Speech and Language Technologies for Audio Indexing and Retrieval

JOHN MAKHOUL, FELLOW, IEEE, FRANCIS KUBALA, TIMOTHY LEEK, DABEN LIU, MEMBER, IEEE, LONG NGUYEN, MEMBER, IEEE, RICHARD SCHWARTZ, MEMBER, IEEE, AND AMIT SRIVASTAVA, MEMBER, IEEE

Invited Paper

With the advent of essentially unlimited data storage capabilities and with the proliferation of the use of the Internet, it becomes reasonable to imagine a world in which it would be possible to access any of the stored information at will with a few keystrokes or voice commands. Since much of this data will be in the form of speech from various sources, it becomes important to develop the technologies necessary for indexing and browsing such audio data. This paper describes some of the requisite speech and language technologies that would be required and introduces an effort aimed at integrating these technologies into a system, called Rough'n'Ready, which indexes speech data, creates a structural summarization, and provides tools for browsing the stored data. The technologies highlighted in this paper include speaker-independent continuous speech recognition, speaker segmentation and identification, name spotting, topic classification, story segmentation, and information retrieval. The system automatically segments the continuous audio input stream by speaker, clusters audio segments from the same speaker, identifies speakers known to the system, and transcribes the spoken words. It also segments the input stream into stories, based on their topic content, and locates the names of persons, places, and organizations. These structural features are stored in a database and are used to construct highly selective search queries for retrieving specific content from large audio archives.

Keywords*—*Audio browsing, audio indexing, information extraction, information retrieval, named-entity extraction, name spotting, speaker change detection, speaker clustering, speaker identification, speech recognition, story segmentation, topic classification.

I. INTRODUCTION

In a paper on how much information there is in the world, M. Lesk, director of the Information and Intelligent Systems division of the National Science Foundation, concludes: "So in only a few years, we will be able to save *everything*—no information will have to be thrown out—and the typical piece of information will *never* be looked at by a human being." [1] Much of that information will be in the form of speech from various sources: television, radio, telephone, meetings, presentations, etc. However, because of the difficulty of locating information in large audio archives, speech has not been valued as an archival source. But, after a decade or more of steady advances in speech and language technologies, it is now possible to start building automatic content-based indexing and retrieval tools, which, in time, will make speech recordings as valuable as text has been as an archival resource.

This paper describes a number of speech and language processing technologies that are needed in developing powerful audio indexing systems. A prototype system incorporating these technologies has been built for the indexing and retrieval of broadcast news. The system, dubbed *Rough'n'Ready*, provides a *rough* transcription of the speech that is *ready* for browsing. The technologies incorporated in this system, and described in this paper, include speaker-independent continuous speech recognition, speaker segmentation, speaker clustering, speaker identification, name spotting, topic classification, story segmentation, and information (or story) retrieval. The integration of such diverse technologies allows Rough'n'Ready to produce a high-level structural summarization of the spoken language, which allows for easy browsing of the data.

The system and approach reported in this paper is related to several other multimedia indexing systems under development today. The Informedia system at Carnegie-Mellon University (CMU) [2]–[4] and the Broadcast News Navigator at MITRE Corporation [5], [6], both have the ability to automatically transcribe and time-align the audio signal in broadcast news recordings and to locate proper names in the transcript and retrieve the audio with information retrieval techniques. The focus of both systems, however, is on features of the video stream. These systems demonstrate that

Manuscript received October 20, 1999; revised April 20, 2000. This work was supported in part by DARPA and monitored by the Air Force Rome Laboratory under Contract F30602-97-C-0253.

The authors are with BBN Technologies, Cambridge, MA 02138 USA (e-mail: makhoul@bbn.com; fkubala@bbn.com; tleek@bbn.com; dliu@bbn.com; lnguyen@bbn.com; schwartz@bbn.com; asrivast@bbn.com).

Publisher Item Identifier S 0018-9219(00)08102-0.

0018–9219/00$10.00 © 2000 IEEE

cues from the video are very effective in locating the boundaries between news stories. They also make extensive use of the closed-captioned text that accompanies most television news programming in the United States today.

Another multimedia system is being developed at CMU for indexing and browsing meetings from video [7]. In this domain, no closed-captioning is available, so there is a stronger reliance on the automatic transcription. But the video is also exploited to detect speaker changes and to interpret gestures such as gaze direction and head/hand movement.

