if we use it similar fashion of re-issuing the query with more relevant terms.

Even though there are many NE detection systems [Baluja et al., 2000, Mikheev et al., 1999] with high accuracy, we do not have many systems that relate NEs without explicitly defined relationships. Given a NE, finding all related NEs is a difficult problem because NEs can be related to each other in many different ways and in many different contexts. Thus, we have not seen a knowledge source that can provide all the related NEs for a given NE. Such relationships can be very useful for summarization. Currently, we use the counts of NEs and NE types to weight the sentences to be included in a summary in ConciseSpeech. If we know the related NEs we can weight the counts of NEs according to the number of related NEs available for the document. In order to find the related NEs we built a knowledge source which we call Named Entity Network (NE-NET).

Unlike WordNet where human annotators relate words, we built our network automatically. Also, unlike other NE relations mined from the web [Matsuo et al., 1999]

our relations have a higher precision with very little noise. To accomplish this, we make use of Wikipedia, a structured knowledge source created by contributors and constantly edited for accuracy.

8.2.2 Related Work

A significant amount of work has been done in building networks of "person names" in the social network research and in social theory research. [Matsuo et al., 1999] presents a system, Polyphonet, that uses search engines to measure the co-occurrence of names assuming that co-occurrence with some frequency represents some kind of relationship between the co-occurring person names. [Culotta et al., 2004] identifies unique individuals in email messages, finds their homepage and fills out their personal information; links are placed between the owner of the web page and persons discovered on the web pages. Even though these techniques work reasonably well, the resulting network does not have an accuracy of the same degree as that of WordNet, such that it can be used as a knowledge source. Also, we are not interested in a network of just any person names; we are interested in Named Entities that may be of some importance to many NLP tasks, and usually these NEs have some probability of occurring in the news or on the web.

Another relevant work is WordNet [Miller et al., 1990]. WordNet provides relationships between words such as synonyms, hyponyms, and hypernyms while FrameNet provides relations between semantic concepts. Our work is different from these because we do automatic acquisition of knowledge from Wikipedia. There has been recent work based on WordNet and Wikipedia [Suchanek et al., 2007] which builds a

semantic knowledge with ontology based on Wikipedia categories. [Suchanek et al., 2007] produces an ontology that defines relationships such as "Elvis" $born-in$ "1943" by identifying all the persons born in category '1943' listed in Wikipedia. In our NE-NET we have not limited our network by trying to define relationships to build an ontology. We let any two aspects be related if they are annotated as such by Wikipedia annotators.

8.2.3 Building NE-NET

Wikipedia contributors continuously annotate Wikipedia pages by linking important aspects on each page to other Wikipedia pages, which may or may not yet exist. (Unfortunately, this procedure is not as clean as it seems; contributors may refer to the same page in different ways, and one page may have several titles.) We downloaded all Wikipedia articles and the links between the aspects of the document. (Aspects are words that have been linked to other documents.) We traversed through each document and each aspect in the document. We built a new node for each new aspect and added a link to the existing graph. We iterated this method until we processed all the documents in Wikipedia which resulted in a huge graph with nodes as aspects of documents and edges defining relations between them. In total there are about 7 million nodes. Of those, 1.5 million have an actual Wikipedia page, and 35 million edges.

Let us consider an example of a Wikipedia page about Orhan Pamuk, shown in Figure 8.3, winner of 2006 Nobel Prize in literature. The page includes links to more than 90 aspects. Many of these aspects are NEs that are highly relevant to Orhan

Pamuk, such as "Thomas Mann", "Marcel Proust", "Leo Tolstoy", and "Fyodor Dostoevsky." Each of the aspects are listed as nodes in NE-NET and are linked to the node of Orhan Pamuk to obtain a graph shown in Figure 8.4.

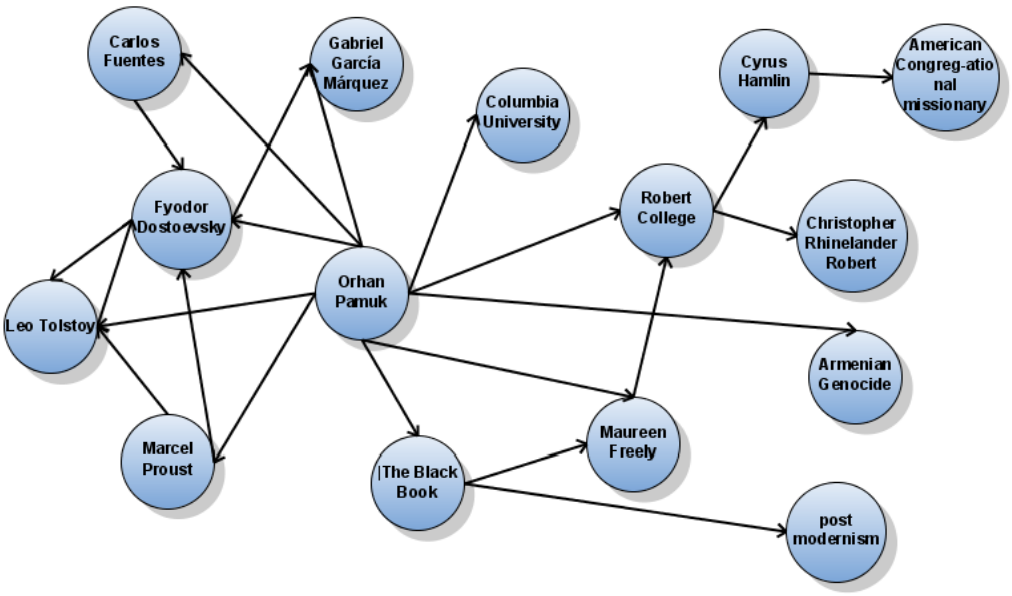


Figure 8.4: A subgraph of Turkish novelist *[Orhan Pamuk]*.

After building the Wikipedia graph we enriched its nodes, as we will describe next, with several useful types of metadata that can be useful for IR.

Category: This field defines the type of an NE and has one of the following values: person, organization, location, misc, and common. If a page is about a NE, the first four values determine what type of NE the page is about. The fifth value indicates that the page is not about a NE [Dakka and Cucerzan, 2008].

Context: This field is a list of all the textual context of an aspect (node in the graph). For example, "Orhan Pamuk" will have 92 context entries as it was mentioned

92 times on other Wikipedia pages.

Surface: Wikipedia contributors refer to an entity by using different terms. For example, for the entity "Orhan Pamuk", contributors use the surface terms "Orhan," "Pamuk," "Orhan Pamuk," "Orhan Parmuk," and others, including ones with spelling mistakes. The value of this field of a particular node contains a list of all the surface terms used in Wikipedia to refer to the entity "Orhan Pamuk."

Abstract: Wikipedia contributors create a few sentences to describe each Wikipedia page. Unfortunately, not all pages have such descriptions. The abstract field in our system contains this human description. Figure 8.5 shows the abstract of the entity "Berlin". The abstract for "Orhan Pamuk" is empty.

First Paragraph: To overcome the problem of missing abstracts, we attach each node to its first paragraph.

