Synthetic News Radio: Content Filtering and Delivery for Broadcast Audio News

by

Keith Jeffrey Emnett

B.S. Electrical Engineering
Honors Program Degree
Oklahoma State University, 1995

Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences

at the

Massachusetts Institute of Technology

June 1999

Signature of Author: ____

Program in Media Arts and Sciences
May 12, 1999

Certified by: ____

Christopher M. Schmandt
Principal Research Scientist, MIT Media Laboratory
Thesis Supervisor

Accepted by: ____

Stephen A. Benton
Chairman, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

Synthetic News Radio: Content Filtering and Delivery for Broadcast Audio News

by

Keith Jeffrey Emnett

Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning,
on May 7, 1999 in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences

ABSTRACT

Synthetic News Radio uses automatic speech recognition and clustered text news stories to automatically find story boundaries in an audio news broadcast, and it creates semantic representations that can match stories of similar content through audio-based queries. Current speech recognition technology cannot by itself produce enough information to accurately characterize news audio; therefore, the clustered text stories represent a knowledge base of relevant news topics that the system can use to combine recognition transcripts of short, intonational phrases into larger, complete news stories.

Two interface mechanisms, a graphical desktop application and a touch-tone drive phone interface, allow quick and efficient browsing of the new structured news broadcasts. The system creates a personal, synthetic newscast by extracting stories, based on user interests, from multiple hourly newscasts and then reassembling them into a single recording at the end of the day. The system also supports timely delivery of important stories over a LAN or to a wireless audio pager.

This thesis describes the design and evaluation of the news segmentation and content matching technology, and evaluates the effectiveness of the interface and delivery mechanisms.

Thesis Supervisor: Christopher M. Schmandt
Title: Principal Research Scientist

Thesis Readers

Reader: ____

Professor Rosalind W. Picard
Associate Professor of Media Arts and Sciences
MIT Media Arts and Sciences

Reader: ____

Walter Bender
Senior Research Scientist
MIT Media Arts and Sciences

Acknowledgements

One of the greatest strengths of an academic institution is the people it brings together. MIT attracts the best and the brightest, and creates an atmosphere that truly challenges you in all aspects of life and learning. Many people have helped and influenced me during my stay here, and I would like to give special thanks to:

- Chris Schmandt, who has been my advisor for the last two years. He gave me the opportunity to study at MIT, and introduced me to the world of speech interface research. He created a group that focuses on how people really use speech and communication technologies, and examines how these technologies can affect everyday life. Thanks for this great experience!

- Members of the Speech Interface Group - Stefan, Natalia, Nitin, and Sean. Natalia, my office-mate, spent countless hours discussing ideas of all sorts with me (both research-related and not). Stefan put up with me as a roommate for the last year and provided a unique psychological viewpoint on many different topics (and helped me with my German as well!). Nitin provided useful advice and guidance as the only senior member of the group. And Sean, as the most recent addition to the group, brought an infusion of new ideas and perspective. All these people had a hand in labeling data, evaluating my project, and reviewing my thesis.

- Rosalind Picard, who served as a reader for my thesis. Her technical knowledge and disciplined approach to research helped my work tremendously. Roz truly leads by example - thanks for inspiring me to do my best by giving me yours.

- Walter Bender, director of the News in the Future consortium and also a reader for my thesis. Walter challenged me to examine how my project fit into the larger scheme of things. He was also a great supporter of collaboration with others at the lab doing similar work.

- Kimiko and Dean, who also helped evaluate my system.

- Matt and Amay, undergraduate researchers in the Speech group, who developed the server application for IBM ViaVoice.

- The sponsors of the MIT Media Lab, especially ABC Radio News for use of their audio and text news feeds, and IBM for use of their ViaVoice speech recognition package.

- My parents, Lee and Barbara, and my sister, Kristine. I cannot thank them enough for their love and support throughout my life. Thanks for repeatedly asking the question, "So how's the thesis?"

- God and the Christian communities of which I am a part - FLC (choir, care-group, and LBF) and LEM. Thanks for encouraging my spiritual as well as intellectual growth.

- The rest of my family and friends, both near and far.

Table of Contents

1. Introduction

There is increasing interest in browsing and filtering multimedia data. The most difficult part of this problem lies in defining the "content" of multimedia data. Some solutions explored in the past include using elements within video, such as character and scene identification, as well as using cues to the content of speech through closed-captioned annotations [1][2][3]. When analyzing only the audio portion of broadcast news, closed-captioning can provide a semantic basis for content analysis, but it requires human annotation at the source of the news distribution, something not routinely done for radio news. In addition, a system utilizing captions must synchronize the text with the audio phrases, as this is not done automatically.

Automatic speech recognition (ASR) can provide the ability to automatically derive semantic content from the speech in a news broadcast. Current commercially available continuous speech recognition software, however, has been designed with dictation as its primary application. It requires a somewhat unnatural speaking style, even though it still supports continuous speech, to achieve high recognition accuracy. When applied to broadcast news, which occurs with more natural and less-than-ideal patterns of speech, commercially available ASR engines misrecognize many words and omit many others. This makes it difficult to use transcripts alone to find story boundaries within an audio stream, and to provide accurate representations of the content.

Synthetic News Radio (SNR) uses additional information (transparent to the user) to help solve this problem, and therefore enables applications in which a user can make content decisions based solely on the broadcast audio news. Through clustering electronic text stories into the main topics of current news, this project can better define story boundaries by comparing these clusters to the audio news transcription. It can then create semantic vector representations for each story within a news broadcast, and use a vector similarity measure to find other relevant stories. This thesis explores the design of applications that support browsing and filtering of structured news audio, and presentation and delivery of important stories. It creates a personal, *synthetic* newscast by extracting stories, based on user interests, from multiple hourly broadcasts and then reassembling them into a single recording at the end of the day. Users can make content-based decisions while listening to a news broadcast, with no other information presented to them. Interaction is via graphical and telephone based interfaces, with newscasts delivered over a LAN or to wireless audio pagers.

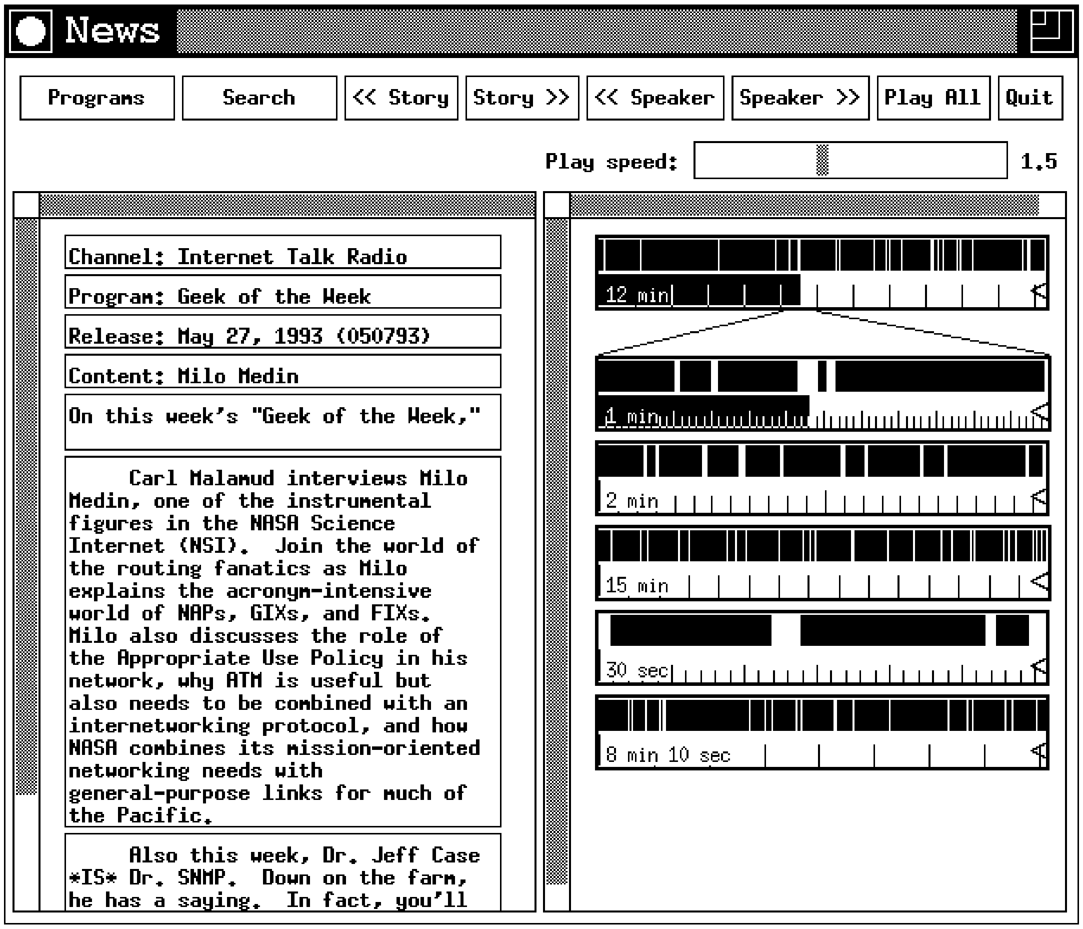
This paper quantitatively evaluates the performance of the systems designed for this task, and subjectively evaluates the effectiveness of the interface and delivery mechanisms designed around this technology.

1.1 Related Work

This section reviews several projects that address issues related to this thesis, identifying both similarities and differences.

1.1.1 NewsTime

Horner's *NewsTime* project structures the audio track of television newscasts by locating pauses and by extracting information from the accompanying closed caption data stream [1]. Story and speaker changes are explicitly defined in the closed caption data, and the system uses keyword detection to tag stories with keyword matches to predefined topics. A graphical interface shows a visual representation of the audio in one window and the time-synchronized closed caption text in another. Icons indicate speaker changes, story changes, and topic locations. The user can select an icon to jump to the corresponding location in the audio stream. The interface, shown in Figure 1.1 supports browsing and story highlighting of structured audio.



Channel: Internet Talk Radio

Program: Geek of the Week

Release: May 27, 1993 (050793)

Content: Milo Medin

On this week's "Geek of the Week,"

Carl Malamud interviews Milo Medin, one of the instrumental figures in the NASA Science Internet (NSI). Join the world of the routing fanatics as Milo explains the acronym-intensive world of NAPs, GIXs, and FIXs. Milo also discusses the role of the Appropriate Use Policy in his network, why ATM is useful but also needs to be combined with an internetworking protocol, and how NASA combines its mission-oriented networking needs with general-purpose links for much of the Pacific.

Also this week, Dr. Jeff Case *IS* Dr. SNMP. Down on the farm, he has a saying. In fact, you'll

Figure 1.1: NewsTime user interface.

Synthetic News Radio defines a similar structure for audio news and supports similar story navigation techniques; however, advances in ASR technology allow it to define structure even without the availability of closed caption data. Also, *SNR* saves semantic representations of each story that it can use to match one story with another. So rather than specifying keywords to match, users can simply select an audio story of interest. Because it used closed captions, NewsTime was useful only for television news; SNR works with the more frequently broadcast radio news.

1.1.2 NewsComm

Roy's *NewsComm* system structures audio from various sources including radio broadcasts, books and journals on tape, and Internet audio multicasts [4]. Content might include newscasts, talk shows, books on tape, and lectures. He defines the audio structure by locating pauses and speaker changes in the audio. Users can then download selected stories from an audio server to a hand-held audio browsing device. An audio manager decides which recordings to download based on a user preference file and on the recent usage history uploaded from the device. Users can navigate by jumping forward or backward to the meaningful structure points. Figure 1.2 shows a picture of the NewsComm device.

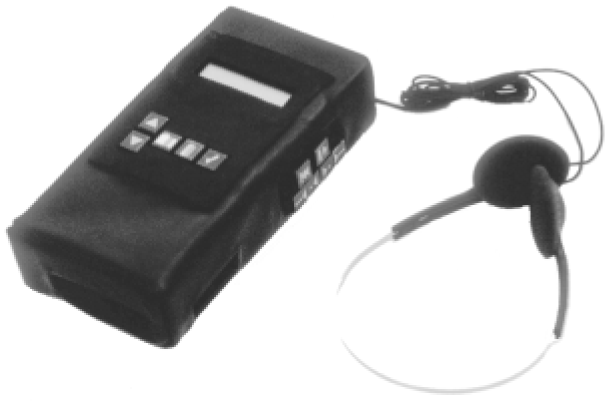


Figure 1.2: Picture of the NewsComm device.

This method allows automatic structuring of audio streams without any additional information and defines useful navigation points for browsing. *Synthetic News Radio* restricts the audio domain to newscasts; but, takes the structuring approach one step further and identifies story boundaries as the most meaningful index marker for navigation.