The Rough'n'Ready system, in contrast, has focused entirely on the linguistic content contained in the audio signal and, thereby, derives all of its information from the speech signal. This is a conscious choice designed to channel all development effort toward effective extraction, summarization, and display of information from audio. This gives Rough'n'Ready a unique capability when speech is the only knowledge source. Another salient feature of our system is that all of the speech and language technologies employed share a common statistical modeling paradigm that facilitates the integration of various knowledge sources.

Section II presents the Rough'n'Ready system and shows some of its indexing and browsing capabilities. The remainder of the sections focus on the individual speech and language technologies employed in the system. Section III presents the basic statistical modeling paradigm that is used extensively in the various technologies. Section IV describes the speech recognition technology that is used and Section V details the three types of speaker recognition technologies: speaker segmentation, speaker clustering, and speaker identification. The technologies presented in the next sections all take as their input the text produced by the speech recognition component. Sections VI–IX present the following technologies in sequence: name spotting, topic classification, story segmentation, and information retrieval.

II. Indexing and Browsing with Rough'n'Ready

A. Rough'n'Ready System

The architecture of the Rough'n'Ready system [8] is shown in Fig. 1. The overall system is composed of three subsystems: indexer, server, and browser. The indexer subsystem is shown in the figure as a cascade of technologies that takes a single audio waveform as input and produces as output a compact structural summarization encoded as an XML file that is fed to the server. The duration of the input waveform can be from minutes to hours long. The entire indexing process runs in streaming mode in real-time on a dual 733-MHz Pentium III processor. The system accepts continuous input and incrementally produces content index with an output latency of less than 30 s with respect to the input.

The server has two functions: one is to collect and manage the archive and the other is to interact with the browser. The server receives the outputs from the indexer and adds them incrementally to its existing audio archive. For each audio session processed by the indexer, the audio waveform is processed with standard MP3 compression and stored on the server for later playback requests from the client (the browser). The XML file containing the automatically extracted features from the indexer is uploaded into a relational database. Finally, all stories in the audio session are indexed for rapid information retrieval.

The browser is the only part of the Rough'n'Ready system with which the user interacts. Its main task is to send user queries to the server and display the results in a meaningful way. A variety of browsing, searching, and retrieving tools are available for skimming an audio archive and finding information of interest. The browser is designed as a collection of ActionX controls, which make it possible to run either as a standalone application or embedded inside other applications, such as an Internet browser.

B. Indexing and Browsing

If we take a news broadcast and feed the audio into a speaker-independent, continuous speech recognition system, the output would be an undifferentiated sequence of words. Fig. 2 shows the beginning of such an output for an episode of a television news program (ABCs *World News Tonight* from January 31, 1998).[1] Even if this output did not contain any recognition errors, it would be difficult to browse it and know at a glance what this broadcast is about.

Now, compare Fig. 2 to Fig. 3, which is a screen shot of the Rough'n'Ready browser showing some of the results of the audio indexing component of the system when applied to the same broadcast. What was an undifferentiated sequence of words has now been divided into paragraph-like segments whose boundaries correspond to the boundaries between speakers, shown in the leftmost column. These boundaries are extracted automatically by the system. The speaker segments have been identified by gender and clustered over the whole half-hour episode to group together segments from the same speaker under the same label. One speaker, Elizabeth Vargas, has been identified by name using a speaker-specific acoustic model. These features of the audio episode are derived by the system using the speaker segmentation, clustering, and identification components.

The colored words in the middle column in Fig. 3 show the names of people, places, and organizations—all important content words—which were found automatically by the name-spotting component of the system. Even though the transcript contains speech recognition errors, the augmented version shown here is easy to read and the gist of the story is apparent with a minimum of effort.

Shown in the rightmost column of Fig. 3 is a set of topic labels that have been automatically selected by the topic classification component of the system to describe the main themes of the first story in the news broadcast. These topic labels are drawn from a set of over 5500 possible topics known to the system. The topic labels constitute a very high-level summary of the content of the underlying spoken language.

The topic labels shown in Fig. 3 are actually applied by the system to a sliding window of words; then the resulting

[1]The data used in the various experiments reported in this paper are available from the Linguistic Data Consortium, University of Pennsylvania, http://www.ldc.upenn.edu/.