<abstract>
Berlin is the capital city and one of the sixteen states of the Federal Republic of Germany. It is the heart of the Berlin-Brandenburg metropolitan region, located in northeastern Germany </abstract>

Figure 8.5: The abstract of the entity "Berlin," provided by Wikipedia contributors.

8.2.3.1 Enriching Nodes with TF·INF

After the building process described above has been completed, every node in the network includes the properties we have described above. Thus, we can easily access NE-NET to find all the related NEs for a given node. We can also use this information

to compute a new measure that we propose — Term Frequency * Inverse Network Frequency (TF·INF). The measure computes the importance of NE given its frequency and the number of relations with other NEs in the network. The measure is motivated by TF·IDF. TF·IDF finds the representative words of the document by computing Term Frequency (TF) and multiplying with Inverse Document Frequency (IDF) which scales the TF by the number of times the word occurred in the other documents. If "Orhan Pamuk" occurs only a few times in a collection of documents and TF is high in the current document, the overall score of TF·IDF will be high. The final score is dependent on the collection of documents used for computing IDF. Note that IDF is determined simply by computing the number of documents in which the term occurred, without taking into account if the word occurred in the document is relevant for the task. On the other hand Inverse Network Frequency INF in TF·INF takes the relevance of the word when computing the counts. We are able to take account of such relevance in computing INF because each node in our graph is a term in Wikipedia. Whenever there is a relevant connection between two terms we have an arc joining them. This means instead of counting all the documents where the term occurs we can count the nodes the term is linked to get INF values. Hence we define our measure with the following equations for a term T_i within a particular document D_j

$$TFoINF_{i,j} = TF_{ij}.INF_i \tag{8.3}$$

where

$$TF_{ij} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{8.4}$$

where the numerator $n_{i,j}$ is the frequency of term in document D_j. The term frequency is normalized by the denominator, the total number of all terms in document D_j. The INF for the term T_i is provided in the following equation.

$$INF_i = log(\frac{|S|}{\sum e(i,k)}) \tag{8.5}$$

where $|S|$ is the size of the graph (the total number of nodes in the graph) and $\sum e(i,k)$ is the sum of edges between the node i and node k where $i \neq k$ and $distance(i,k) = 1$.

TF·INF provides information on the relative importance of a term T_i in the document taking account of its TF and the number of times it is linked by other NE in the NE graph.

8.2.4 Evaluation: Improvement in Document Retrieval Using NE-NET

Query focused summarization involves retrieving documents that are relevant to user's query from a large corpus, then summarizing the retrieved documents. If we provide more relevant documents to a summarization engine it is likely that the summaries can be more relevant. Hence, one of the tasks in building a better query-focused summarizer is to improve the document retrieval component. In this section we show how we can use NE-NET to improve document retrieval.

	Node Properties
1	Total Incoming Arcs
2	Total Outgoing Arcs
3	Category
4	Context
5	Surface
6	Abstract
7	First Para
8	Inv Network Frequency

Table 8.3: Basic Node Level Service in NE-NET

For our experiments we needed a large corpus that can be indexed and a set of queries for testing document retrieval. We chose to index the TDT4 corpus. TDT4 has $58,850$ English documents which include the text of newswire and BN reports from various channels. We indexed all of the TDT4 documents using the UMass Indri index building tool [Strohman et al., 2005]. We chose GALE queries to search the TDT index because we had gold standards provided by LDC.

We performed two sets of experiments to see if we can improve document retrieval using NE-NET. The first experiment involved searching the TDT4 corpus with regular queries. Regular queries are GALE queries formatted in Indri query style. The second experiment involved searching the TDT4 corpus with expanded queries, where expanded terms were retrieved using NE-NET. For example, for the GALE query "Provide information Ehud Barak" we obtain a regular Indri query that only contains

the query-term "Ehud Barak" and the NE-NET query that contains the following relevant NEs as relevant terms:

- *Ehud Barak, taba summit, dan shomron, bandar bin sultan, yehoshua saguy, limor livnat, amnon lipkin-shahak, ahron bregman, yossi sarid, binyamin ben-eliezer, krav maga, sayeret matkal, tal law, education minister of israel, 1973 israeli raid on lebanon, moshe arens, blue line (lebanon), list of defense ministers of israel, electronic data systems, amir peretz, camp david 2000 summit, meretz-yachad, al-aqsa intifada*

We note the high accuracy of our NE-NET in finding related NEs in the example shown above for "Ehud Barak." In order to select only the most relevant NEs to the query we computed INF normalized by the ratio of outDegree and inDegree for the NE. We added these terms in the query using "#combine" operator of Indri.

For our experiments we first took 19 GALE queries which had relevant answers in TDT4 corpus and also had human relevance annotation. The human annotation for relevance of documents was provided by LDC and consisted of a list of document names that were relevant for a given query. These queries were not limited to any particular template type. For each query we used NE-NET to retrieve the related names of the query term and expanded the query as described above. We then searched our corpus twice, first using the regular query and next using NE-NET query. We compared the returned documents with the relevance judgments made by human annotators. The annotation provided a list of relevant documents for each query. We compared the documents retrieved with the gold standard using precision,

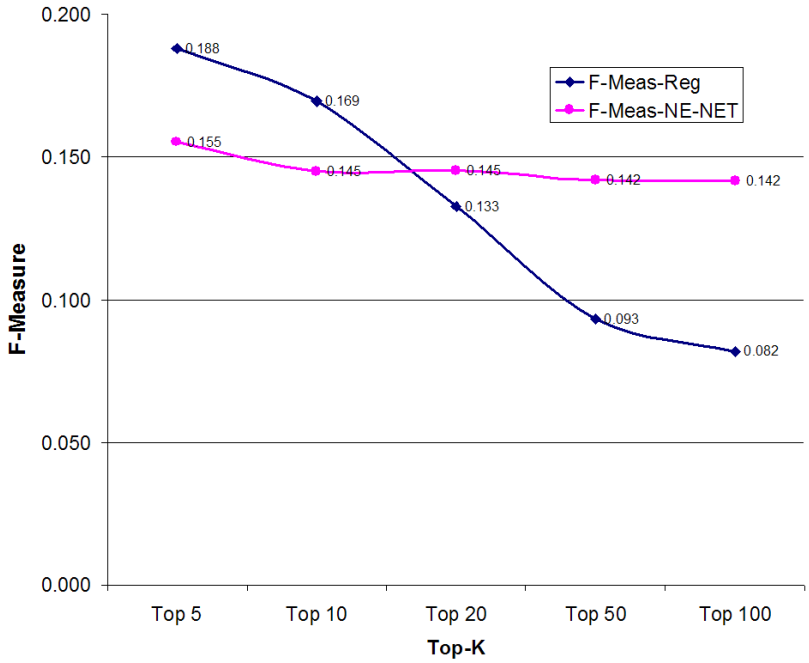


Figure 8.6: NE-NET vs Regular Queries: A Mixture of Query Types Showing an Improvement of NE-NET Queries over Regular Queries for various K values

recall and f-measure. We tested each query by choosing the Top K documents of the returned results with $K = 5, 10, 20, 50, 100$.