1.1.3 A Video Browser That Learns by Example

Wachman developed a video browser that uses high-level scripting information and closed captions to annotate scenes containing a particular character, characters, or other labeled content [2]. Furthermore, this browser applies machine learning techniques to improve the segmentation based on feedback from the user. The learner interactively accepts both positive and negative feedback of content labeled by the user, relating these to low-level color and texture features from the digitized video. The domain is the television situation comedy "Seinfeld." This project imposes a video structure based on character appearances in scenes, arguing that this is a good representation for browsing video of this sort. It is similar to SNR in the way it uses automatic clustering for similarity detection. Differences include the domain (video vs. audio) and the video browser’s ability to learn from user feedback.

1.1.4 CMU Informedia

The Informedia digital video library project uses automatic speech recognition to transcribe the soundtrack of an extremely large online video library [3]. It uses this transcription, along with several other features, to segment the library into "video paragraphs" and to provide indexing and search capabilities. User can retrieve video segments which satisfy an arbitrary subject area query based on words in the transcribed soundtrack. It also supports a structured browsing technique that enables rapid viewing of key video sequences. SNR's use of ASR transcriptions to match audio sequences is similar to the way this project uses transcriptions to match video sequences, but rather than a text-based search, SNR defines audio-based queries that serve as filters for future broadcasts, instead of search criteria for past articles in a database.

1.1.5 AT&T SCAN

Researchers at AT&T labs recently developed SCAN, the Speech Content Based Audio Navigator [5]. This system supports searching and browsing of a large database of news audio. The SCAN search component (global navigation) uses acoustic information to create short "speech documents," and then uses ASR to obtain an errorful transcript of each document. Document transcripts are matched to keyword queries through the SMART information retrieval engine. A search returns a list of programs with the highest relevance score, and when the user selects a program, such as an NPR news broadcast, the interface displays the transcript for each short speech document in that particular program. It also displays a histogram that indicates the number of query words found in the transcript of each speech document. Users can select part of the histogram to initiate playback of the corresponding speech segment. SNR uses a very similar technique to derive semantic information from speech audio. SCAN supports search and retrieval of a news archive through text queries, whereas SNR users select audio segments as queries to match stories in future broadcasts and develop personalized newscasts.

1.2 SNR System Architecture

SNR segments an audio news broadcast into different stories and creates a semantic vector representation for each story, used for content-based retrieval of news audio. A Sun Sparc10 digitally records an ABC News satellite news feed, used for radio affiliate news broadcasts, at a 16 kHz sampling rate with a 16 bit resolution. Another computer also receives text "Newscalls," which help segment and characterize the stories from the hourly broadcasts by providing a knowledge base of recent news topics. Figure 1.3 shows a block diagram of the system architecture.

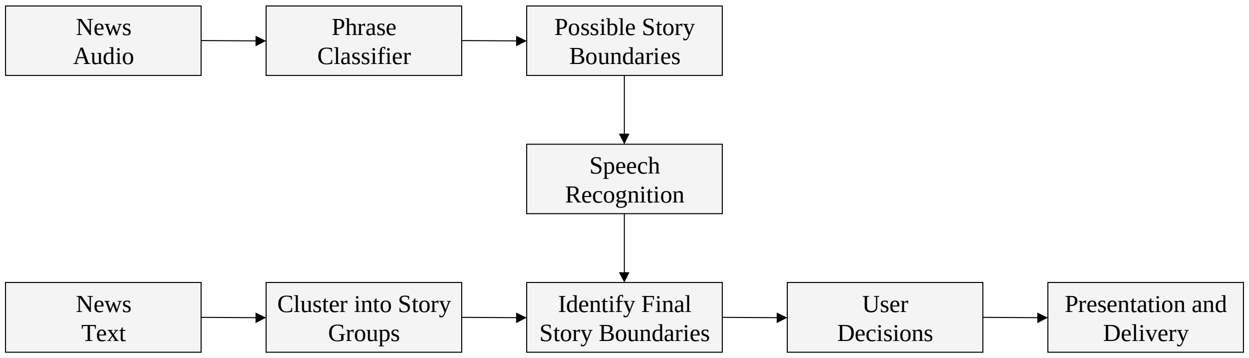

News Audio
Phrase Classifier
Possible Story Boundaries
Speech Recognition
News Text
Cluster into Story Groups
Identify Final Story Boundaries
User Decisions
Presentation and Delivery

Figure 1.3: System block diagram.

Unless otherwise indicated, components were coded in C on a Sun Sparc10 platform. The system uses a combination of two main techniques to find story boundaries in the news audio. First, an intonational phrase classifier identifies phrase boundaries based on features derived from the audio. This module uses libraries and utilities from the Entropic Waves software package. The identified phrase boundaries provide a set of possible story boundaries. Next, semantic features determine which consecutive intonational phrases belong to the same story. Each phrase is sent to a server running on a Windows NT computer, which transcribes the audio using IBM ViaVoice, and returns the transcript. Two undergraduate researchers in the speech group, Matt Hanna and Amay Champaneria, developed the ViaVoice server using C++ classes from the ViaVoice software developers kit.

The text Newscalls create a knowledge base that, by comparison to intonational phrase transcripts, helps identify story boundaries. A program running on a 500 MHz DEC Alpha uses the Newscalls to create a Latent Semantic Index (a measure of text similarity), utilizing the CLAPACK math library for matrix manipulation routines. The system then clusters these text stories into semantically similar story groups, and uses these groups to help combine intonational phrases that refer to the same general news story. Hourly news broadcasts are annotated with system derived story boundaries and the ASR transcription for each story, creating a news meta-structure useful for browsing and story retrieval.

User decisions and news delivery occur mainly through two interface applications. The graphical *xnews* X Windows application, developed with the Athena widget library and several audio libraries developed by the Speech Interface Group at the MIT Media Lab, uses the news meta-structure to provide advanced browsing and audio query selection. An interactive telephone application, which also supports structured browsing and query selection, uses audio and ISDN phone routines also developed by the Speech

Interface Group. A program then uses the listener's audio query selections to filter and rearrange future news broadcasts, creating a personalized, *synthetic* newscast tailored to the listener's own interests.

1.3 Document Overview

- Chapter two describes the design, implementation, and performance results of the intonational phrase classifier.
- Chapter three describes the generation of the Latent Semantic Index, and the design and evaluation of the Newscall clustering algorithm.
- Chapter four describes the process of using ASR transcripts from the intonational phrases along with the Newscall story clusters to find the story boundaries. It also discusses the technique used to match stories across news broadcasts.
- Chapter five describes the objectives and design of the user interfaces.
- Chapter six describes news delivery mechanisms and the construction of a synthetic newscast.
- Chapter seven presents the conclusions of this thesis and discusses future directions of the work.

2. Intonational Phrase Classifier

This thesis uses a multi-stage approach to automatically identify story boundaries in audio news broadcasts. The first part, an *intonational phrase* classifier, identifies *possible* story boundaries by breaking the news broadcast into a series of utterances. These utterances represent a basic unit of analysis in spoken language interpretation [6]. This assertion implies that a single phrase will express at most a single thought or idea. Story boundaries represent more drastic changes in the topic of a spoken discourse; therefore, phrase boundaries will occur more often than story boundaries. However, when a story boundary does occur, it should coincide with a phrase boundary. Previous studies have shown good results in the automatic identification of intonational phrases [6][7].

2.1 PHRASE BOUNDARY INDICATORS

Many researchers have identified features that indicate transitions from one basic unit of speech to the next. Chafe describes basic, functionally relevant segments of speech as *intonation units*. He states that any or all of the following can characterize intonation units: changes in fundamental frequency (perceived as pitch), changes in duration (perceived as the shortening or lengthening of syllables or words), changes in intensity (perceived as loudness), alterations of vocalization with silence (perceived as pausing), changes in voice quality of various kinds, and sometimes changes of turn [8]. Pierrehumbert and Hirschberg describe the primary unit of meaning analysis in speech as the *intonational phrase* [9]. In describing these phrases they distinguish:

- Stress **-** the rhythmic pattern of relative prominence of syllables in an utterance
- Tune – the abstract source of fundamental frequency patterns
- Phrasing – how a complex utterance is divided up
- Pitch range – the difference between the highest fundamental frequency value (F0) and the baseline value (lowest F0 value over all utterances by a single speaker)

Stress can be an important feature in finding emphasis and importance in acoustic data, however for my classifier I am interested only in finding acoustic structure at this point. In determining the phrasing, specifically the ends of intonational phrases, Pierrehumbert and Hirschberg succinctly define possible tune patterns, or ways in which the fundamental frequency can change. They call these patterns boundary tones, and in their notation use a **%** as a diacritic marker. Four possible tone combinations can indicate a boundary tone: **L L%**, **H L%**, **L H%**, and **H H%**. Here an **L** refers to a localized low value of F0 and **H** refers to a localized high value.

2.2 ACOUSTIC STRUCTURE

In addition to their use in identifying story boundaries, intonational phrases as a basic unit of structure present several other benefits in the context of this application. After the system identifies phrases, it will send these phrases one at a time through an automatic speech recognition (ASR) engine. An intonational phrase classifier is a useful front end for an ASR engine [6], and will hopefully improve recognition performance by providing salient acoustic and semantic boundaries. Intonational phrases also serve as a good basis unit for audio browsing applications [6][7].

2.3 PAUSE DETECTION

I initially investigated a simple classifier that segments audio recordings into intervals of speech and non-speech audio. Arons created a speech detection algorithm that uses time-domain measures of energy and zero-crossings to segment a speech recording into intervals of speech and background noise [10]. An initial pass through the speech data determines a noise floor threshold, and classifies all audio above the threshold as speech. Additional passes then use a *fill-in* and *hangover* technique to smooth the previously identified speech and noise intervals [11]. An analysis of the ABC news broadcasts that this project uses shows that this technique will identify some of the story boundaries; however, story boundaries sometimes occurred without an associated pause. Boundaries without associated pauses would occur with a change in the fundamental frequency of a single speaker, a jump in F0 associated with a change of speaker, a change in localized energy, etc. Another major reason why simple pause detection did not work well is that it is difficult to accurately combine the short identified phrases into longer segments, due to a minimum phrase length restriction.

2.4 MINIMUM PHRASE LENGTH

A later stage of the system imposes one additional design constraint: the classifier must produce phrases longer than a minimum phrase length of four seconds. The subsequent stage calculates statistical semantic correlations for the transcript of each intonational phrase. A minimum phrase length is required to ensure the transcription is long enough, or that it contains a sufficient amount of semantic information, to calculate valid and useful correlations. Pauses by themselves are not good indicators of topic shifts in spoken discourse. However, pauses combined with other features, especially fundamental frequency contours, offer the potential for much better performance.

2.5 FRAME CLASSIFICATION APPROACH

In my first attempt to use more features I created a system similar to the one designed by Hirschberg and Nakatani [1]. Their classifier uses a straight pattern recognition approach to identify 10 ms frames of audio as either IN_PHRASE (inside an intonational phrase) or IN_BREAK (inside a break between intonational phrases), and then uses a smoothing algorithm to finalize the selection of phrase boundaries. Performance on the ARPA/NIST HUB-IV Broadcast News database showed recall results as high as 0.95 and precision as high as 0.79 from classifiers trained on a different corpus, the Boston Directions Corpus of task-oriented monologue speech. I also used the get_f0 utility from the Entropic Waves software package to extract useful features from the news audio. Get_f0 implements a fundamental frequency estimation algorithm using the normalized cross correlation function and dynamic programming. The algorithm is similar to the pitch tracking algorithm described by Secrest and Doddington [12]. Get_f0 determines the following values:

- F0 – fundamental frequency estimate
- PROB_VOICE- probability of voicing (binary value of 0 or 1)
- RMS – rms energy value
- AC_PEAK – peak value of cross correlation

Figure 2.1 shows an Entropic Xwaves window, which displays the data obtained from get_f0 superimposed over the original audio file.