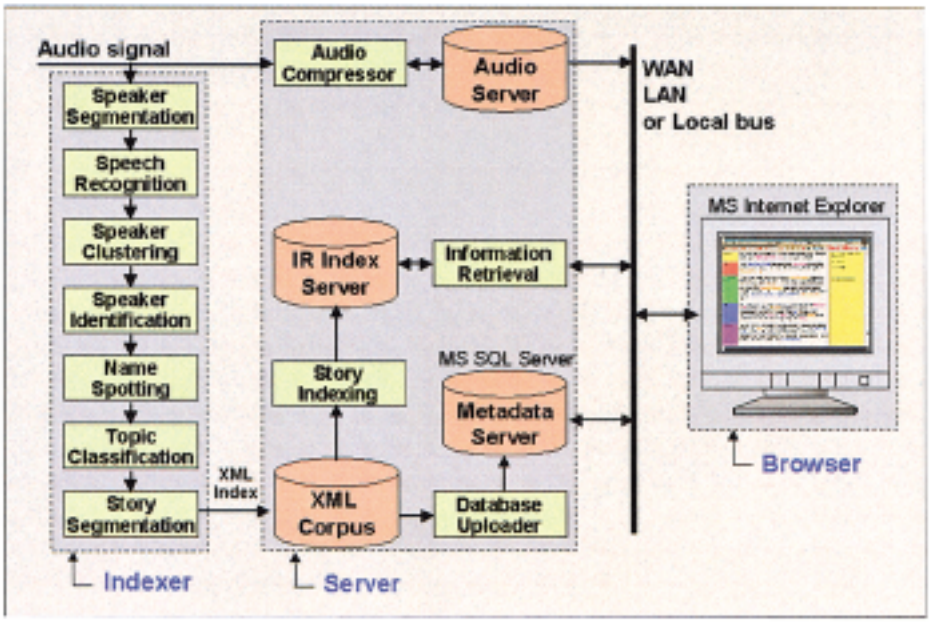

Fig. 1. Distributed architecture of the Rough'n'Ready audio indexing and retrieval system.

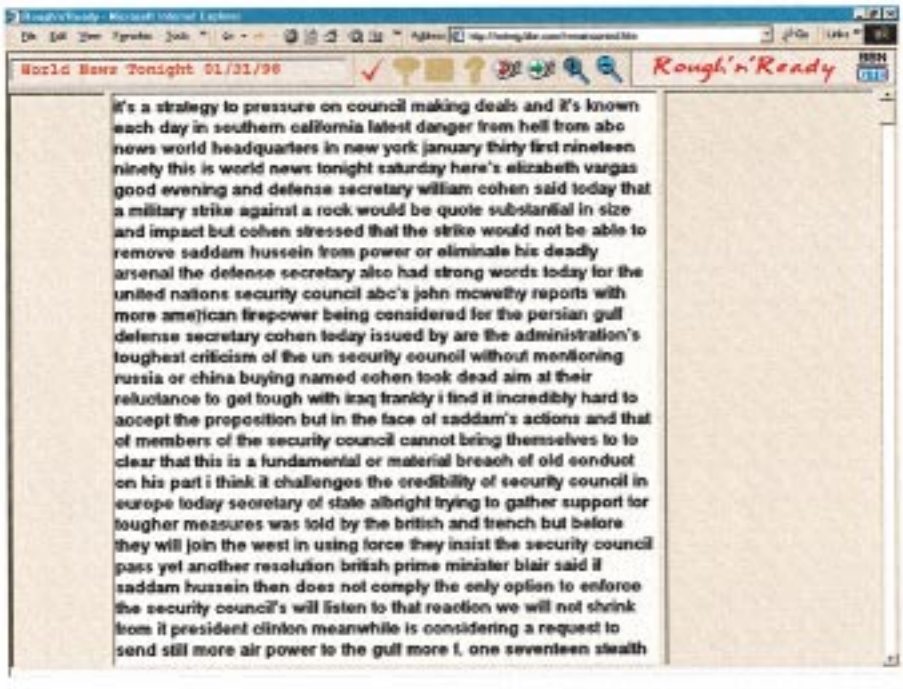

Fig. 2. Transcription of a *World News Tonight* audio broadcast as produced by the BBN Byblos speech recognition system.

sequence of topic labels is used by the story segmentation component of the system to divide the whole news broadcast into a sequence of stories. The result of the story segmentation for this episode is shown in Fig. 4, which is another screen shot of the audio browser.