The average precision, recall and f-measure for various K values are shown in Figure 8.6. We note that regular queries quickly degrade as we increase the value of K. but the f-measure for NE-NET queries do not degrade at all – even when we choose more top ranked documents. In fact, for top 50 and 100 documents, f-measure for NE-NET queries is higher than for regular queries by 5.97% and 4.86% respectively.

We see in Table 8.6 that most of the gain in f-measure is due to better precision. Retrieving more documents with only the query terms quickly degrades for regular queries. For NE-NET queries, our expansion technique provided enough rel-

K	Type	Prec	Recall	F-Measure
Top 5	REG	0.316	0.211	0.188
	NE-NET	0.242	0.188	0.155
Top 10	REG	0.232	0.238	0.169
	NE-NET	0.205	0.206	0.145
Top 20	REG	0.145	0.273	0.133
	NE-NET	0.192	0.217	0.145
Top 50	REG	0.079	0.289	0.093
	NE-NET	0.188	0.217	0.142
Top 100	REG	0.064	0.309	0.082
	NE-NET	0.188	0.217	0.142

Table 8.4: Precision Recall and F-Measure for NE-NET and Regular Queries

evant terms to retrieve relevant documents even for higher K values. We see that the precision of NE-NET queries is 12.30% and 10.88% higher for the top 100 and 50 documents in comparison with regular queries. For the same queries we see a degradation of recall. However, the ratio of increase in precision and decrease in recall is high enough that we see significant improvement in f-measure. We should also note that when we choose very few documents, such as the top 5 documents, regular queries perform better than NE-NET queries. We expect this behavior because the documents that are ranked in the top 5 or 10 are likely to be very relevant as there will be documents with an exact match to the query terms. Usually the problematic documents are the documents that are ranked lower than the top 10, and these were

the ones NE-NET was able to handle better. We should also note that the query set we chose has a mix of different GALE query types. Hence the use of NE-NET is flexible enough for us to use in all query types for GALE.

In this chapter we extended ConciseSpeech to handle user's queries. We tested our system within the GALE distillation task and found that our system produced reasonable user-focused speech summaries. We further showed that the information retrieval module of user-focused summarization can be improved by expanding queries with related NEs that were automatically obtained from our knowledge source, NE-NET.

Chapter 9

Conclusion, Limitations and Future Work

In this thesis, we presented a technique to summarize BN spoken documents. Our approach is based on extracting significant segments of speech and then combining them. Our results show that our method for summarization works significantly better than a baseline summarizer that extracts the lead N sentences of a story. In order to compute the significance of sentences for summaries, we showed that there are challenges we have to address that are specific to the speech domain, such as word errors, disfluency and segmentation. We presented techniques to address these problems. We then presented IE methods to extract relations particular to BN such as speaker roles, soundbites and commercials. We then showed how to combine these sources of information in one coherent statistical framework to predict the significance of segments for inclusion in a summary. The segments predicted to be significant were combined by our algorithm that optimized significance, redundancy and length to

produce a summary of a desired summarization ratio. We further extended the summarizer to handle user queries and produce a query-focused summary. To handle user queries better we created a knowledge resource, NE-NET, and showed that the use of NE-NET significantly improves document retrieval for query-focused summarization.

There are a few limitations to the work we have presented in this thesis. One of the biggest limitations of our summarizer is that it does not perform any kind of syntactic analysis on ASR transcripts to rewrite long sentences; this would be especially important since BN spoken sentences tend to widely vary in length. We should note though that text generation is a difficult task, and generating sentences from ASR transcripts is probably even harder as sentences in ASR transcripts do not adhere to grammatical rules and contain many word errors.

Another limitation of our summarizer is that we perform most of the experiments at the sentence level. We noted that using intonational phrases could potentially produce a better summarizer (at least using the ROUGE metric) but we had our final summarizer extract sentences because we did not know the effect on comprehension when we concatenate intonational phrases. If we can find a better way to combine IPs such that it does not degrade the readability of the summaries, we should explore the possibility of building a summarizer based on phrases or words.

NE-NET is a very useful resource for retrieving related entities but it cannot be used to identify the relationship between two entities. The lack of such information can limit the usability of NE-NET, especially for tasks where we need to identify all the related entities pertaining to the type of the current relation.

Although our summarizer has limitations, we believe we have made contributions

in increasing knowledge of how to summarize BN. The main research contributions of the thesis are as follows:

- A speech summarizer, ConciseSpeech, that can summarize BN documents by optimizing significance, redundancy and length
- Experiments showing that it is possible to summarize speech just with prosodic information [Maskey and Hirschberg, 2006b]
- Definition and use of novel structural information for speech summarization [Maskey and Hirschberg, 2003]
- Use of word confidence scores to improve clustering of spoken documents with text for summarization
- A phrase-based machine translation approach for disfluency removal [Maskey et al., 2006]
- New IE techniques for extracting information pertaining to speaker roles, soundbites and commercials [Maskey and Hirschberg, 2006a]
- Experiments showing the possible uses of intonational phrases for summaries
- A knowledge resource, NE-NET, that can relate NEs and can be used to improve document retrieval

We should note that the algorithms we have presented here were testing in BN domain. They may not work equally well for other genres such as talk shows, live broadcasts, documentary and television series. We trained our system on BN shows,

so all of our training parameters in our statistical model is tuned to BN speech. For example, our model for commercial detection can potentially be used in shows in other genres such as sports and entertainment but will have to be tuned accordingly. On the other hand, module such as headline detection cannot be used for documentary shows at all. If we want to increase the domain coverage of our summarizer we have to tune our summarizer by removing some modules such as headline detection while we may have to add new modules to extract relations particular to the domain.

Even though we need to change the overall structure of our summarization system to use the summarizer for other domains, there are some key modules and algorithms that are general enough to be used for other domains. We proposed an algorithm in Section 7.2 that optimizes significance, redundancy and length which we believe is general enough to be used for other domains. The algorithm extracts segments and adds the most significant segment that is not redundant in each iteration. This iterative method of building a summarizer can be used for many other domains such as sports or entertainment news where there are a lot of redundancy in the reporting method of the program. Also, many television and radio broadcasts structure is planned carefully before the show starts so extracting structural features in similar ways to ConciseSpeech can be useful for summarizing such shows.

For our future work we would like to address the limitations we have previously mentioned. We want to perform syntactic processing such that we can identify significance on smaller segments of sentences, such as phrases. Also, such processing would help in rewriting sentences. We want to improve NE-NET by adding the type of relationship on arcs of the network such that NE-NET can be queried not only for

related entities but for entities that fall under given relationship types. We want to build upon our IE engines to extract more relations from speech that can be useful for summarization.

Besides addressing these limitations, we would also like to explore the statistical learning algorithms that can model the hierarchical nature of information in BN. BN contains a strict hierarchy over the type of segment we are examining. BN shows contain stories which contain turns, turns contain sentences, and sentences subsume phrases and words. Each of these segments can be used to extract information particular to their segment type. For example, we can extract POS or word senses for words while we can obtain parse trees for sentences and extract speaker roles for turns. In most of the summarization algorithms, we see that the hierarchical nature of this information is not maintained. For example, turn and story information is flattened to sentence level. We would like to address how to maintain the hierarchical nature of information in BN documents to improve speech summarization as a part of our future work.