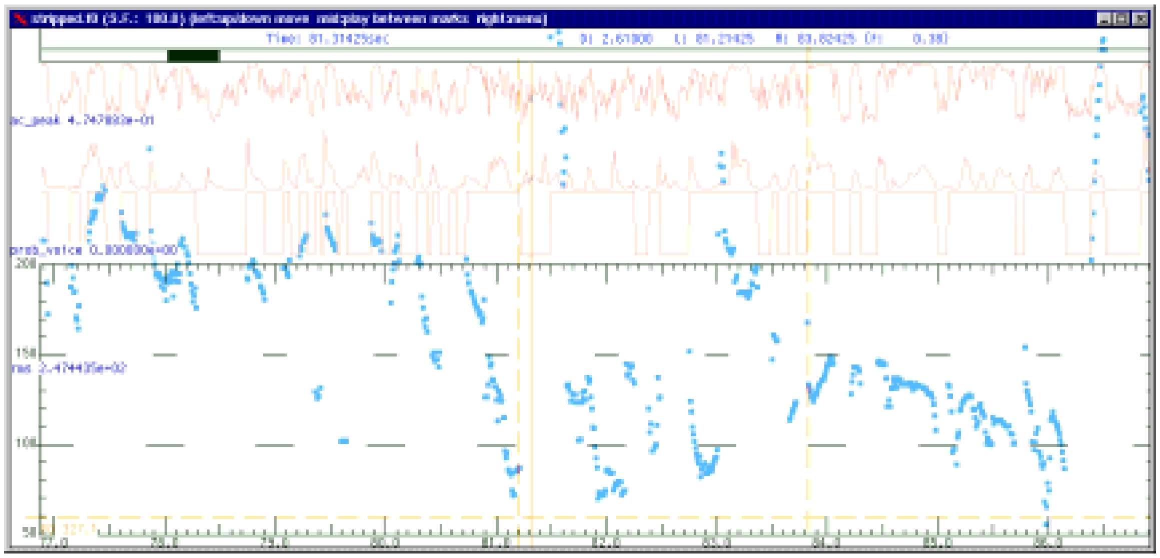


Figure 2.1: Entropic xwaves analysis tool showing estimates for fundamental frequency, rms energy, probability of voicing, and AC peak cross-correlation.

Using the Xwaves display I hand-labeled approximately nine minutes of news audio from three different ABC News broadcasts, classifying each 10 ms frame as either IN_PHRASE or IN_BREAK. I followed the ToBI labeling standard [13], which describes four tiers for prosodic labeling: tones, orthographics, break indices, and miscellaneous. The ToBI standard states that break indices "represent a rating for the degree of juncture between each pair of words and between the final words and the silence at the end of the utterance." I used only the break indices tier and classified frames as IN_BREAK if they matched a level four break index. A level four break represents the boundary of a Pierrehumbert and Hirschberg defined major phrase. The end of a major phrase is marked by a boundary tone (**H%** or **L%**).

Along with the basic features listed above, I also derived normalized and multi-frame based features. The multi-frame features were averaged over a window of about 20 frames to provide context information. I then used a CART (Classification and Regression Tree Analysis) algorithm to derive a classification tree based on the hand-labeled training data [14]. CART deduces classification criteria from training data that will maximize the number of correctly classified data points in the training set.

Using different combinations of features and testing via cross-validation on the training data, I achieved frame classification accuracies between 68% and 91%. Higher rates were achieved when training and validating a single broadcast, but performance dropped significantly when training data from multiple broadcasts was combined. In addition to poor results across multiple broadcasts, I found that, in general, the classifier did not produce enough accuracy to develop a smoothing algorithm that could reliably find phrase boundaries - even in the training data of a single broadcast.

I believe this approach failed largely due to insufficient training data, and to a lesser degree, because of inaccurate labeling of the training data. Hand-labeling each 10 ms frame of audio news was a tedious and error-prone process, so I therefore decided to take a different approach to designing the phrase classifier and instead look for specific patterns in the audio indicative of phrase boundaries.

2.6 RULE BASED APPROACH

As an alternative to the frame classification approach, I designed a rule-based classifier that searches for several of the boundary indicators described earlier. I again used the get_f0 utility to determine useful features in the audio data. Table 2.1 shows an example of the raw data returned from the get_f0 program.

Table 2.1: Example data from features returned by the Entropic get_f0 program.

F0	PROB_VOICE	RMS	AC_PEAK
143.993	1	131.988	0.848857
142.622	1	135.237	0.906679
150.085	1	140.228	0.949933
151.228	1	143.431	0.986421
154.514	1	144.261	0.977107

I focused primarily on finding boundaries indicated by pauses (extended periods with PROB_VOICE = 0) and localized changes in fundamental frequency. I believed that localized changes in F0 would correlate well with the end-of-phrase boundary tones defined by Pierrehumbert and Hirscherg. To this end I used only two of the four features – unnormalized F0 and PROB_VOICE, along with two derived features. The derived features were window averages of F0 (FO_WINDOW_MEAN) and PROB_VOICE (PROB_VOICE_WINDOW_MEAN). The windows were 120 ms long (12 frames) and were right aligned with the current sample, so that only past information was examined. The final rules for finding phrase boundaries follow:

- Flag a localized frequency jump if (F0 > 1.8*F0_WINDOW_MEAN) or (F0 < 0.5*F0_WINDOW_MEAN)
- Declare a phrase boundary if PROB_VOICE_WINDOW_MEAN falls below 0.10 within 100 ms (10 frames) of a frequency jump

- Declare a phrase boundary if (PROB_VOICE = 0) continuously for 350 ms (35 frames)
- Combine phrases to meet a minimum phrase length of four seconds
- Attempt to combine across phrase boundaries first, choosing shorter pauses over longer pauses
- If a phrase is still too short, combine across the frequency jump with the shortest associated pause

2.7 FIXED BOUNDARIES

The ABC News broadcasts occur once an hour, last five minutes, and are surrounded by silence. I defined several fixed phrase boundaries within each broadcast. To find the beginning of each news segment a simple algorithm averages the samples of a period of known silence to find the ambient noise level, then searches for a 20% deviation from that level to signify the start of the broadcast. Two commercial segments occur at relatively constant offsets and lengths from the start of the news. The system automatically removes these from the file and declares phrase boundaries at their endpoints.

2.8 EVALUATION CRITERIA

The primary purpose of the intonational phrase classifier is to provide a list of possible locations for story boundaries while maintaining a minimum length for each identified phrase. To evaluate performance based on these criteria, I created metrics that quantify the number of story boundaries found exactly as well as the average error for boundaries not found exactly. I also evaluate the performance with the minimum phrase restriction removed, which shows how much error this restriction introduces.

2.9 RESULTS

I compiled the following results from a single day (March 24, 1999) of news broadcasts from 8:00am to 6:00pm. I did not include the 2:00pm broadcast, or the first segment (120 seconds, up until the first commercial break) of the 3:00pm – 6:00pm broadcasts because each consisted only of a single story. These stories were unusually long because they covered the start of the NATO air offensive against Yugoslavia. The total data set includes 26 minutes of news and 49 actual internal story boundaries representing 65 stories. The remaining boundaries coincided with the starts and ends of commercials and the broadcasts themselves. Table 2.1 summarizes the results and Figure 2.2 shows a histogram of phrase lengths. The bins are one second each beginning with the value below each bin.

Table 2.2: Results from analysis of the intonational phrase classifier.

Total phrases identified:	188
Average phrase length:	8.30 seconds
Minimum phrase length:	4.02 seconds
Maximum phrase length:	22.39 seconds
Story boundaries found exactly:	31/49 = 63%
Average error for missed boundaries:	3.05 seconds
Average error for all boundaries:	1.12 seconds

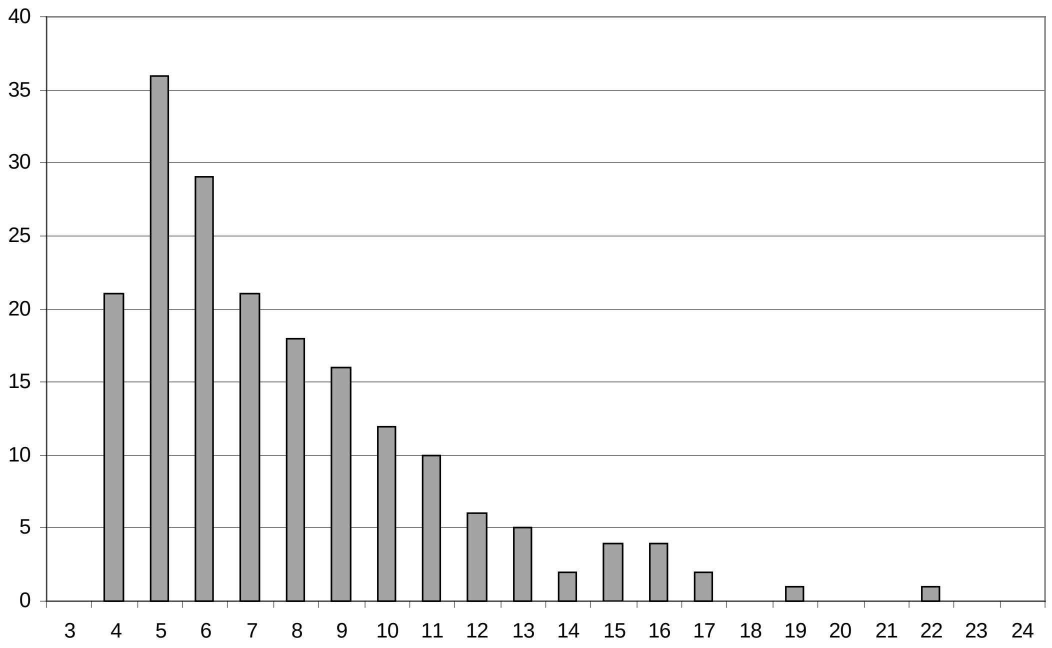


Figure 2.2: Lengths of phrases identified by the intonational phrase classifier, separated into one second bins.

For every actual internal story boundary the system found 3.8 phrase boundaries. The average (true) story length was 24 seconds, and the average identified phrase length was 8.3 seconds. The average error for missed boundaries, 3.05 seconds, is the average amount of time between an actual story boundary that was missed by the system and the closest phrase boundary automatically identified by the system. An analysis of the missed story boundaries shows that some did not have noticeable phrase breaks as the announcer maintained speech momentum through the end of one story into the beginning of the next. More often, however, phrase breaks were initially identified but later lost due to the combination of phrases to meet the minimum phrase length. This is shown by the fact that the average error for missed boundaries, 3.05 seconds, is less than the minimum phrase length of four seconds. An analysis of the classifier with the minimum phrase length restriction removed shows that 11 of the 18 missed boundaries were originally found by the classifier, with an average error of 2.15 seconds for the remaining seven. Overall, the minimum phrase length restriction was not very detrimental because the content of a story as a whole can still be understood without the first or last few seconds of the story.

The frame classification approach could offer better performance with significantly more training data; however, it will still be hampered by the same minimum phrase length restriction. Furthermore, phrase combination may be more difficult because the regression tree method cannot easily identify the reasoning behind a boundary decision, i.e. pause, localized frequency jump, change in rms energy, etc. Further testing of this approach is needed to determine if it can find phrases that coincide with story boundaries better than the rule-based approach.

3. Text Similarity and Newscall Clustering

The intonational phrase classifier described in the last chapter provides many possible story boundaries. In order to better discriminate among these possibilities, and eventually match stories from one broadcast to another, the system needs a way to derive semantic information from the news audio. Chapter four will describe how *Synthetic News Radio* uses automatic speech recognition to obtain this semantic information, and it will also demonstrate that an additional source of contextual knowledge is ultimately needed to help combine intonational phrases into full stories, and to match one story to another.

This chapter describes the use of text news stories, called *Newscalls*, to provide a basis of knowledge about the most prominent news stories over a short period of time - up to one day. An ABC satellite news feed provides Newscalls to affiliate radio stations to allow them to assemble their own newscasts with local announcers. A Newscall consists of an audio portion, typically a sound bite from a prominent person in the news, and a text portion, which gives a brief contextual description for the sound bite along with a transcription of the audio quote, if applicable. Figure 3.1 shows an example of the content of a text Newscall.

```
FOR REASONS OF MONEY, SAFETY, AND LESSENING THREAT, THE  PENTAGON IS
CONSIDERING A PLAN TO CUT BACK THE NUMBER OF  NUCLEAR WARHEADS IN
AMERICA’S ARSENAL. ABC’S BARBARA STARR HAS  DETAILS FROM THE
PENTAGON... (START TREATY) :18

VERBATIM: ’’THE PROPOSAL COULD SAVE HUNDREDS OF MILLIONS OF  DOLLARS
OVER THE NEXT FIVE YEARS. IF APPROVED BY CONGRESS, THE  PENTAGON WOULD
SLASH THE U.S. ARSENAL OF NUCLEAR WARHEADS BY NEARLY  HALF.... TO
ABOUT 3000 WEAPONS. THE DEFENSE DEPARTMENT IS  CONSIDERING THE PLAN IN
PART, BECAUSE RUSSIA’S PARLIAMENT HAS YET  TO RATIFY THE START
TREATY.’’
```

Figure 3.1: Sample text portion of an ABC Newscall.