Breaking a continuous stream of spoken words into a sequence of bounded and labeled stories is a novel and powerful capability that enables Rough'n'Ready to effectively transform a large archive of audio recordings into a collection of document-like units. In the view of the browser shown in Fig. 4, an audio archive consisting of 150 h of broadcast news[1] is organized as a collection of episodes from various content producers. One particular episode (CNN *Headline News* from January 6, 1998) is expanded to show the sequence of stories detected by the system for this particular episode. Each story is represented by a short list of topic labels that were selected by the system to describe the themes of the story. The net effect of this representation is that a human can quickly get the gist of the contents of a news broadcast from a small set of highly descriptive labels.

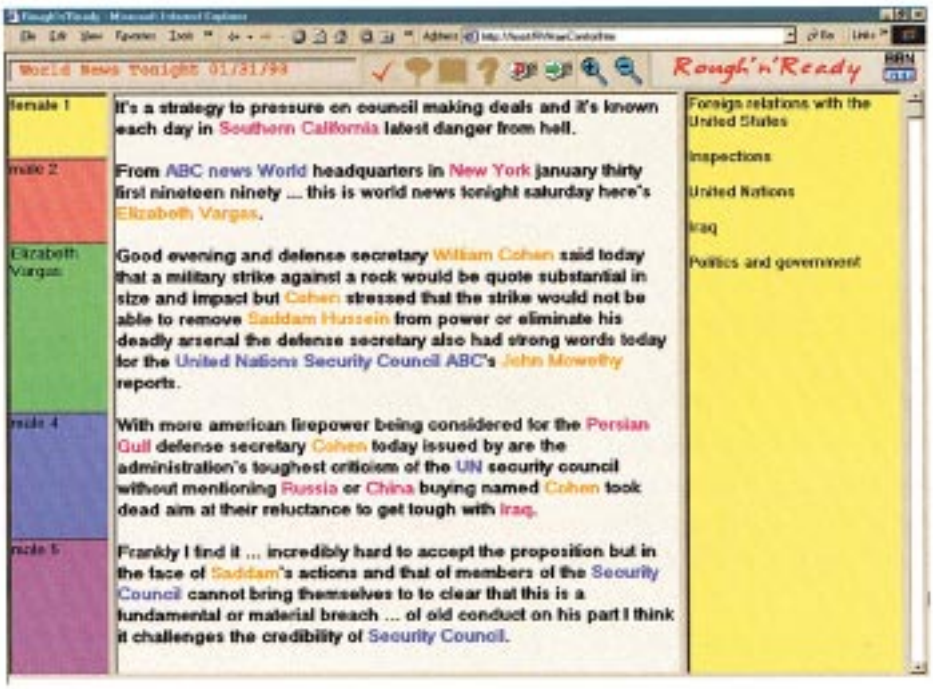

Fig. 3. Elements of the automatic structural summarization produced by Rough'n'Ready from the text that appears in Fig. 2. Speaker segmentation and identification is shown to the left; names of people, places, and organizations are shown in color in the middle section; and topics relevant to the story are shown to the right—all automatically extracted from the news broadcast.