Appendix A

Data Processing

A.1 Example of RTTMX file

```
...
on NA SENT_146 NA 10/01/2000#14:15:46.84 TURN_10
cnn NA SENT_146 NA 10/01/2000#14:15:46.84 TURN_10
kitty NA SENT_147 NA 10/01/2000#14:15:46.84 TURN_10
goes INSUMMARY_38 SENT_148 NA 10/01/2000#14:16:00.11 TURN_10
...
website NA SENT_150 NA 10/01/2000#14:16:00.11 TURN_10
napster NA SENT_151 NA 10/01/2000#14:16:00.11 TURN_10
...
```

Rttmx file contains tuples of data where columns are attributes for every word in the document. Every tuple represents a word with the first column stating the word value.

A.2 Web-Interface for Summarization Annotation

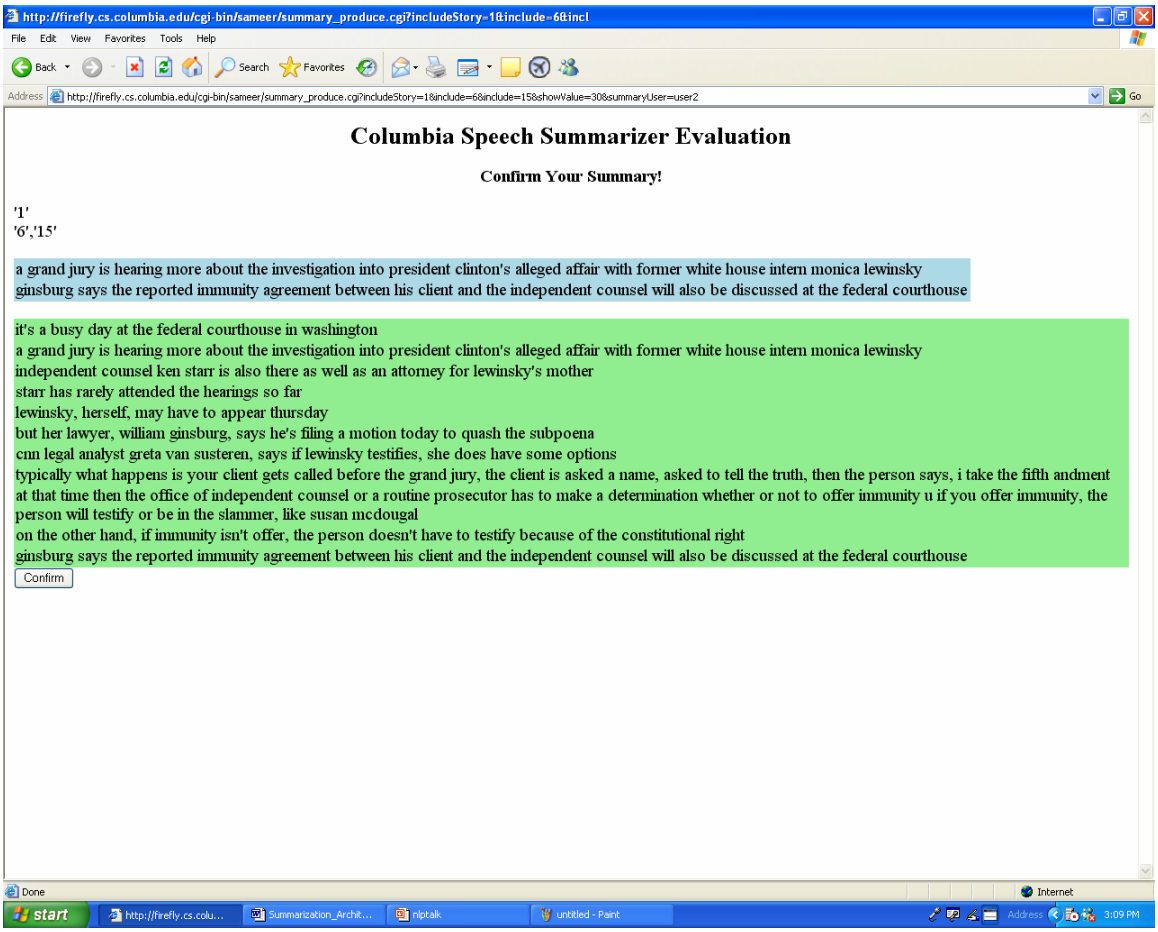


Figure A.1: Confirming Segment Selection for Summarization Annotation

The annotators were asked to reconfirm the selected sentences for the summary by making them able to read the summary and story in a same page. They could go back and change the summary if they wanted to.

Appendix B

User Queries for Summarization

B.1 Instruction for GALE Templates 4 and 5 in Year 2

B.1.1 Template 4: Provide information on [organization ‖ person]

Examples for Template 4

- PROVIDE INFORMATION ON [May 1 terrorist organization]

  Relevant:

  Objective: May 1 is an extremist organization

  Time: May 1 was active between 1987 and 1995

  History/Activity: May 1 was involved in numerous bomb attacks

Members: May 1 announced its merger with ELA

Activity: May 1 was involved in the murder of a policeman

- PROVIDE INFORMATION ON [Paul Bremer] Activity Date: 2003-01-01 to 2003-12-31

  Relevant:

  Role: Iraq's U.S. administrator, Bremer

  Statement: Iraq's U.S. administrator, [[Bremer, said]] Thursday [2003-07-31] [[that general elections could be held in Iraq within a year.]]

  Meeting: [[British Foreign Secretary Jack Straw went into a meeting]] on Monday [2003-08-21] [[with Paul Bremer.]]

  Meeting: "The meeting reached some positive results on the plan to create constitutional and political councils," Barzani told reporters after meeting Bremer in Salaheddin, close to Arbil.

B.1.2 Template 5: Find statements made by or attributed to [person] ON [topic(s)]

Examples for Template 5

- FIND STATEMENTS MADE BY OR ATTRIBUTED TO [Iraqi Deputy Prime Minister Tariq Aziz] ON [Iraqi President Saddam]

  Irrelevant:

Tariq Aziz should be able to provide "important information" about Saddam's regime (not actually saying that Aziz did or would say something-just hoping for something)

Tariq Aziz is talking, we don't know how much, we don't know how accurate (says there is the existence of a statement, but doesn't provide any actual or hypothesized content of a statement)

- FIND STATEMENTS MADE BY OR ATTRIBUTED TO [Gao Xingjian] ON [politics]

  Relevant:

  No comment: Gao Xingjian has chosen not to talk about politics

  No comment: Gao Xingjian avoided the questions, when asked repeatedly whether he had felt any political pressure

  No comment: Gao Xingjian had no comment on whether he had been under any political pressure

  Irrelevant:

  Gao Xingjian was convinced that the situation in China would deteriorate (information was likely to be received by a statement, but there's no actual evidence to confirm a statement)

Appendix C

Labeling Instructions and Examples of Stories and Automatic Summaries

C.1 Labeling Instructions

We asked the annotators to first generate manual abstractive summaries.[1]

- Read the document from the start to the end, i.e. text inside start and end DOC tags.
- Write a summary in your own words but use the words from the document whenever possible.