The system discards the audio portions but uses the text segments to define the important news stories specific to the present time. An algorithm clusters these Newscalls, based on their semantic similarity, into larger groups which each represent a single major news story. The system will later use these clusters to help group intonational phrase transcripts into stories as well as match stories across multiple broadcasts. The following sections first describe the methods used to create semantic vector representations for each segment of text, and then describe the development of the algorithm used to cluster the text Newscalls.

3.1 MEASURING SEMANTIC SIMILARITY

Finding and improving ways to determine the semantic similarity of text documents is an active research topic. Salton developed one of the original text matching systems, SMART, in the late 1960s [15]. SMART creates a *term vector* for each document based on the frequency of each word (term) within each passage. The form of this vector for a document *d* with a total of *m* words in the entire document collection is

$$f_d = [t_1, t_2, \ldots, t_m]^T$$

where each t_k represents the number of times term *k* occurs in document *d*. Next terms are weighted in some way to provide distinctions among them. Weighting functions alter the relative importance of individual terms within the entire collection. The weighted vector becomes

$$w_d = [w_{f1}t_1, w_{f2}t_2, \ldots, w_{fm}t_m]^T$$

The SMART system uses the following vector similarity formula to mathematically determine the similarity between two documents f_a and f_b:

$$similarity(f_a, f_b) = \frac{\sum_{i=1}^{m} w_{ai} \cdot w_{bi}}{\sqrt{\sum_{i=1}^{m} (w_{ai})^2 \cdot \sum_{i=1}^{m} (w_{bi})^2}} = \frac{w_a^T w_b}{|w_a||w_b|}.$$

This method ignores word order within a document, and simply calculates how close two documents are in the term-document space. The choice of weighting system significantly affects the overall retrieval results [16]. Some newer methods [17] take into account the word order, although others argue that the majority of the information about the meaning of a passage can be carried by words independently of their order [18].

3.2 LATENT SEMANTIC INDEXING

Latent Semantic Indexing (LSI) [19], or Latent Semantic Analysis (LSA), is a variant of earlier information retrieval techniques such as SMART. It differs from those methods in that it also extracts information about the natural co-occurrences of words. It does not use any information about transition probabilities from word to word, but instead uses a singular value decomposition (SVD) to deduce relationships between words that appear in large natural passages. These passages presumably contain utterances of the kind people use to convey complete ideas. Documents are represented with the same term vectors and weighted in similar ways; however, LSI uses an entire collection of domain representational texts to form a large matrix A, which will be used to transform the term vector representation for each query document. LSI creates this transform matrix, or Latent Semantic Index, by performing an SVD on the matrix A and then reducing the dimensionality by keeping only the vectors associated with the most significant singular values. This analysis "induces human-like relationships among passages and words [19]." Comparisons to conventional methods have shown LSI perfomance ranging from comparable to 30% better [20]. I use the collection of recent Newscalls as the domain specific knowledge used to create the index. Section 3.5 describes the exact process in more detail.

3.3 STEMMING

When analyzing Newscalls to create an index, the first step of the process stems words to remove morphological variations. Plural endings, adverb endings, verb forms, etc. do not significantly alter the meaning of a word; therefore, these are removed so that all forms of a word are seen as identical to the system. I used a C implementation, developed by Frakes and Cox, of the Porter stemming algorithm [21].

3.4 REMOVING COMMON WORDS

The next step of the process removes words that appear on a list of 578 common words. These words appear often and do not carry a significant amount of semantic meaning. Some examples include: *also*, *after*, *and*, *did*, *do*, and *each*. I used the list of common words implemented in the SMART system, along with some additions specific to Newscalls, such as *ABC* and *verbatim*.

3.5 CREATING THE INDEX

A program automatically generates a new Latent Semantic Index once an hour throughout the day, using all Newscalls received for the current day. After words have been stemmed and the common ones removed, the following steps describe the procedure for generating an index [19].

(1) A term vector is created from each Newscall, and then all the term vectors are combined into a matrix *A*, with total number of words *m* and number of Newscall documents *n*:

$$A = [f_1, f_2, \ldots, f_n].$$

Each row represents an individual word, and each column and individual Newscall. Once again each cell a_{ij} contains the frequency of the word *i* within the passage of text *j*.

(2) The next step transforms each f_d by term weighting functions [20] so that *A* becomes

$$\hat{A} = [w_1, w_2, \ldots, w_n].$$

Term weighting functions assign more importance to terms that discriminate better, and they generally have two components. A global weight, applied to an entire row, indicates that word's overall importance in the collection as an indexing term. A local weight, applied to each term a_{ij}, transforms the term's frequency within the document. Each weighted cell, $\hat{a}_{ij}$, within matrix $\hat{A}$ then becomes a weighted function of the original a_{ij}, according to

$$\hat{a}_{ij} = l(i, j) \cdot g(i),$$

where *l* is the local weighting as a function of the term and the document, and *g* is the global weighting as a function of each term (both detailed below). A comparison study [20] of different term weighting schemes for LSI demonstrated the best results with a log-entropy combination of local and global weights, respectively. Similar results were shown with this combination for standard vector retrieval methods [22]. The log local weighting,

$$l(i, j) = \log(a_{ij} + 1)$$

reduces the effects of large difference in frequencies. Like other global weights, the entropy weighting scheme gives less weight to terms that occur frequently in a large number of documents. Here, g_i is the global frequency, or the total number of times word i appears in matrix A:

$$g(i) = 1 - \sum_{j=1}^{n} \frac{p_{ij} \log(p_{ij})}{\log(n)} \qquad \text{where} \qquad p_{ij} = \frac{a_{ij}}{g_i}.$$

The amount of entropy (or noise) is scaled by *log(n)* and added to one, which assigns high weights to terms with less noise and lower weights to terms that are equally distributed across the collection (i.e. when $p_{ij} = 1/n$).

(3) After creating the weighted matrix $\hat{A}$ the system performs a singular value decomposition [23]

$$\hat{A} = U \quad S \quad V^T ,$$

in which U (m x k) and V (n x k) have orthonormal columns, S (k x k) is a diagonal matrix of singular values, $k <= \max(m,n)$, and $m > n$. Dimensionality reduction is achieved by keeping only the k largest singular values and setting the rest to zero. In my implementation I preserve the 20 largest singular values, a number determined through initial testing. This will also be the length of the reduced vectors calculated for each text passage.

(4) For each passage of text that the system needs to compare to others, it creates a term vector f_d, as in step (1). Words that do not appear in the index are not included in f_d. It then transforms the vector with the same log-entropy weighting scheme into w_d, using the same global weight *g(i)* for each term i calculated in step (2). The index transforms each passage of text into a reduced vector representation r_d (k x 1) via

$$r_d = \left(S^{-1} \quad U^T \quad w_d\right).$$

The cosine, or dot product, between two reduced vectors r_a and r_b scaled to unit length becomes a measure of their semantic similarity:

$$similarity(r_a, r_b) = \frac{r_a^T r_b}{|r_a||r_b|} .$$

The higher the cosine, the more similar the documents.

3.6 NEWSCALL CLUSTERING

Every hour, immediately after the system creates an index, it executes an algorithm to cluster the Newscalls into larger groups representing prevalent topics in the news. A few major stories will have a large number of Newscalls representing them, while others will have just a few. My first attempt at clustering the Newscalls used reduced vector representations in a basic iterative optimization technique with the minimized sum-of-squared-error criterion, J_e [24]. This method clusters the Newscalls (or their normalized semantic vectors, $r_1, r_2, \ldots, r_n$) into a predefined number of c classes (story groups), with n_k Newscalls in each class. The error criterion is determined by

$$J_e = \sum_{k=1}^{c} J_k ,$$

where

$$J_k = \sum_{i=1}^{n_k} \| r_i - m_k \|^2 ,$$

and

$$m_k = \frac{1}{n_k} \sum_{i=1}^{n_k} r_i .$$

The iterative optimization tentatively moves one Newscall to a different cluster and calculates a new value for J_e. It keeps moves that reduce J_e, and then repeats this process for all the Newscalls until J_e ceases to change.

One immediately apparent problem with this approach is that it requires the number of classes to be known. The system has no way of knowing how many different stories might appear in the news beforehand. For this algorithm it uses an empirically based assumption of n/5 classes, and initially makes random, evenly-distributed assignments of the Newscalls to these classes. This technique found accurate relationships between Newscalls; however, two major problems tended to arise. First, for stories represented by a small number of Newscalls per story, it would group many of these small story groups into a single large cluster. Second, for stories that had a very large number of Newscalls, it would break these groups apart into several large clusters.

An examination of the Newscalls showed a very high degree of similarity between Newscalls about the same stories. Many would have topic descriptions that were almost verbatim the same. This led to a two-stage approach consisting of 1) word matching to create initial Newscall clusters, and 2) using LSI to match the remaining unassigned passages. The final algorithm uses the following technique:

1. Group together Newscalls that have more than 15% identical words between them, counting only stemmed words that appear in the index.
2. For the each remaining Newscall, determine which already assigned Newscall the remaining one has the highest LSI correlation with, and assign it to the same cluster, as long as the highest cosine similarity value is > 0.7.
3. Flag any Newscalls left at this point and do not use them in the future.

This method precludes the existence of a single-story cluster; however, singleton stories are not likely to appear in the audio news broadcast.

3.7 RESULTS

The evaluation data consists of the Newscalls collected for one entire day (November 23, 1998). The Newscalls were grouped according to topic similarity by one person into a total of 36 story clusters. The above algorithm was then used to automatically cluster the Newscalls into a total of 23 clusters. Figure 3.2 shows the assignments from the hand-labeled data next to the assignments from the algorithm.

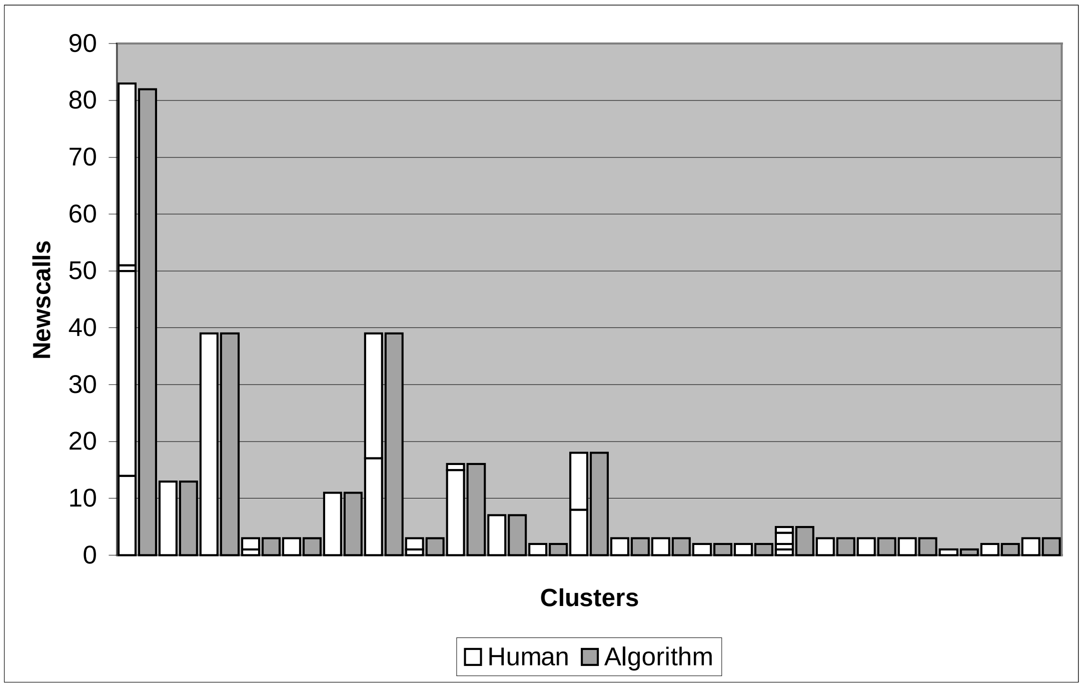


Figure 3.2: Results from analysis of the Newscall clustering algorithm. Bars with lines represent multiple clusters.

Each pair of bars in the figure represents the assignments of the exact same Newscalls, not just the same number of Newscalls. Bars with horizontal lines represent more than one cluster. The figure shows that, in some cases, the algorithm combined several clusters defined by the hand-labeler into a single large cluster. For example, the first cluster on the left shows that the person placed 83 Newscalls into four different clusters and that the algorithm placed 82 of those same 83 Newscalls into a single cluster. The extra Newscall was left unclassified by the algorithm. However, the algorithm rarely separated Newscalls grouped together by the labeler into different groups. This happened only once with the single Newscall that was left unclassified. These performance results mean that the system will lean towards identifying two unrelated stories as relevant rather than failing to match two stories that are indeed the same.