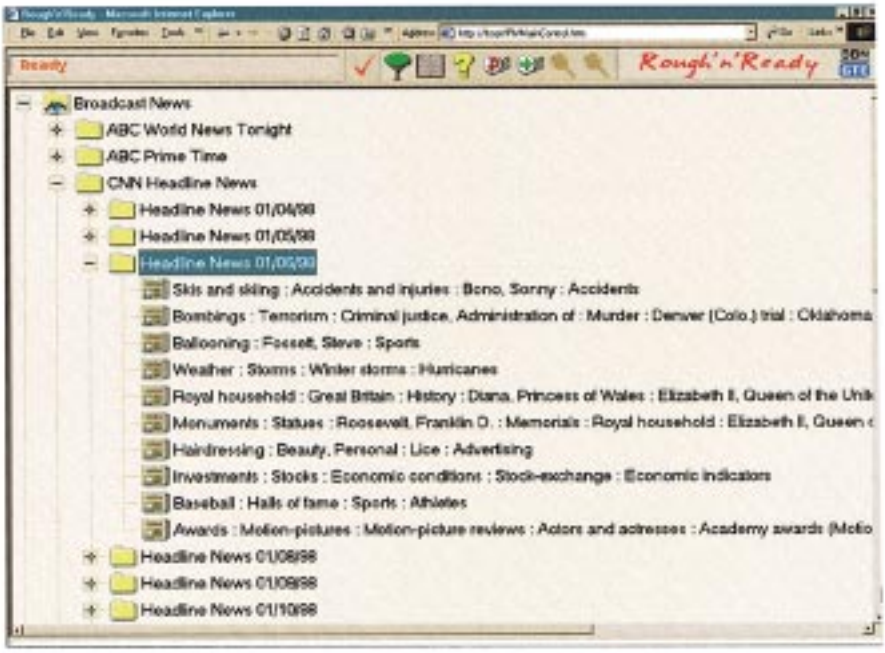

Fig. 4. A high-level organization of an audio archive showing a *Headline News* episode as a sequence of thematic stories, all extracted automatically from the news broadcast.

The first story in the expanded episode in Fig. 4 is about the fatal skiing accident suffered by Sonny Bono. The three important themes for this story—skiing, accidents, and Sonny Bono—have all been automatically identified by the system. Just as important, the system rejected all of the other 5500 topic labels for this story, leaving only the concise list of four topic labels shown here to describe the story. Note that the system had never observed these topics together before in its training set, for Bono died only once. Nonetheless, it was able to select this very informative and parsimonious list of topics from a very large set of possibilities at the same time that it was segmenting the continuous word stream into a sequence of stories.

The entire audio archive of broadcast news is automatically summarized in the same fashion as the expanded episode shown in Fig. 4. This means that the archive can be treated as a collection of textual documents that can be navigated and searched with the same ease that we associate with Internet search and retrieval operations. Every word of the transcript and all of the structural features extracted by the system are associated with a time offset within the episode, which allows the original audio or video segment to be retrieved from the archive on demand. The actual segment to be retrieved can be easily scoped by the user as a story, as one or more speaker segments, or as an arbitrary span of consecutive words in the transcription. This gives the user precise control over the segment to be retrieved.

We now turn to the main topic of this paper, which is a description of the various speech and language technologies employed in the Rough'n'Ready system, preceded by a brief exposition of the general modeling paradigm for these technologies. The descriptions for more recent contributions are provided in more detail than those that had been under development for many years.

III. Statistical Modeling Paradigm

The technologies described in this paper follow the same statistical modeling paradigm shown in Fig. 5. There are two parts to the system: training and recognition. Given some statistical model of the data of interest, the recognition part of the system first analyzes the input data into a sequence of features, or feature vectors, and then performs a search for that output sequence that maximizes the probability of the output sequence, given the sequence of features. In other words, the output is chosen to maximize $P(\text{output}|\text{input, model})$, the probability of the output, given the input and the statistical model. The training program estimates the parameters of the statistical model from a corpus of analyzed training data and the corresponding ground truth (i.e., the desired recognized sequence for that data). The statistical model itself is specified by the technology developer.

Some of the properties of this approach are as follows.

1) A rigorous probabilistic formalism, which allows for the integration of information from different knowledge sources by combining their probabilities.
2) Automatic training algorithms for the estimation of model parameters from a corpus of *annotated* training data (annotation is the process of providing ground truth). Furthermore, the annotation is affordable, requiring only domain knowledge, and can be performed by students or interns.
3) Language-independent training and recognition, requiring only annotated training data from a new language. The training and recognition components generally remain the same across languages.
4) State-of-the-art performance.
5) Robust in the face of degraded input.

We will see below how this paradigm is put to work in the different technologies.

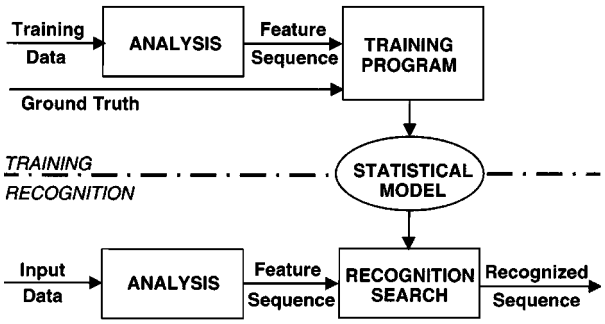

Fig. 5. The statistical modeling paradigm employed in the speech and language technologies presented in this paper.