[1] All the relevant labeling manuals with instructions for generating summaries and annotating relations are available at www.cs.columbia.edu/~smaskey/thesis.

- The summary should not have any redundant information.
- Summary should not be longer than 30% of the length of the document.

After they generated manual summaries we asked them to mark sentences in their story to support the summary they generated.

- Read the story and your summary
- Find the sentences in the summary that support the summary you generated
- Extract (select) the sentences you have found that supports your summary
- Use start and end INSUMMARY tags to annotate the sentence. Use dLabel tool as this has the button you need
- Make sure the the SENT tag is outside of INSUMMARY tags.

C.2 Sample Automatic Summaries and Human Summaries

These are examples of stories and their corresponding summaries. Each story is followed by a baseline summary, a summary generated by ConciseSpeech using ASR transcripts, a summary generated by ConciseSpeech using manual transcripts and a summary generated by humans using manual transcripts. Each summary was rated by 5 humans in a scale of 1 to 5 following DUC evaluation instructions. We present

summaries and average human scores for three stories in BN.[2]

STORY

CNN20001211.1400.1485.story.txt

some french butchers are giving away meat in an attempt to regain public confidence in their products

jennifer eccleston explains

a bevy of butchers ferry steaming platters of barbecued beef to crowds in a central paris park. it's sane cow day in paris, according to the 200 butchers who organized the free cookout

they're hoping locals will cast aside mounting fears in france over bse, the deadly brain wasting disease in cattle, and its human equivalent, variant creutzfeldt jakob disease

i said to my husband, david, that it would be good to go the luxembourg gardens and see representatives of the beef producers, and taste good meat

and the meat is really good

we like it very much

when asked if she was afraid of bse, or as it's commonly known, mad cow disease, michele nicolas said it didn't scare her at all music to the ears of today's sponsors

the bse panic broke out in france in october when it was revealed that potentially contaminated meat, from a herd in which one bse case had been identified, had been put on sale in supermarkets

as a result, stores in paris took beef off the shelves, and it was removed from school menus nationwide

today's barbecue hopes to shore up consumer confidence in the troubled beef industry

the aim of this operation is, above all, to make consumers understand that we are small butchers, and we work in our shops with beef raised exclusively for meat, and with that kind of beef, there is no problem

these are beef that have not eaten animal based feed

they eat grains, and not animal feed

---

[2]All the story and summary pairs used in the experiments for this thesis are available at www.cs.columbia.edu/~smaskey/thesis.

so, there is no risk
france's agriculture ministry announced four new cases of bse in cattle last friday, bringing this year's total to over 120 cases, more than four times the figure for 1999
cases have also been reported in britain, ireland and belgium, and, most recently, in germany
european union ministers, reacting to europe wide concerns, last week announced a six month ban on the use of meat and bone meal in all animal fodder
it's considered the main route for bse contamination
jennifer eccleston, cnn,

BASELINE SUMMARY

CNN20001211.1400.0015.summaryB.txt
Average Human Score : 2.40

five
some french butchers are giving away meat in an attempt to regain public confidence in their products
jennifer eccleston explains
yeah
a heavy of what's your is very steaming platters of hard to keep peace to proud in a central paris park it's saying how day in paris forty two hundred @reject@ we work out
they're hoping @reject@ has to side mounted fears his land over the s. e. c.

AUTOMATIC SUMMARY based on ASR transcripts

CNN20001211.1400.1485.summaryA.txt
Average Human Score: 3.20

Some french butchers are giving away meat in an attempt to regain public confidence in their products

A heavy of what's your is very steaming platters of hard to keep peace to proud in a central paris park it's saying how day in paris forty two hundred we work out
They're hoping has to side mounted fears his land over the SEC
When asked if she was afraid of BSE or as it's commonly known as mad cow disease michele said it didn't scare her at all
The BSE panic broke out in france in october when it was revealed that tend to weaken time and it meet won the case had been identified had been put on sale unsupervised
European union ministers reacting to your like stern's last week announced a six month ban on the use of meat and bone meal in all animal fodder

AUTOMATIC SUMMARY based on Manual transcripts

CNN20001211.1400.1485.summaryM.txt
Average Human Score : 4.20

Some french butchers are giving away meat in an attempt to regain public confidence in their products
They're hoping locals will cast aside mounting fears in france over bse the deadly brain wasting disease in cattle and its human equivalent variant creutzfeldt jakob disease
When asked if she was afraid of bse or as it's commonly known mad cow disease michele nicolas said it didn't scare her at all music to the ears of today's sponsors
The bse panic broke out in france in october when it was revealed that potentially contaminated meat from a herd in which one bse case had been identified had been put on sale in supermarkets
France's agriculture ministry announced four new cases of bse in cattle last friday bringing this year's total to over 120 cases more than four times the figure for 1999
Cases have also been reported in britain ireland and belgium and most recently in germany
European union ministers reacting to europe wide concerns last week announced a six month ban on the use of meat and bone meal in all animal fodder

HUMAN SUMMARY based on Manual transcripts

CNN20001211.1400.1485.summaryH.txt
Average Human Score: 3.80

some french butchers are giving away meat in an attempt to regain public confidence in their products.
a bevy of butchers ferry steaming platters of barbecued beef to crowds in a central paris park. it's sane cow day in paris, according to the 200 butchers who organized the free cookout.
they're hoping locals will cast aside mounting fears in france over bse, the deadly brain wasting disease in cattle, and its human equivalent, variant creutzfeldt jakob disease.
france's agriculture ministry announced four new cases of bse in cattle last friday, bringing this year's total to over 120 cases, more than four times the figure for 1999.

STORY

CNN20001211.1400.0433.story.txt

california's strained power grid is getting a boost today which might help increasingly taxed power supplies
a unit at diablo canyon nuclear plant is expected to resume production today
it had been shut down for maintenance
coupled with another unit, it can provide enough power for about 2 million people
meanwhile, a cold snap in the pacific northwest is putting an added strain on power supplies
the area shares power across many states
energy officials are offering tips to conserve electricity, they say, to delay holiday lighting until after at night
set your thermostat at 68 degrees when you're home, 55 degrees when you're away
try to use electrical appliances before p.m. and after p.m. and turn off computers,

copiers and lights when they're not being

BASELINE SUMMARY

CNN20001211.1400.0433.summaryB.txt
Average Human Score : 3.00

a california strained power grid is getting a boost today which might help increasingly taxed power supplies
a unit d. m. blow can you nuclear plant is expected to resume its production today it had been shut down for maintenance coupled with another unit it can provide enough power for about two million people
A california strained power grid is getting a boost today which might help increasingly taxed power supplies
Meanwhile a cold snap in the pacific northwest and putting an added strain on power supplies
Set your thermostat at sixty eight degrees when you're home and fifty five when you're not there trying to use electrical appliances before four PM in after seven PM and turn off computers copiers in lights when they're not being