Further tests, performed when designing the procedures in the next chapter, showed that disproportionately large clusters still tend to produce high correlations with unrelated stories, so the final version of the clustering algorithm limits the number of Newscalls per cluster to 25, and discards the remaining ones. The Newscalls are generally evenly distributed so that if a large cluster actually contains more than one story, eliminating extra Newscalls from it does not eliminate an entire story.

4. Finding Story Boundaries

The last two chapters described the process of identifying intonational phrases and of clustering text Newscalls by creating semantic vector representations for each passage of text. This chapter discusses the final step in finding story boundaries - using the information provided by the Newscall clusters to properly combine the intonational phrases into full stories. This phase uses automatic speech recognition (ASR) software to transcribe each intonational phrase. Vector representations of these phrases, created using the Newscall-generated Latent Semantic Index, are then matched to story clusters to help determine which phrases are likely about the same story.

4.1 Automatic Speech Recognition

Synthetic News Radio uses a large-vocabulary continuous speech recognition engine, specifically a commercially available product, IBM ViaVoice. There are many different speech recognizers currently available, and each has strength and weaknesses dependent on the target application. The complexity of the expected speech determines the required vocabulary size. Most command-and-control applications (programs that use speech input to control interface commands) use small-vocabulary recognition systems, where the acceptable input is narrowly defined to a few commands in a given context. For example, a desktop word processor could accept commands such as "save file" or "copy text," and the acceptable commands could change as the user moves through different functions of the program. This helps limit recognition errors by reducing the number of possibilities. These limitations also make these recognizers speaker-independent, meaning the application can recognize commands from many different people, even though their speech may vary significantly. This happens because the recognition constraints are much looser for small-vocabulary systems than for large-vocabulary systems.

Large-vocabulary systems (typically >10,000 words), on the other hand, must discriminate among a much larger set of possibilities. These systems are designed to understand formal, spoken discourse, and are targeted mainly at dictation into a word processor. Older large-vocabulary, discrete recognizers required pauses between each word. Only through relatively recent improvement in large-vocabulary, continuous speech recognition systems, has recognition of broadcast news become possible. Continuous speech recognizers can process normal speech and determine for themselves the locations of the word boundaries. Dictation recognizers can operate as speaker-independent systems; however, their accuracy improves when trained by a specific speaker using a known text. Nevertheless, a combination of informal speech (including slang, non-speech utterances, music, and environmental sounds), frequent speaker changes, and the inability to train the recognizer on individual speakers results in poor transcription performance on broadcast news. Figure 4.1 shows a sample transcription from an intonational phrase about the air strikes in Yugoslavia, generated by IBM ViaVoice.

```
Back and half from ABC News LAN had math and I didn’t.  Limits is not
going to begin a unlike the Gulf War U.S. led NATO air strikes and
Syria could come anytime an ABC military analysts on the court is in
cells is a dangerous mission water’s going to be bad that terrain is
mountainous
```

Figure 4.1: Example transcription of audio news by IBM ViaVoice.

A study of ASR performance on broadcast news showed wide variations, from a maximum of 88% words correctly recognized to a minimum of 35%, with a mean of 67% [5]. ViaVoice does not offer much additional information. It does not return any measure of confidence, nor does it give any indication if it fails to recognize anything from certain portions of audio. New proper nouns in the news, especially names, sometimes contribute significant error because they can not be recognized correctly. Using semantic information from the transcripts alone is not enough to accurately find story boundaries, so the system uses additional information in the form of the text story clusters. Although there is currently a strong DARPA/NIST-funded research effort to improve recognition accuracy for broadcast news, high accuracies are not really needed for this project. The goal is not to obtain an accurate transcript, but simply to extract enough semantic information to create a representation that the system can use to find relevant stories. The interfaces all provide ways for users to interact with the actual audio news, because audio is still a much richer medium of communication. Even if recognizers could provide perfect transcripts, they are still a long way from understanding all the ancillary information that speech has to offer, including tone, intonational contours, intention, emotion, and attentive states.

4.2 Grouping Intonational Phrases

Every hour the system generates a new Latent Semantic Index, and the most recent index is used to transform each intonational phrase transcript into a representational vector. The system then determines which Newscall cluster has the highest correlation with each phrase. The hypothesis is that sequential phrases about the same story will have high correlations with the same story cluster. A story boundary then occurs when the phrase correlations switch from one story cluster to another. Figure 4.2 depicts this process.

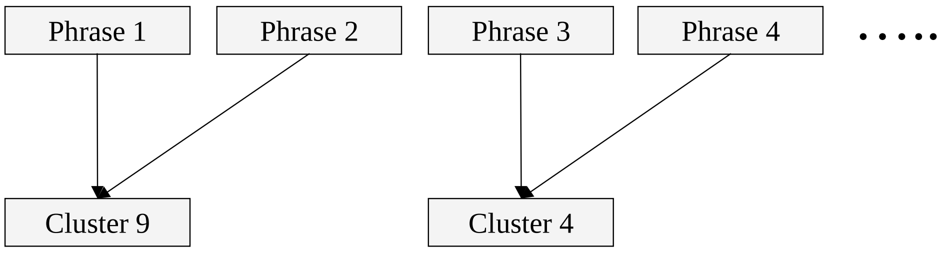


Figure 4.2: Illustration of the process used to combine intonational phrases into story groups by matching their transcripts with Newscall clusters.

The actual results do not look as clean as this figure, so the sequencing algorithm described in the next section attempts to locate the best matches and smooth out the errors.

4.2.1 Sequencing Algorithm

The sequencing algorithm generates a vector for each transcript according to the process outlined in section 3.5 step (4), using only words that appear in the index. This helps eliminate recognition errors, because words that don't appear in the index were most likely transcribed incorrectly. In one specific exception it searches for a story match based on keywords. It does this for Wall Street stock market updates, because results showed that this story almost always occurs in the broadcast, but generally has very few, if any, Newscalls associated with it. The algorithm uses both the similarity cosine and also the absolute magnitude of the vector (measured before it is scaled to unit length for the cosine determination). Vectors with low magnitudes generally do not have enough semantic information to make accurate matches. For each intonational phrase, the system finds the story cluster that has the highest correlation (cosine), and then uses the following sequence of steps to make a story assignment to each phrase. A story cluster (or simply cluster) is a clustered group of Newscalls, and a story group is a group of intonational phrases that the algorithm decides refer to the same topic.

1. Attempt to identify a stock market story group by searching for keywords in the transcript of each phrase. Consecutive phrases with any of the following words in their transcripts are assigned to the same story group: "Dow", "Jones", "Industrials", "stock", "market", or "Wall Street".
2. Determine which cluster has the most phrases assigned to it, counting only phrases that have a cosine correlation > 0.7. Assign these phrases to the same story group. If these phrases surround any other

unassigned phrases, then group the unassigned phrases with the same story. For example, if phrases two and four have a high correlation with cluster four, then phrases two, four, and three are all grouped together as the same story. Typically the phrase(s) in the middle did not have enough information to provide a high correlation with the appropriate cluster. Repeat this step for the cluster with the second highest number of phrases assigned to it.

3. Make a new story assignment for any phrases that have both a high correlation (cosine > 0.9) with a story cluster and a strong magnitude (> 0.08).
4. Search for unassigned phrases that have been matched to the same story cluster as a neighboring phrase. If these phrases have a cosine > 0.7, then assigned them to the same story group as the neighboring phrase with the identical cluster assignment.
5. Search for any unassigned phrases that are surrounded by two phrases with story group assignments. If the unassigned phrase has the same cluster assignment as one of the neighboring phrases, assign the phrase to the same story group as the neighboring phrase; otherwise, create a new story group for the unassigned phrase.
6. If one to three phrases at the beginning of the broadcast do not yet have story group assignments, then assign them to the same group as the first assigned phrase in the broadcast. Many times recognition accuracy at the beginning of the broadcast is poor due to music and sound effects.
7. Assign any leftover phrases to new story groups, putting consecutive unassigned phrases with the same cluster assignment together in the same story group.

The hourly ABC news broadcasts have two main continuous segments - one before the first commercial break and one after. The preceding sequence of steps is used to assign story groups independently for both segments.

4.2.2 Results

The evaluation for this phase uses the same audio data analyzed in Chapter 2 for the intonational phrase classifier, as well as the corresponding Newscall clusters for that day (March 24, 1999). The segments omitted before are omitted again here. The data consists of 26 minutes of news, which includes 67 actual stories (identified by hand-labeling) and 188 phrases identified by the intonational phrase classifier. Table 4.1 summarizes the results.

Table 4.1: Results from the intonational phrase classifier analysis.

Total hand-labeled (HL) stories:	67
Total system stories:	90
Total HL internal boundaries:	49
Total system internal boundaries:	74
Accurate system boundaries:	26 / 49
Average error for missed boundaries:	4.0 seconds

I only evaluated performance on *internal* story boundaries, which do not include fixed boundaries defined by the beginning of the broadcast, the end of the broadcast, and the commercial breaks. The perceived performance is therefore somewhat better than the performance shown here because the fixed boundaries are always accurate. The average error for missed boundaries is about one second higher than the same error for the intonational phrase classifier. This shows that the algorithm would occasionally group together transcripts across a real story boundary; but, the large ratio of found boundaries to actual boundaries (1.5:1) shows that it was more likely to break a real story into multiple segments. An analysis of errors in grouping transcripts across a real boundary shows that the grouping was off by a single intonational phrase and that this phrase contained very little story-related semantic information, such as a speaker and location identification message. Errors in breaking apart stories into smaller segments occurred either due to lack of information in the ASR transcription (some times due to extended periods of live, natural speech, such as a witness account) or due to poor correlation with the appropriate Newscall cluster.

4.2.3 Audio Auxiliary File

The process of finding story boundaries automatically creates an auxiliary file for each news broadcast audio file. This contains a record for each story that includes the start time, stop time, and ASR transcription. Other programs can use an associated library to easily access this information, providing formal routines for manipulating the new, structured audio. Figure 4.3 shows an example of a story record.

```
{Record
{{StoryNum} {1}
{StartTime} {0}
{EndTime} {22540}
{Transcription} {At an altitude laugh hydraulic ACLU and have amassed
By lowering a hot Buttons: selling the reserves in housing loan is
quite well read a limited air strikes The heavy guard in the locker
rooms in just a few hours that when the NATO attacks Who began the
first wave expected to consist of cruise missiles and have won 17
stealth aircraft }
}
```

Figure 4.3: Story record from the audio auxiliary file.

4.3 Matching Stories Across Multiple Broadcasts

Matching a story from one broadcast to that of another broadcast works in much the same way as the sequencing algorithm. The transcripts of the selected stories are saved in the user query file. Figure 4.4 displays a sample entry.

```
FOLLOW  04/16/99  8:00   8  It was on the state ten years ago was
super tanker ran a ground just off the coast of Alaska the Exxon about
these 11 million gallons of oil spilled a little water and
environmental nightmare) that noble ideals for members that they and
says despite all the cleanup A dozen. Invite metals claims they didn’t
tell the allied effort tied at the type Restore a laugh because until
the beach that there were worse. The beach with hot war Bernstein They
built by blooms big employers cures for in barges we do that all
summer long High that no doubt that there’s still lots of oil down in
those rocks increases tied) told all yields the Exxon. The still in
service not transports oil the Mideast  ENDSTORY
```

Figure 4.4: Example entry from the user query file.

Every time the system tries to make a match, it creates a new semantic vector representation for the story using the most recently generated index. Next, it calculates cosine similarities for each defined story in a segmented broadcast. It matches the query to the story with the highest correlation, as long as that value is above a threshold of 0.7. One exception to this method occurs again with the stock market stories. For these queries the system attempts to match keywords, using the list identified in section 4.2.1 step 1.

A current limitation to this technique is its inability to detect subtle topic differences. It can determine if two stories generally convey information about the same topic; but, it cannot determine if a story has changed, or if one contains any additional information that a previous version did not have. I do not think information retrieval techniques have advanced to a point that makes this possible; and, future techniques will most likely need to include some measure of natural language processing. I have not found this limitation to be very detrimental to the design of this project; however, a more detailed topic analysis and understanding would certainly lead to more sophisticated browsing and filtering capabilities.

5. Interface Design

This chapter presents the structure of news audio created by *Synthetic News Radio*, and introduces user interfaces for browsing news and providing interactive feedback to personalize news on an ongoing basis. Previous chapters described the process of creating a news *meta-structure* that automatically annotates audio news broadcasts with story boundaries and semantic vector representations. Browsing applications, including a visual, desktop interface and a touch-tone driven phone interface, utilize the meta-structure to give users more control over how they listen to news. News structure allows a user to jump quickly from one story to the next, similar to accessing the tracks on a compact disc. Audio time compression simulates fast-forwarding, and also allows faster than real-time playback.