IV. Speech Recognition

Automatic transcription of broadcast news is a challenging speech recognition problem because of frequent and unpredictable changes that occur in speaker, speaking style, topic, channel, and background conditions. The transcription in Rough'n'Ready is created by the BBN Byblos large-vocabulary speaker-independent speech recognition system [9]. Over the course of several years of participation in the DARPA Broadcast News evaluations, the Byblos system has evolved into a robust state-of-the-art speech recognition system capable of transcribing real-life broadcast news audio data [10].

The Byblos system follows the statistical paradigm in Fig. 5. In the analysis part, the system computes mel-warped cepstral coefficients every 10 ms, resulting in a feature vector of 15 coefficients as a function of time. To deal effectively with the continuous stream of speech in broadcast news, the data are divided into manageable segments that may depend on speaker or channel characteristics (wide-band for the announcer's speech or narrow-band for telephone speech). Segmentation based on speaker, described in the next section, is followed by further segmentation based on detected pauses [11].

The overall statistical model has two parts: acoustic models and language models. The acoustic models, which describe the time-varying evolution of feature vectors for each sound or phoneme, employ continuous-density hidden Markov models (HMMs) [12] to model each of the phonemes in the various phonetic contexts. The context of a phoneme model can extend to as many as two preceding and following phonemes. Weighted mixtures of Gaussian densities—the so-called Gaussian mixture models—are used to model the probability densities of the cepstral feature vectors for each of the HMM states. If desired, the models can be made gender-dependent and channel-specific, and can also be configured to capture within-word and cross-word contexts. To deal specifically with the acoustics of spontaneous speech, which is prevalent in broadcast news, algorithms are developed that accommodate pronunciations typical of spontaneous speech—including those of very short duration—as well as special acoustic models for pause fillers and nonspeech events, such as music, silence/noise, laughter, breath, and lip-smack [13].

The language models used in the system are n-gram language models [14], where the probability of each word is a function of the previous word (for a bigram language model) or the previous two words (for a trigram language model). Higher order models typically result in higher recognition accuracy, but at a slower speed and with larger storage requirements.

To find the best scoring word sequence, the Byblos system employs a multipass recognition search strategy [15], [16] that always starts with an approximate but fast initial forward pass—the *fast-match* pass—which narrows the search space, followed by other passes that use progressively more accurate models that operate on the smaller search space, thus reducing the overall computational cost. For Rough'n'Ready, the system employs two passes after the fast-match pass: the first is a backward pass (from the end of an utterance to the beginning), which generates a list of the top-scoring *N*-best word-sequence hypotheses (*N* is typically anywhere between 100 and 300), and the last pass performs a restoring of the *N*-best sequence, as described below. The final top-scoring word sequence is given as the recognized output.

The fast-match pass, which is performed from the beginning to the end of each utterance, is a time-synchronous search that uses the Single-Phonetic-Tree algorithm [17] with a robust phonetically tied mixture (PTM) acoustic model and an approximate word bigram language model. The output is a word graph with word ending times that are used to guide the next pass. In a PTM acoustic model, all states of the HMMs of all context-dependent models of a phoneme are tied together, sharing a Gaussian mixture density of 256 components; only the mixture weights vary across states. The *N*-best generation pass with a traceback-based algorithm [16] uses a more accurate within-word state-clustered tied-mixture (SCTM) acoustic model and a word trigram language model. Corresponding states of the HMMs of all models of a phoneme are clustered into a number of clusters sharing a mixture density of 64 Gaussian components. A typical SCTM system usually uses about 3000 such clusters. The final pass rescores the *N*-best hypotheses using a cross-word SCTM acoustic model and a word trigram language model and then selects the most likely hypothesis as the recognition output.

Unsupervised adaptation of the Byblos system to each speaker can be performed to improve recognition accuracy. The process requires the detection of speaker-change boundaries. The next section describes the speaker segmentation used in the Rough'n'Ready system to compute those boundaries. The adaptation performed in Byblos is based on the maximum-likelihood linear regression (MLLR) approach developed at the University of Cambridge [18].