AUTOMATIC SUMMARY based on ASR transcripts

CNN20001211.1400.0433.summaryA.txt
Average Human Score: 3.20

A california strained power grid is getting a boost today which might help increasingly taxed power supplies
Meanwhile a cold snap in the pacific northwest and putting an added strain on power supplies
Set your thermostat at sixty eight degrees when you're home and fifty five when you're not there trying to use electrical appliances before four PM in after seven PM and turn off computers copiers in lights when they're not being

AUTOMATIC SUMMARY based on Manual transcripts

CNN20001211.1400.0433.summaryM.txt
Average Human Score : 3.60

Meanwhile a cold snap in the pacific northwest is putting an added strain on power supplies
Set your thermostat at 68 degrees when you're home 55 degrees when you're away
Try to use electrical appliances before pm and after pm and turn off computers copiers and lights when they're not being

HUMAN SUMMARY based on Manual transcripts

CNN20001211.1400.0433.summaryH.txt
Average Human Score: 4.00

california's strained power grid is getting a boost today which might help increasingly taxed power supplies.
a unit at diablo canyon nuclear plant is expected to resume production today.
meanwhile, a cold snap in the pacific northwest is putting an added strain on power supplies.

STORY

CNN20001212.1400.0629.story.txt

an international terrorist captured six years ago is still at the center of legal disputes
bob glascoff has his story in today's edition of headliners
in december of 1997, the international terrorist known as carlos the jackal goes on trial in paris for murder
the jackal gained international notoriety during the cold war for staging a string of deadly bombings, assassinations and hostage seizures. he was linked to the assassination of israeli athletes during the 1972 munich olympics, the 1975 seizure of opec

ministers in vienna and the 1976 hijacking of an air france jet to uganda
ilich ramirez sanchez was captured in sudan by french agents in 1994. he was taken to paris to stand trial for the 1975 murder of two investigators and a lebanese national ramirez was convicted and sentenced to life in prison
last year, austria tried unsuccessfully to extradite him to stand trial for his alleged role in the raid on the opec meeting 25 years ago
venezuela, his native country, is currently pushing for ramirez' return, claiming his extradition six years ago was illegal
for headliners, i'm bob

BASELINE SUMMARY

CNN20001212.1400.0629.summaryB.txt
Average Human Score : 1.80

it or national terrorist captured six years ago is still at the center of legal disputes
bob glascoff has the story in today's headliners @reject@ in december of nineteen ninety seven the international terrorist known as power lunch that jekyll goes on trial in paris for murder

AUTOMATIC SUMMARY based on ASR transcripts

CNN20001212.1400.0629.summaryA.txt
Average Human Score: 2.40

Bob glascoff has the story in today's headliners in december of nineteen ninety seven the international terrorist known as power lunch that jekyll goes on trial in paris for murder
Last year austria tried unsuccessfully to extradite him to stand trial for his alleged role in the raid on the hope that maybe twenty five years ago venezuela his native country is currently pushing for a mere as return morning his extradition six years ago was illegal

AUTOMATIC SUMMARY based on Manual transcripts

CNN20001212.1400.0629.summaryM.txt
Average Human Score : 4.00

The jackal gained international notoriety during the cold war for staging a string of deadly bombings assassinations and hostage seizures he was linked to the assassination of israeli athletes during the 1972 munich olympics the 1975 seizure of opec ministers in vienna and the 1976 hijacking of an air france jet to uganda
Ilich ramirez sanchez was captured in sudan by french agents in 1994 he was taken to paris to stand trial for the 1975 murder of two investigators and a lebanese national
Last year austria tried unsuccessfully to extradite him to stand trial for his alleged role in the raid on the opec meeting 25 years ago

HUMAN SUMMARY based on Manual transcripts

CNN20001212.1400.0629.summaryH.txt
Average Human Score: 4.00
bob glascoff has his story in today's edition of headliners.
in december of 1997, the international terrorist known as carlos the jackal goes on trial in paris for murder .
the jackal gained international notoriety during the cold war for staging a string of deadly bombings, assassinations and hostage seizures. he was linked to the assassination of israeli athletes during the 1972 munich olympics, the 1975 seizure of opec ministers in vienna and the 1976 hijacking of an air france jet to uganda.
ilich ramirez sanchez was captured in sudan by french agents in 1994. he was taken to paris to stand trial for the 1975 murder of two investigators and a lebanese national .

Bibliography

[Allan et al., 1998] Allan, J., Carbonell, J., Doddington, G., Yamron, J., and Yang, Y. (1998). Topic detection and tracking pilot study: Final report. In *Proceedings of the Broadcast News Understanding and Transcription Workshop*, pages 194–218.

[Baluja et al., 2000] Baluja, S., Mittal, V. O., and Sukthankar, R. (2000). Applying machine learning for high-performance named-entity extraction. *Computational Intelligence*, 16(4):586–596.

[Banerjee and Rudnicky, 2006] Banerjee, S. and Rudnicky, A. I. (2006). Smartnotes: Implicit labeling of meeting data through user note-taking and browsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Demonstrations*, pages 261–264, New York City, USA. Association for Computational Linguistics.

[Barzilay et al., 2000] Barzilay, R., Collins, M., Hirschberg, J., and Whittaker, S. (2000). The rules behind roles: Identifying speaker role in radio broadcasts. In *Proc. the American Association for Artificial Intelligence conference (AAAI)*.

[Barzilay and Elhadad, 1997] Barzilay, R. and Elhadad, M. (1997). Using lexical chains for text summarization. In *Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS'97)*, Madrid, Spain.

[Boersma, 2001] Boersma, P. (2001). Praat, a system for doing phonetics by computer. *Glot International*.

[Bolinger, 1982] Bolinger, D. (1982). The network tone of voice. *Journal of Broadcasting*, 26:725–728.

[Brown et al., 1993] Brown, P., Della Pietra, S., Della Pietra, V., and Mercer, R. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

[Carbonell and Goldstein, 1998] Carbonell, J. and Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia.

[Christensen et al., 2004] Christensen, H., Kolluru, B., Gotoh, Y., and S., R. (2004). From text summarisation to style-specific summarisation for broadcast news. In *Proceedings of ECIR*.

[Culotta et al., 2004] Culotta, A., Bekkerman, R., and McCallum, A. (2004). Extracting social networks and contact information from email and the web.

[Dakka and Cucerzan, 2008] Dakka, W. and Cucerzan, S. (2008). Augmenting wikipedia with named entity tags. In *The Third International Joint Conference on Natural Language Processing (IJCNLP)*.

[Eisner, 2002] Eisner, J. (2002). Parameter estimation for probabilistic finite-state transducers. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics*.

[Galley and McKeown, 2007] Galley, M. and McKeown, K. (2007). Lexicalized Markov grammars for sentence compression. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 180–187, Rochester, New York. Association for Computational Linguistics.

[Galley et al., 2004] Galley, M., McKeown, K., Hirschberg, J., and Shriberg, E. (2004). Identifying agreement and disagreement in conversational speech: Use of bayesian networks to model pragmatic dependencies. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL–04)*, pages 669–676, Barcelona, Spain.