News personalization occurs through explicit feedback to the system based on audio heard by the user. The applications store the underlying semantic vector representations in a user query file for use in generating customized broadcasts. Users can also specify a variety of delivery mechanisms, including a complete, interactive, personalized news broadcast, playback of important stories on a desktop computer via a LAN, and transmission to a wireless audio device.

5.1 DESIGN CRITERIA

The interface designs focus around several key ideas. Having an underlying news structure influences the way people browse the news, and having pervasive access to the news from a variety of locations and situations influences the types of applications users can interact with.

5.1.1 Nonlinear Browsing

Typically people watch or listen to a news broadcast in a linear fashion. Content developers determine the presentation sequence, which takes into account their thoughts about which stories will appeal most to a broad and diverse audience, including the timeliness and importance of each story. The audience then listens to the broadcast non-interactively, which means that they have no way of changing or affecting the presentation in any way. Creating an underlying structure for audio news gives users control over the presentation. A simple structure that defines story boundaries within the audio suddenly creates drastic changes in the way people can listen to news. It makes nonlinear browsing possible. A listener can move quickly from one story to another, listening carefully to stories of interest, and skipping past stories of disinterest. One analogy between the normal linear method and *Synthetic News Radio's* non-linear browsing technique is listening to music over an FM radio broadcast versus listening to music on a CD player.

5.1.2 Unobtrusive Feedback Mechanisms

Along with automatically defining story boundaries, *Synthetic News Radio* saves a mathematical representation for each story transparent to the user. This allows people to provide feedback about an audio story without interacting with anything except the audio itself. The interfaces offer users the ability to mark particular stories they want to *follow* or *ignore* in the future. This informative feedback can be used to alter the way news is presented in subsequent newscasts.

5.1.3 Timely Access to Information

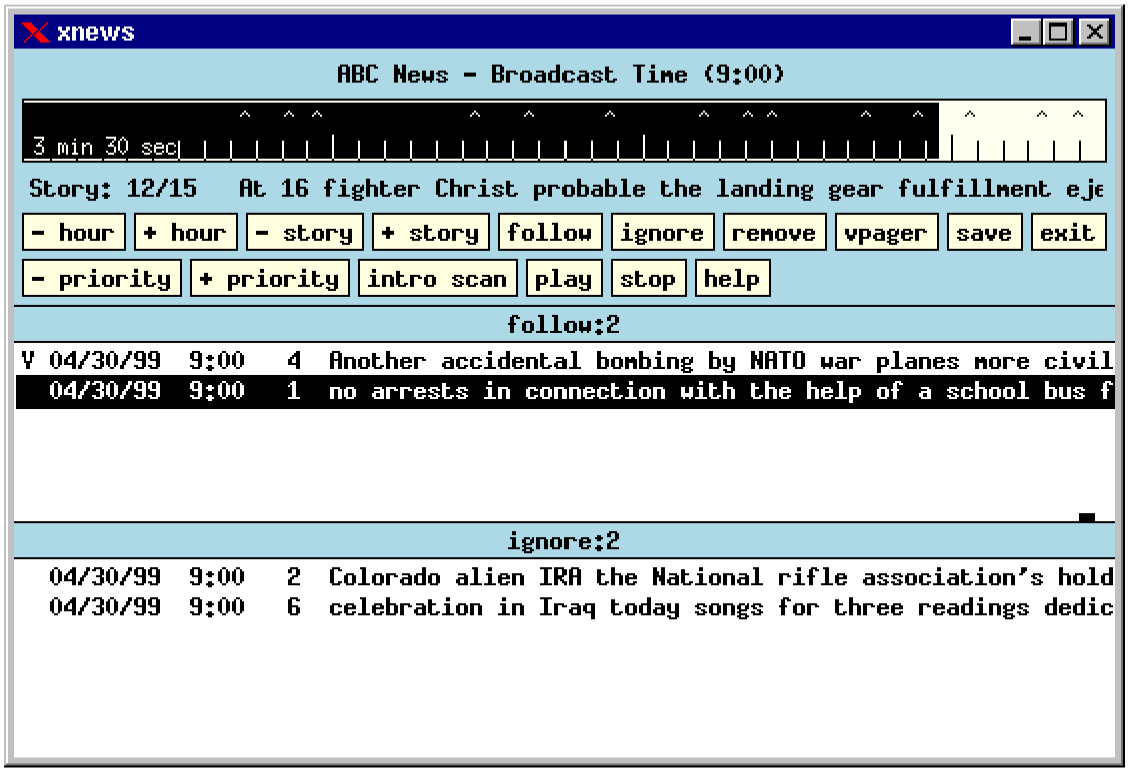
News changes constantly, and this system receives a new ABC news broadcast every hour. If a person really wants to stay updated, then the system will need to provide several different ways to access the news, which take into account the communication modalities available to the user in different situations. The design of the applications should pay strong attention to the capabilities of each communication medium.

5.1.4 Iterative User Evaluations and Design Changes

Many interface features were introduced, changed, or deleted through user evaluation sessions. These changes are noted along with the discussions on that each particular interface element. A total of seven people participated in evaluations at one time or another. Each evaluation session lasted approximately 30-60 minutes and consisted initially of using the two main interfaces to browse news broadcasts and select audio queries. Synthetic newscasts were then generated for some of the users, and additional evaluation sessions conducted later the same day, or early the next day.

5.2 VISUAL DESKTOP INTERFACE

A graphical X-Windows application, *xnews*, provides the primary interface for a user who is sitting at a desktop computer. Figure 5.1 shows the main window, and Table 5.1 briefly describes each of the commands. The following sections will describe each element in more detail.



Story: 12/15 At 16 fighter Christ probable the landing gear fulfillment eje

follow:2

```
V 04/30/99  9:00   4  Another accidental bombing by NATO war planes more civil
  04/30/99  9:00   1  no arrests in connection with the help of a school bus f
```

ignore:2

```
  04/30/99  9:00   2  Colorado alien IRA the National rifle association's hold
  04/30/99  9:00   6  celebration in Iraq today songs for three readings dedic
```

Figure 5.1: Graphical interface for the *xnews* application.

Table 5.1: Summary of *xnews* commands.

Command	**Definition**
- HOUR	Load the broadcast from the previous hour.
+ HOUR	Load the next hourly broadcast.
- STORY	Skip to the beginning of the previous story.
+ STORY	Skip to the beginning of the next story.
FOLLOW	Place the current story on the follow list. The system will use the representation of this story to match future audio news segments.
IGNORE	Similar to follow, except it places the story on the ignore list.
REMOVE	If a story on one of the lists is highlighted, *remove* will remove that story from the list.
VPAGER	If a story on the follow list is highlighted, *vpager* will toggle the voice pager flag (V) on or off. In addition to acting as a normal follow story, when the system matches a story to a voice pager flagged story, it will automatically forward the audio to the user's voice pager and also play it on the user's desktop computer.
SAVE	Save the follow and ignore lists.
EXIT	Quit the browser.

- PRIORITY	Decrease the priority of a story in the follow list. The highest story has the highest priority.
+ PRIORITY	Increase the priority of a story on the follow list.
INTRO SCAN	Begins playing the first few seconds of each story within the current broadcast.
PLAY	Begins audio playback at the current time indicated by the audio widget.
STOP	Stops audio playback.
HELP	Display help window.

The *xnews* interface allows browsing of multiple hourly broadcasts as well as the user's most recent synthetic broadcast generated by the system. It also lets the user select stories which it will use for future filtering and newscast construction decisions.

5.2.1 SoundViewer Widget

The audio of the current broadcast is the main focus of the application window, as shown in the *SoundViewer* widget of Figure 5.2 [25].

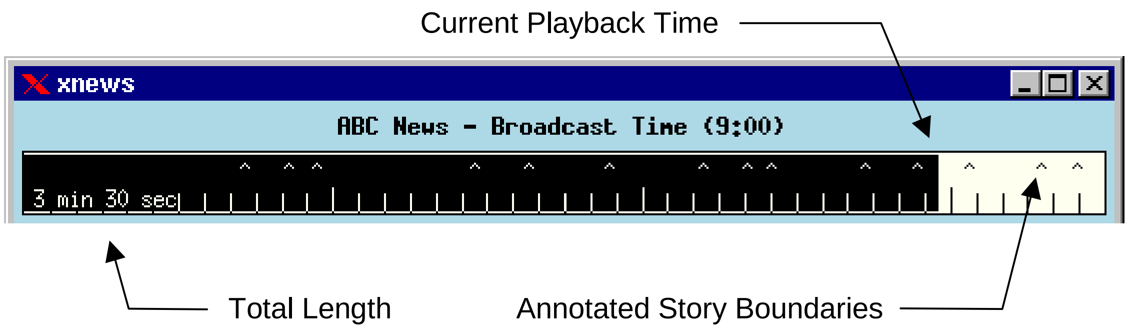


Figure 5.2: Detail of the SoundViewer widget.

This widget controls playback of the audio and has many built-in features. Via mouse control (clicking and dragging in the widget window itself) the user can start and stop playback, and jump instantaneously to any point in the audio file. The widget also implements a version of the "synchronous overlay and add" (SOLA) audio time compression algorithm [26], allowing playback at higher speeds than normal without changing the pitch, but gradually degrading the comprehensibility.

Controlling audio playback through the SoundViewer widget was not very intuitive at first to many users, so the "play" and "stop" buttons were added as alternative methods. Users also expressed a desire to have a visual representation of the length of each story, so I added widget annotations that mark the locations of story boundaries, using carat marks to indicate the positions of the story boundaries within each newscast. The widget reads the boundaries from the *audio auxiliary file*, the file that stores the audio meta-structure. Table 5.2 lists the library functions that were added to support story navigation through the SoundViewer widget.

Table 5.2: Library functions to support story navigation.

DB ns_initdb(char *sound_filename)	Initializes the story database (in memory).
int ns_getNumStories(DB ns_db)	Returns the total number of stories in the database.
int ns_getCurrentStoryFromTime(DB ns_db, long current_time)	Returns the index number of the current story based on the current time.
int ns_getCurrentStory(DB_REC story)	Returns the story index number of a database record.
DB_REC ns_setCurrentStory(DB ns_db, int story)	Returns the database record for the indicated story number.
long ns_getStoryStartTime(DB_REC story)	Returns the start time (in milliseconds) of the indicated story record.
long ns_getStoryStopTime(DB_REC story)	Returns the stop time (in milliseconds) of the indicated story record.
long ns_getStoryLength(DB_REC story)	Returns the total length (in milliseconds) of the indicated story record.
char *ns_getStoryTranscription(DB_REC story)	Returns the automatic speech recognition transcript for the indicated story record.

5.2.2 Story Navigation

The SoundViewer widget can quickly move playback to the annotated story boundaries by using the +/- story buttons. Along with the relative time position in the broadcast shown by the widget, there are two other indicators that describe the current story. First, a field displays the current story index number along with the total number of stories for the current broadcast. Second, a portion of the transcript from the automatic speech recognition (ASR) engine for that particular story is displayed. Figure 5.3 shows this portion of the interface.

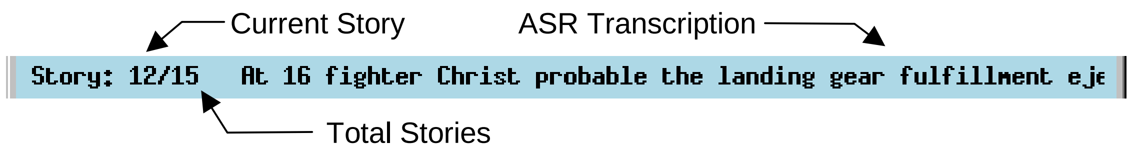
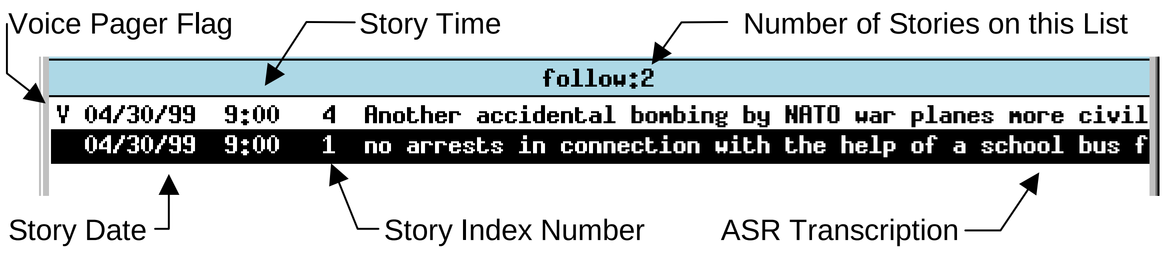

Current Story
ASR Transcription
Story: 12/15 At 16 fighter Christ probable the landing gear fulfillment eje
Total Stories

Figure 5.3: Description of the current story as displayed in the *xnews* interface.

Another technique, the *intro scan*, performs similar to the familiar radio function that scans multiple radio stations. It starts with the first story of the broadcast and proceeds to play a few seconds from the beginning of each story, automatically advancing from one story to the next until the user stops the process by pushing the intro scan button again, pushing the stop button, or by using one of the story navigation buttons. A configuration file sets the amount of time the intro scan plays for each story. Some user thought the intro scan was a useful tool for quickly browsing and/or selecting stories; others thought it was easier to simply use the navigation buttons to move from story to story, arguing that the intro scan did not offer enough information to accurately determine the topic of every story.

5.2.3 Selecting Query Stories

While listening to a broadcast, the user can tell the system to either *follow* or *ignore* the current story in the future. This adds a description of the story to the appropriate list in the application window, and when *save* is selected, it saves the current lists in a user query file. The visual depiction of the story includes the date and time of the broadcast, the index number of the story, and a text description, illustrated in Figure 5.4.



```
                              follow:2
V 04/30/99  9:00   4  Another accidental bombing by NATO war planes more civil
  04/30/99  9:00   1  no arrests in connection with the help of a school bus f
```

Figure 5.4: Close-up of the *xnews* follow list.

When the user decides to ignore the current story, the application adds the story to the ignore list, and also automatically advances the audio to the beginning of the next story. A story in the follow list may also have a special "V" flag that identifies the story as a particularly important one, and tells the system to attempt immediate delivery of future matching stories. Chapter 6 discusses news delivery.

Follow stories also have a priority associated with them, indicated by their position on the list. The most important one is at the top and the least important at the bottom. The relative priority will affect the story presentation sequence in future synthetic newscasts, with more important stories presented earlier in the newscast. Two interface buttons allow the user to increase or decrease the priority of a story once it has been added to the list.

The text descriptions in the lists come from the automatic speech recognition engine transcription. This helps a user quickly identify which stories have been selected. These transcriptions are not very accurate, and can possibly be very misleading, especially if the story boundary was not defined precisely. The transcription text at the beginning of a story may actually be from the previous story. To help alleviate this problem, the system uses the transcription text that correlates with the time in the story that the user actually made the selection. It assumes a linear relationship between number of words and time length of the story. For example, if the user pressed *follow* half way through the audio of a story, then the application would display text from the middle of the transcript for that story. This presents the information most salient to the user's thoughts at the time of the selection.

User studies showed that listeners were generally able to use the transcription segments in the list to help them remember which audio stories they selected during a single session; however, if they reviewed their selections the next day, they had a much more difficult time associating the text segment with news topics. Some wanted the ability to review the entire ASR transcript for a story, while others wanted to see only a few keywords that accurately summarized the topic.

5.3 TELEPHONE BASED INTERFACE

A touch-tone driven phone application, built as part of the Phoneshell [27] system, gives users a more flexible method of listening to the news. A telephone interface significantly increases the accessibility of audio news, especially considering the growing popularity of cellular phones. It enables access to audio news in environments far from the standard desktop computer, including ones where the user is mobile, such as walking, riding in a car, or travelling. The application gives feedback through a Dectalk text-to-speech synthesis (TTS) module and pre-recorded audio cues. Figure 5.5 shows a layout of the telephone interface, and Table 5.3 briefly describes each function.

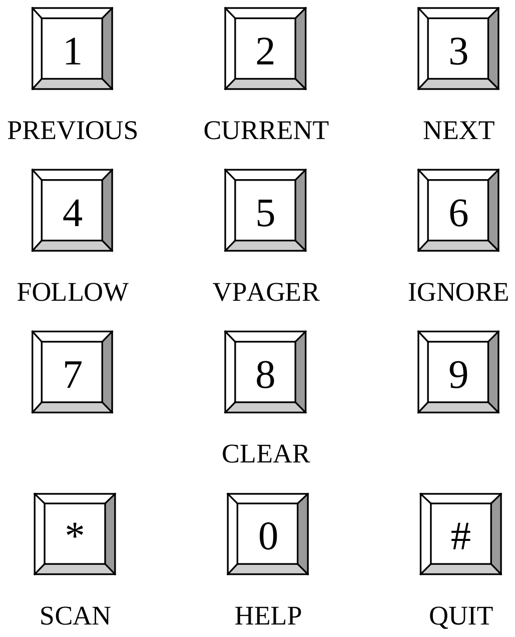


Figure 5.5: Command to key mapping for the telephone interface.

Table 5.3: Summary of commands available through the telephone interface.

Command	Definition
PREVIOUS	Moves audio playback to the beginning of the previous story.
CURRENT	Moves audio playback to the beginning of the current story.
NEXT	Moves audio playback to the beginning of the next story.
FOLLOW	Adds the current story to the user's follow list.
VPAGER	Adds the current story to the user's voice pager follow list. Lists on the voice pager follow list are automatically included on the normal follow list as well.
IGNORE	Adds the current story to the user's ignore list.
CLEAR	Clears the user's follow and ignore lists.
SCAN	Intro scan.
HELP	Gives a list of the available commands.
QUIT	Quits the application.

5.3.1 Story Navigation

Story navigation works similar to the graphical interface, but without the associated visual feedback that indicates the story number and relative position within the entire broadcast. The system usually does not give any explicit audio feedback as the user switches from story to story, because the change in news audio already indicates that the navigation operation was successful. However, it does notify the user if navigation beyond the first or last story is attempted, saying "first story" if the user hits *previous* while on the first story, or "last story" if the user hits *next* during the last story in the broadcast. This interface also supports the *intro scan* function described in Section 5.2.2.

5.3.2 Selecting Query Stories

Although the phone interface is meant primarily as a browsing tool for customized newscasts, users can also make additions to their query file. The Phoneshell program uses the same selection technique as the *xnews* application, allowing users to identify stories to follow or ignore as they listen to the audio. Instead of flagging a follow story (with a "V" in *xnews*) as a special voice pager story, the interface uses a separate selection key to distinguish between the two types. Currently the interface does not support story priority definition as the *xnews* application does. It simply appends any stories you select to the end of the query file, in essence assigning priority by the order of selection. Contrary to story navigation, picking a query story does require an explicit confirmation to let the user know the operation was a success. Two different types of feedback are discussed next.

5.3.3 Audio Confirmations

In many cases where the Phoneshell application must provide explicit feedback about the result of a user action, users can customize the audio confirmations they receive through a configuration file. User evaluations of the telephone interface have shown a wide variety of feedback mechanisms requested by users for different reasons. In the first type of response, the Dectalk module synthesizes text such as "following story" into speech. This responds with a very descriptive remark, but people sometimes have difficulty understanding the computer-generated speech unless they are customarily exposed to it. Users can easily customize this response via a text configuration file; however, it does not sound very natural.

The second method uses pre-recorded audio, which can be spoken phrases or audio *icons* [28]. Spoken phrases eliminate the comprehension problem associated with TTS, however they take a long time to utter and can interrupt the user's train of thought when listening to the news stories. An audio icon, or cue, conveys the same meaning in a much shorter time by using short, non-speech audio segments such as a cartoon "beep." The association between an audio cue is usually intuitively obvious, so the user will not forget it over time. This could be a soft "beep" for *follow* and a harsh alarm sound for *ignore*. Most users liked the short audio cues the best. Interestingly, however, one user thought any sort of audio response whatsoever was enough to interrupt the train of thought, so she opted for no feedback to story selections over the phone. Through the configuration file, users can select a default set of audio cues or TTS responses; or, they can create their own individual responses for each operation.

5.4 User Query File

The interfaces above described ways to indicate interest and disinterest in particular stories heard by the users. The applications save the selections in a user query file, which is unique to each user of the system. Stories selected one day automatically remain in the file in the future until they are explicitly deleted. This works well for interest in longer term stories that may continue for days or even weeks. The system will continue to search for matches to the descriptions it has in the query file.

6. Presentation and Delivery

This chapter describes the method of generating synthetic news broadcasts and presenting them to the user. It supports two main usage scenarios. In the first scenario, users listen to an entire synthetic newscast, which has been custom generated according to each user's interests, much in the same way they would listen to the evening news at the end of the day. However, along with news customization they can also use the advanced navigation and browsing features supported by the structured audio in the synthetic newscast. In the second situation, the system finds a match to an important story that has been marked for special delivery, and then attempts to immediately present this story to the user. This ensures that users receive news about specific stories in a timely fashion. The system supports usage of either of these scenarios independently, or in conjunction with one another.

6.1 Synthetic Newscast

A synthetic newscast is a custom compilation of stories drawn from recent news broadcasts. The selections made by the user described in chapter 5 affect which stories the system selects for the synthetic newscast, as well as the presentation sequence of those stories. In one possible usage scenario, a user listens to one or more early broadcasts in the morning, giving the system feedback about topic preferences, and then receives a full synthetic newscast near the end of the day, tailored to the preferences defined that morning, or even to preferences defined in previous days.

6.1.1 Construction

This section describes the actual construction of a synthetic newscast. Along with providing the query file, the user also tells the system which radio broadcasts to use when creating a newscast. The system can filter several broadcasts, choosing the best stories for the synthetic one. Figure 6.1 illustrates the sample configuration of a newscast generated from two evening radio broadcasts, 4:00pm and 5:00pm.

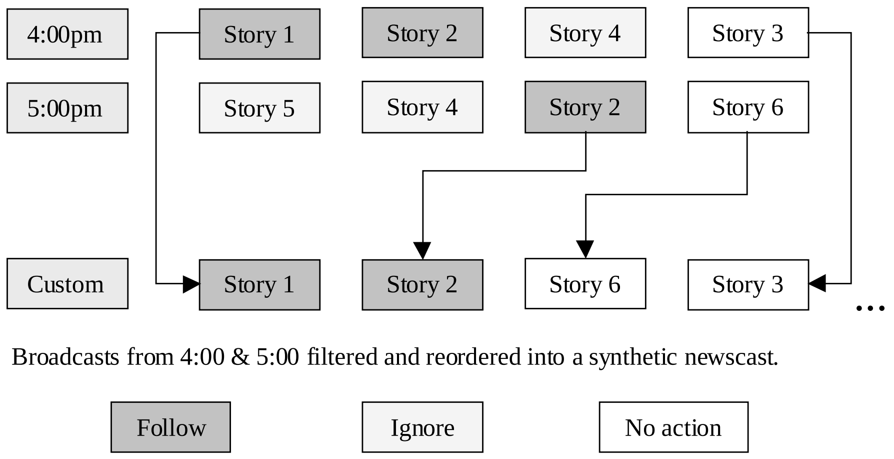


Figure 6.1: Construction of a synthetic newscast.

In this example stories 1, 2, 6, and 3 matched ones listed in the user query file. The system chose the more recent version of story 2 (the one from the 5:00pm broadcast) for the custom newscast. Construction follows this procedure:

- Starting with the highest priority follow story (the first one listed in the user query file), search for a match for each story. If matches are found in multiple broadcasts, then select the most recent one.
- Search for stories that match entries in the ignore list, and remove each of those from consideration.
- Place the remaining unmatched stories at the end of the synthetic broadcast, also adding the most recent ones first.
- Place the final "This is ABC news" blurb at the end.

Typically the system will automatically generate custom newscasts one to three times per day.

User evaluations of the story selection process showed that users were sometimes confused about how to follow or ignore stories with inaccurate boundary definitions. This happened mostly when long stories were split into two or more segments, but also occurred when more than one story was combined into a single segment. For the first case, selecting all story segments about a topic of interest generally produced the best matching results in the synthetic newscast. Matching for the second case was somewhat unpredictable due to the combined information about more than one story.

6.1.2 Example

Table 6.1 shows detailed results from the construction of a synthetic newscast.

Table 6.1: Detailed analysis of a synthetic newscast construction.