[Galley et al., 2003] Galley, M., McKeown, K. R., Fosler-Lussier, E., and Jing, H. (2003). Discourse segmentation of multi-party conversation. In Hinrichs, E. and Roth, D., editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL–03)*, pages 562–569, Sapporo, Japan.

[He et al., 1999] He, L., Sanocki, E., Gupta, A., and Grudin, J. (1999). Auto-summarization of audio-video presentations. In *ACM Multimedia (1)*, pages 489–498.

[Heckerman, 1995] Heckerman, D. (1995). A tutorial on learning with bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, Redmond, WA.

[Heeman et al., 1996] Heeman, P., ho Loken-Kim, K., and Allen, J. F. (1996). Combining the detection and correction of speech repairs. In *Proceedings of ICSLP*, pages 363–365.

[Hirschberg, 2002] Hirschberg, J. (2002). Communication and prosody: functional aspects of prosody. *Speech Commun.*, 36(1-2):31–43.

[Hirschberg and Nakatani, 1996] Hirschberg, J. and Nakatani, C. (1996). A prosodic analysis of discourse segments in direction-given monologues. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL–96)*, pages 286–293, Santa Cruz, California.

[Hirschberg and Nakatani, 1998] Hirschberg, J. and Nakatani, C. (1998). Acoustic indicators of topic segmentation. In *Proc. of ICSLP*.

[Honal and Schultz, 2003] Honal, M. and Schultz, T. (2003). Correction of disfluencies in spontaneous speech using a noisy channel approach. In *Eurospeech 2003*.

[Honal and Schultz, 2005] Honal, M. and Schultz, T. (2005). Automatic disfluency removal on recognized spontaneous speech rapid adaptation to speaker dependent disfluencies. In *ICASSP 2005*.

[Hori, 2002] Hori, C. (2002). *A Study on Statistical Methods for Automatic Speech Summarization*. PhD thesis, Tokyo Institute of Technology.

[Hori et al., 2002] Hori, C., Furui, S., Malkin, R., Yu, H., and Waibel, A. (2002). Automatic summarization of english broadcast news speech. In *Proc. of the Human Technology Conference (HLT–02), San Diego*.

[Hori et al., 2003] Hori, T., Hori, C., and Minami, Y. (2003). Speech summarization using weighted finite-state transducers. In *Proceedings of Eurospeech*, pages 2817–2820.

[Hovy and Lin, 1997] Hovy, E. and Lin, C. (1997). Automated text summarization in SUMMARIST. In *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 18–24, Madrid, Spain.

[Inoue et al., 2004] Inoue, A., Mikami, T., and Yamashita, Y. (2004). Improvement of speech summarization using prosodic information. In *Proc. of Speech Prosody*.

[Johnson and Charniak, 2004] Johnson, M. and Charniak, E. (2004). A tag-based noisy channel model of speech repairs. In *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*.

[Johnson et al., 2004] Johnson, M., Charniak, E., and Lease, M. (2004). An improved model for recognizing disfluencies in conversational speech. In *RT04*.

[Kim et al., 2004] Kim, J., Schwarm, S. E., and Ostendorf, M. (2004). Detecting structural metadata with decision trees and transformation-based learning. In *Proceedings of HLT/NAACL*.

[Koumpis and Renals, 2000] Koumpis, K. and Renals, S. (2000). Transcription and summarization of voicemail speech. In *Proc. of the International Conference on Spoken Language Processing (ICSLP), Beijing*.

[Kupiec et al., 1995] Kupiec, J., Pedersoen, J., and Chen, F. (1995). A trainable document summarizer. In *Proceedings of the 18th ACM-SIGIR Conference*, pages 68–73.

[LDC, 2002] LDC (2002). Topic detection and tracking corpus, tdt2.

[LDC, 2006] LDC (2006). Language data consortium.

[Lin, 2004] Lin, C.-Y. (2004). ROUGE: a package for automatic evaluation of summaries. In *Proc. of workshop on text summarization, ACL-04*.

[Liu, 2006] Liu, Y. (2006). Initial study on automatic identification of speaker role in broadcast news speech. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 81–84, New York City, USA. Association for Computational Linguistics.

[Liu et al., 2003] Liu, Y., Shriberg, E., and Stolcke, A. (2003). Automatic disfluency identification in conversational speech using multiple knowledge sources. In *EuroSpeech 2003*.

[Liu et al., 2006] Liu, Y., Shriberg, E., Stolcke, A., Hillard, D., Ostendorf, M., and Harper, M. (2006). Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1526–1540.

[Marcu, 1997a] Marcu, D. (1997a). From discourse structures to text summaries. In *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 82–88, Madrid, Spain.

[Marcu, 1997b] Marcu, D. (1997b). From discourse structures to text summaries. In *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 82–88, Madrid, Spain.

[Marton and Radul, 2006] Marton, G. and Radul, A. (2006). Nuggeteer: Automatic nugget-based evaluation using descriptions and judgements. In *Proceedings of NAACL/HLT*.

[Maskey and Hirschberg, 2003] Maskey, S. and Hirschberg, J. (2003). Automatic summarization of broadcastnews using structural features. In *In Proceedings of Eurospeech 2003*.

[Maskey and Hirschberg, 2005] Maskey, S. and Hirschberg, J. (2005). Comparing lexial, acoustic/prosodic, discourse and structural features for speech summarization. In *Proc. of Eurospeech*.

[Maskey and Hirschberg, 2006a] Maskey, S. R. and Hirschberg, J. (2006a). Soundbite detection in broadcast news domain. In *Proceedings of Interspeech*.

[Maskey and Hirschberg, 2006b] Maskey, S. R. and Hirschberg, J. (2006b). Summarizing speech without text using hidden markov models. In *Proceedings of HLT-NAACL*.

[Maskey et al., 2006] Maskey, S. R., Zhou, B., and Gao, Y. (2006). A phrase-level machine translation approach for disfluency detection using weighted finite state transducers. In *Proceedings of Interspeech*.

[Matsuo et al., 1999] Matsuo, Y., Mori, J., Hamasaki, M., H.Takeda, NIshimura, T., Hasida, K., and Ishizuka, M. (1999). Polyophonet: An advanced social network extraction system. In *Proc of WWW Conference*.

[McCallum, 2002] McCallum, A. (2002). MALLET: A machine learning for language toolkit. http://mallet.cs.umass.edu.

[McKeown et al., 2003] McKeown, K., Barzilay, R., John Chen, D. E., Evans, D., Klavans, J., Nenkova, A., Schiffman, B., and Sigelman, S. (2003). Columbia's newsblaster: New features and future directions (demo). In *Proceedings of NAACL-HLT*.

[McKeown et al., 2001] McKeown, K. R., Barzilay, R., Evans, D. K., Hatzivassiloglou, V., Kan, M.-Y., Schiffman, B., and Teufel, S. (2001). Columbia multi-document summarization: Approach and evaluation. In *Proceedings of the Document Understanding Workshop (DUC 2001)*, New Orleans, USA.