Time	Story Num	Topic	Start Time (seconds)	Cluster	Cosine	Magnitude	Synthetic Story
FOLLOW	1	Kosovo		4	0.77		
FOLLOW	2	Wall Street		12	0.55		
FOLLOW	3	Wyoming murder		6	0.9		
IGNORE	1	Toyota Camry		15	0.69		
4:00pm	1	Kosovo	0.0	4	0.81	0.29	
	2	Kosovo	106.6	17	0.48	0.06	8
	3	Kosovo	111.0	4	0.59	0.05	9
	4	737 crash	120.0	12	0.72	0.16	10
	5	Wyoming murder	156.4	6	0.96	0.12	
	6	Susan McDougal	172.0	15	0.85	0.08	11
	7	Susan McDougal	179.7	8	0.64	0.02	12
	8	Jesse Jackson	184.2	15	0.82	0.08	13
	9	Wall Street/Kosovo recap	189.7	4	0.57	0.08	
5:00pm	1	Kosovo	0.0	4	0.71	0.23	1
	2	Kosovo	84.8	15	0.83	0.09	4
	3	Kosovo	90.6	4	0.54	0.10	5
	4	blood clots in legs	120.0	12	0.87	0.10	6
	5	Wyoming murder/Anthrax	135.8	6	0.77	0.16	3
	6	Anthrax vaccination	154.1	6	0.52	0.07	7
	7	Wall Street/Kosovo recap	180.8	4	0.6	0.08	2

In this example, three stories were selected as follow stories and one as an ignore story, with their relative priorities indicated in the "Story Num" column. The system then assembled a synthetic newscast from the 4:00pm and 5:00pm broadcasts on March 24, 1999. Several segmentation errors are shown, including the separation of the main "Kosovo" story in each broadcast into one large and two smaller stories. Some other stories combine across story boundaries, i.e. "Wyoming murder / Anthrax vaccination." The "Synthetic Story" column shows the reordering of the two broadcasts into a synthetic newscast. The three follow stories were all successfully found in the 5:00 broadcast by matching stories to clusters four and six, and by matching keywords for the Wall Street story. The first Kosovo story from the 4:00pm broadcast was removed because it also had a high correlation with cluster four, however the system was unable to remove the smaller Kosovo segments because they did not have enough semantic information to correlate well with the Kosovo story cluster. This same problem caused the original story segmentation error. The system placed the remaining stories at the end of the newscast. These stories did not correlate highly with the clusters assigned to the queries, or the queries themselves did not correlate highly with a cluster.

6.1.3 Listening to a Synthetic Newscast

Once the system has generated a synthetic newscast, the user can listen to and browse/navigate the newscast with either the graphical *xnews* application or the Phoneshell interface. The system does not give the user any explicit notification about the completion of this process because it occurs at a user-defined time. Usability studies showed that listeners had a difficult time evaluating how well the system matched stories to the queries they had selected - an important feedback mechanism that people can use to determine the capabilities and limitations of the system. Listening to the broadcast was not generally sufficient, as the users did not always remember what stories they had selected, nor could they determine direct relationships between their queries and the stories in the synthetic newscast. When using the graphical desktop interface, most people wanted a visual representation of how the system created the newscast. People requested that the application display a few of the most discriminating words used to make the match for the current story, perhaps highlighting the parts of the query transcript that the system deemed most relevant to the matched story. When listening to the newscast over the phone, presentation of this ancillary information was not deemed as important due to the already restricted communicative capabilities of the medium.

6.2 Timely Stories

Timely stories are ones that match queries designated by the user for quick, or timely, delivery. The system should attempt to present audio news to the user taking into account the user's current location and access to communication infrastructure, and then adapting its method of delivery based on any contextual information it may have. *Synthetic News Radio* supports timely playback of audio on a user's desktop computer, and also has the capability to send an audio news story to a portable communication device called a voice pager. It can determine limited contextual information about the user's location through the Unix "finger" utility.

6.2.1 Matching Important Stories

New ABC radio broadcasts arrive once every hour, and the system searches each of these broadcasts for matches to important query stories. It cannot accurately determine if the latest version of the story contains new information, or if it has changed significantly since the last one received, so it always forwards matches from new broadcasts.

6.3 Immediate Delivery

For more timely delivery, the system can forward news as received by extracting and presenting only those stories of special interest in each hourly broadcast. A news daemon running on the desktop computer monitors whether a user is at the computer or away. It can then begin playing news as received if the user is active, or wait until the user returns to play any new news.

Another asynchronous method forwards stories to a Motorola Tenor Voice Pager when the user is away. A voice pager is a portable electronic device that receives 60 second voice clips as pages. Figure 6.2 has a picture of this device.

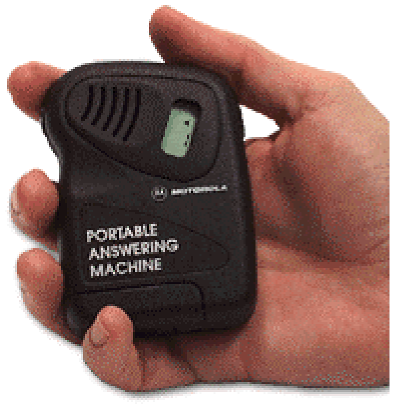


Figure 6.2: The Motorola Tenor voice pager.

A script uses a telephone interface utility to dial up the voice pager answering machine and leave the news clip as a message. Usage of the voice pager reveals that it has a pretty bad interface, especially the alerting mechanisms. It offers two options - an audible alert, and a vibration mode, both of which are fairly loud. If the user does not listen to a message after the initial alert, then the device continues the alerts until the message is accessed. This can become extremely annoying and forces the user to respond to the device's interruptions. A more intelligent and adaptive design will greatly improve the usability of this pager.

7. Conclusions and Future Work

This chapter presents concluding remarks about the design and implementation of the thesis components, and then suggests areas for further research and development.

7.1 Conclusions

This thesis developed a system that automatically finds story boundaries in radio news broadcasts, and then creates an annotation file useful for browsing and filtering. Automatically generating structure for news audio enabled the creation of new interfaces that support non-linear navigation, and generating semantic story representations created a mechanism for assembling a synthetic newscast from audio-based queries. Some research contributions include:

- Using automatic speech recognition with clustered text news stories to find story boundaries in radio news broadcasts.
- Interfaces that allow audio-based queries.
- An application that filters and rearranges news stories to create a synthetic newscast according to personalized interests.

The intonational phrase classifier performed reasonably well, and correctly identified most of the story boundaries among its possibilities. When it missed a boundary, it still found a natural sounding break in the speech stream. The Newscall clustering algorithm identified groups of stories similar to those defined by a real person; however, it sometimes spread the topic definition by combining too many topics into a single group.

User interfaces demonstrated effective use of the annotated story boundaries as a browsing and story navigation aid. Users were initially confused about how to select stories with inaccurate boundary definitions for proper matching in future broadcasts. In this case, users can learn the capabilities and limitations of the system through extended use. The audio query selections were successfully applied to future radio broadcasts to create synthetic newscasts; however, users requested more feedback as to exactly how the system matched the selected stories to the audio based queries. An additional feedback mechanism, such as a display of the most discriminating semantic features for a match, would help users better understand the operation and appropriate use of the system.

7.2 Future Work

This section describes some enhancements and additional areas into which the *Synthetic News Radio* system can expand.

7.2.1 Feedback for Story Matches

The graphical interface should provide more information about exactly how it determined a match to a query. The system should find a way to extract the semantic information that best served as an indicator of a story match made by the story ASR transcriptions through the Latent Semantic Index.

7.2.2 Machine Learning through Relevance Feedback

The system could improve its performance by applying machine learning techniques to relevance feedback collected from the user about how well it made story matches. For instance, the user could optionally rate the success of a match as a percentage from 0% to 100%, and the system could use this information to adapt its matching algorithm and/or story representation to provide better matches to this topic in the future. The system could also allow users to provide additional semantic information about particular stories, or correct misidentified story boundaries.

7.2.3 General Topic Profiling

Compiling representative clusters for general news topic areas, such as weather, sports, etc., could help provide a better understanding of global user interests. Right now the system only has an understanding of short term story interests for each user. Through interaction it could also begin to learn more about the general interests of each specific user, and then take these into account in the generation of a synthetic newscast.

7.2.4 Better Understanding of Attentive State

Timely delivery of important stories could be much improved through a better understanding of the user's attentive state at any give time. Researchers are beginning to create systems that can prioritize incoming information as it relates to the user's current activity, immediate physical environment, and even their emotions or affective state. News delivery could be incorporated into one of these systems in order to present the most relevant news at the most appropriate time.

8. Bibliography

[1] C. Horner, "NewsTime: a graphical user interface to audio news," SM Thesis: Massachusetts Institute of Technology, 1991.

[2] J. S. Wachman, "A video browser that learns by example," SM Thesis: Massachusetts Institute of Technology, 1996.

[3] A. Hauptmann, M. Witbrock, "Informedia: News-on-Demand Multimedia Information Acquisition and Retrieval," In *Intelligent Multimedia Information Retrieval*," M. T. Maybury, Ed., AAAI Press, 1997, pp. 213-239.

[4] D. K. Roy and C. Schmandt, "NewsComm: a hand-held interface for interactive access to structured audio," In *Proc. ACM CHI '96*, 1996, pp. 173-180.

[5] S. Whittaker, J. Hirschberg, J. Choi, D. Hindle, F. Pereira, and A. Singhal, “SCAN: Designing and evaluating user interfaces to support retrieval from speech archives,” In *Proc. ACM SIGIR ’99*, 1999.

[6] J. Hirschberg and C. Nakatani, "Using machine learning to identify intonational segments," In *Proc. AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*, 1998. http://www.aaai.org/Press/Reports/Symposia/Spring/SS-98-01.html

[7] L. Stifelman, "Augmenting real-world objects: a paper-based audio notebook," In *Proc. ACM CHI '96*, 1996.

[8] W. Chafe, "Intonation Units," in *Discourse, Consciousness, and Time*. Chicago: The University of Chicago Press, 1994, pp. 53-71.

[9] J. Pierrehumbert and J. Hirschberg, "The Meaning of Intonational Contours in the Interpretation of Discourse," in *Intentions in Communication*, P. R. Cohen, J. Morgan, and M. E. Pollack, Eds., The MIT Press, 1990, pp. 271-311.

[10] B. Arons, "Interactively skimming recorded speech," PhD thesis: Massachusetts Institute of Technology, 1994.

[11] J. Gruber, "A comparison of measured and calculated speech temporal parameters relevant to speech activity detection," *IEEE Transactions on Communications*, vol. COM-30, no. 4, Apr., pp. 728-738.

[12] B. Secrest and G. Doddington, "An integrated pitch tracking algorithm for speech systems," In *Proc. IEEE ICASSP '93*, 1993.

[13] M. E. Beckman and G. A. Elam, "Guidelines for ToBI Labelling," [Online document], Mar. 1997, [cited 1999 Apr 30], Available HTTP: http://www.ling.ohio-state.edu/phonetics/E_ToBI/

[14] M. Riley, "Some applications of tree-based modeling to speech and language," In *Proc. IEEE ICASSP '89*, 1989, pp. 627-630.

[15] G. Salton, Ed., *The SMART Retrieval System - Experiments in Automatic Document Retrieval*. Englewood Cliffs, NJ: Prentice Hall, Inc., 1971.

[16] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, no. 5, 1988, pp. 513-523.

[17] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell, "Learning to classify text from labeled and unlabeled documents," In *Proc. AAAI '98*, 1998, pp. 792-799.

[18] T. K. Landauer, D. Laham, B. Rehder, and M. E. Schreiner, "How well can passage meaning be derived without using word order? A comparison of latent semantic analysis and humans," In *Proc. of the 19th annual meeting of the Cognitive Science Society*, M. G. Shafto and P. Langley, Eds., Mawhwah, NJ: Erlbaum, 1997, pp. 412-417.

[19] T. K. Landauer, D. Laham, and P. W. Foltz, "Learning human-like knowledge by singular value decomposition: a progress report," In *Advances in Neural Information Processing Systems 10*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Eds., Cambridge: MIT Press, 1998, pp. 45-51.

[20] S. Dumais, "Enhancing performance in latent semantic indexing (LSI) retrieval," Technical Report, Bellcore, Sept., 1992.

[21] M. F. Porter, "An algorithm for suffix stripping," *Program-Automated Library and Information Systems*, vol. 14, no. 3, July 1980, pp. 130-137.

[22] D. Harman, "An experimental study of factors important in document ranking," In *Proc. ACM SIGIR*, 1986, pp. 186-193.

[23] M. Berry, "Large scale singular value computations," *International Journal of Supercomputer Applications*, vol. 6, 1992, pp. 13-49.

[24] R. O. Duda, P. E. Hart, "Unsupervised Learning and Clustering," In *Pattern Classification and Scene Analysis*, New York: John Wiley and Sons, 1973, pp. 226-227.

[25] C. Schmandt, *Voice Communication with Computers*. New York: Van Nostrand Rheinhold, 1994, pp. 282-285.

[26] S. Roucos and A. M. Wilgus, "High quality time-scale modifications for speech," In *Proc. ICASSP '85*, 1985, pp. 493-496.

[27] C. Schmandt, "Phoneshell: the telephone as a computer terminal," In *Proc. ACM Multimedia '93*, 1993, pp. 373-382.

[28] W. Gaver, "The sonicfinder: an interface that uses auditory icons," *Journal of Human Computer Interaction*, vol. 4, no. 1, 1989, pp. 67-94.