[Mikheev et al., 1999] Mikheev, A., Moens, M., and Grover, C. (1999). Named entity recognition without gazetteers. In *A. Mikheev, M. Moens, and C. Grover. 1999. Named Entity recognition without gazetteers. In Proc. of EACL, Bergen, Norway. EACL.*

[Miller et al., 1999] Miller, D., Schwartz, R., Weischedel, R., and Stone, R. (1999). Name entity extraction from broadcast news. In *Proceedings of DARPA Broadcast News Workshop*.

[Miller et al., 1990] Miller, G., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. (1990). Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography (special issue)*, 3(4):235–245.

[Murray et al., 2005a] Murray, G., Renals, S., and Carletta, J. (2005a). Extractive summarization of meeting recordings. In *Proc. of Eurospeech*.

[Murray et al., 2005b] Murray, G., Renals, S., and Carletta, J. (2005b). Extractive summarization of meeting recordings. In *Proc. Interspeech*.

[Nakatani and Hirschberg, 1994] Nakatani, C. and Hirschberg, J. (1994). A corpus based study of repair cures on spontaneous speech. In *Journal of the Acoustical Society of America*, pages 1603 – 1616.

[Navarro, 2001] Navarro, G. (2001). A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88.

[Prince, 1992] Prince, E. (1992). The zpg letter: subjects, definiteness, and information-status.

[Radev et al., 2004] Radev, D., Allison, T., Blair-Goldensohn, S., Blitzer, J., Celebi, A., Drabek, E., Lam, W., Liu, D., Otterbacher, J., Qi, H., Saggion, H., Teufel, S., Topper, M., Winkel, A., and Zhang, Z. (2004). MEAD - A platform for multi-document multilingual text summarization. In *Proceedings of the 4th International*

Conference on Language Resources and Evaluation (LREC 2004), Lisbon, Portugal, May 2004.

[Radev et al., 2000] Radev, D., Jing, H., and Budzikowska, M. (2000). Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *Proceedings of the ANLP/NAACL 2000 Workshop on Automatic Summarization.*

[Radev and McKeown, 1998] Radev, D. and McKeown, K. R. (1998). Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3).

[Ratnaparkhi, 1996] Ratnaparkhi, A. (1996). A maximum entropy part-of-speech tagger. In *Proc. of EMNLP.*

[Robertston and Jones, 1976] Robertston, S. and Jones, K. S. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, pages 129–146.

[Rosenberg and Hirschberg, 2006] Rosenberg, A. and Hirschberg, J. (2006). Story segmentation of broadcast news in english, mandarin and arabic. In *Proceedings of Interspeech.*

[Rosenberg and Hirschberg, 2007] Rosenberg, A. and Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure.

[Salzberg, 1994] Salzberg, S. L. (1994). Book review: *C4.5: Programs for Machine Learning* by J. ross quinlan. morgan kaufmann publishers, inc., 1993. *Machine Learning*, 16(3):235–240.

[Santen, 1992] Santen, J. P. H. (1992). Contextual effects on vowel duration. *Speech Commun.*, 11(6):513–546.

[Schiffman et al., 2002] Schiffman, B., Nenkova, A., and McKeown, K. (2002). Experiments in multi-document summarization. In *In HLT 2002.*

[Shriberg et al., 1997] Shriberg, E., Stolcke, A., and Bates, R. (1997). A prosody only decision tree model for disfluency detection. In *Proceedings of Eurospeech.*

[Shriberg et al., 2000] Shriberg, E., Stolcke, A., Tur, D., and Tur, G. (2000). Prosody-based automatic segmentation of speech into sentences and topics. In *Speech Communication, Vol. 32*, pages 127–154.

[Snover et al., 2004a] Snover, M., Dorr, B., and Schwartz, R. (2004a). A lexically-driven algorithm for disfluency detection. In *Short Paper Proceedings of North American Association for Computational Linguistics (NAACL) and Human Language Technology (HLT) Conference.*

[Snover et al., 2004b] Snover, M., Dorr, B., and Schwartz, R. (2004b). A lexically driven algorithm for disfluency detection. In *Short Paper Proceedings of HLT-NAACL 2004.*

[Spiker et al., 2000] Spiker, J., Klarner, M., and Gorz, G. (2000). Processing self corrections in a speech to speech system. In *Proceedings of 18th Conference in Computational Linguistics*.

[Stolcke and Shriberg, 1996] Stolcke, A. and Shriberg, E. (1996). Statistical language modeling for speech disfluencies. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing, vol. 1*, pages 405–409.

[Stolcke et al., 1998] Stolcke, A., Shriberg, E., Bates, R., Ostendorf, M., Hakkani, D., Plauche, M., Tur, G., and Lu, Y. (1998). Automatic detection of sentence boundaries and disfluencies based on recognized words. In *Proceedings of ICSLP*, pages 2247–2250.

[Strohman et al., 2005] Strohman, T., Metzler, D., Turtle, H., and Croft, W. B. (2005). Indri: A language-model based search engine for complex queries extended version.

[Suchanek et al., 2007] Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: A Core of Semantic Knowledge. In *16th international World Wide Web conference (WWW 2007)*, New York, NY, USA. ACM Press.

[Tweed et al., 2005] Tweed, D., Fisher, R., Bins, J., and List, T. (2005). Efficient hidden semi-markov model inference for structured video sequences. In *ICCCN '05: Proceedings of the 14th International Conference on Computer Communications and Networks*, pages 247–254, Washington, DC, USA. IEEE Computer Society.

[Witbrock and Mittal, 1999] Witbrock, M. and Mittal, V. (1999). Ultra-summarization: A statistical approach to generating highly condensed non-extractive summaries. In *Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR–99), Poster Session*, pages 315–316, Berkeley, CA.

[Witten and Frank, 2000] Witten, I. H. and Frank, E. (2000). *Data Mining: practical machine learning tools and techniques with java implementations*. Morgan Kaufmann.

[Xu and Croft, 1996] Xu, J. and Croft, W. B. (1996). Query expansion using local and global document analysis. In *SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 4–11, New York, NY, USA. ACM.

[Zechner and Waibel, 2000] Zechner, K. and Waibel, A. (2000). Diasumm: Flexible summarization of spontaneous dialogues in unrestricted domains. In *Proceedings of COLING-2000, Saarbruecken, Germany*, pages 968–974.

[Zheng et al., 2007] Zheng, J., Cetin, O., Hwang, M.-Y., Lei, X., Stolcke, A., and Morgan, N. (2007). Combining discriminative feature, transform, and model training for large vocabulary speech recognition. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on Speech and Signal Processing*.

[Zhou et al., 2005] Zhou, B., Chan, S., and Gao, Y. (2005). Constrained phrases-based translation using weighted finite state transducer. In *ICASSP 2005*.

[Zhou and Gao, 2006] Zhou, B. and Gao, Y. (2006). Fast phrase-based statistical translation using fst's. In *ACL*.

[Zhu and Penn, 2006] Zhu, X. and Penn, G. (2006). Summarization of spontaneous conversations. In *Proceedings of the 9th International Conference on Spoken Language Processing*.