In practical applications, such as Rough'n'Ready, it is important that the speech transcription be performed as fast as possible. In addition to the search strategy described above, further speedups have been necessary to bring the computation down to real-time. Major speedup algorithms in the last few years include Fast Gaussian Computation (FGC), Grammar Spreading, and N-Best Tree Rescoring [19].

Since the number of Gaussians associated with each HMM state is very large (typically around 250 000), Gaussian computation is a major bottleneck. Byblos' FGC implementation is a variation of a decision-based FGC developed at IBM [20]. Conceptually, the whole acoustic space can be partitioned through a decision tree into smaller regions such that, for each region, and for any codebook of Gaussians, there is only a short list of Gaussians that can cover that region. During recognition, the decision tree is used to determine the small acoustic region that corresponds to each input feature vector, where only a few Gaussians are used to calculate the likelihood. FGC speeds up the fast-match by a factor of three and the *N*-best generation by a factor of 2.5, with almost no loss in accuracy.

Beam search algorithms can be tuned to run very fast by narrowing the beams. However, aggressive narrow beams can often prematurely prune out correct theories at word boundaries due to the sudden change in likelihood scores caused by the language model score applied at these boundaries. To ameliorate this effect, we have developed an algorithm that "spreads" the language model probabilities across all the phonemes of a word to eliminate these large score spikes [19]. When the decoder is at a word boundary transition, say, from w_1 to w_2, instead of using the bigram probability $P(w_2|w_1)$, we use the probability ratio $P(w_2|w_1)/P(w_2)$. Then we compensate for the division by $P(w_2)$ by multiplying the scores between phone–phone transitions in w_2 by $P(w_2)^{1/k}$, where k is the number of phones in w_2. We call this process "grammar spreading," and we find that it allows us to use a much narrower beam in the backward pass, thus saving a factor of two in computation with no loss in accuracy.

Finally, the *N*-best rescoring pass is also sped up by a factor of two by using a Tree Rescoring algorithm [19] in which all *N* hypotheses are arranged as a tree to be rescored concurrently to eliminate redundant computation.

When we run Byblos on a 450-MHz Pentium II processor at three times real-time (3 $\times$ RT), the word error rate on the DARPA Broadcast News test data, using a 60 000-word vocabulary, is 21.4%. The error rate decreases to 17.5% at 10 $\times$ RT and to 14.8% for the system running at 230 $\times$ RT [10].

V. Speaker Recognition

One of the major advantages of having the actual audio signal available is the potential for recognizing the sequence of speakers. There are three consecutive components to the speaker recognition problem: speaker segmentation, speaker clustering, and speaker identification. Speaker segmentation segregates audio streams based on the speaker; speaker clustering groups together audio segments that are from the same speaker; and speaker identification recognizes those speakers of interest whose voices are known to the system. We describe each of the three components below.

A. Speaker Segmentation

The goal of speaker segmentation is to locate all the boundaries between speakers in the audio signal. This is a

difficult problem in broadcast news because of the presence of background music, noise, and variable channel conditions. Accurate detection of speaker boundaries provides the speech recognizer with input segments that are each from a single speaker, which enables speaker normalization and adaptation techniques to be used effectively on one speaker at a time. Furthermore, speaker change boundaries break the continuous stream of words from the recognizer into paragraph-like units that are often homogeneous in topic.

We have developed a novel two-stage approach to speaker change detection [21]. The first stage detects speech/nonspeech boundaries (note from Fig. 1 that, at this point in the system, speech recognition has not taken place yet), while the second stage performs the actual speaker segmentation within the speech segments. Locating nonspeech frames reliably is important since 80% of the speaker boundaries in broadcast news occur within nonspeech intervals.

To detect speech/nonspeech boundaries, we perform a coarse and very fast gender-independent phoneme recognition pass of the input. We collapse the phoneme inventory into three broad classes (vowels, fricatives, and obstruents), and we include five different models for typical nonspeech phenomena (music, silence/noise, laughter, breath, and lip-smack). Each phone class is modeled with a five-state HMM and mixtures of 64 Gaussian densities. The model parameters are estimated reliably from only 20 h of acoustic data. The resulting recognizer performs the speech/nonspeech detection at each frame of the input reliably over 90% of the time.

The second stage performs the actual speaker segmentation by hypothesizing a speaker change boundary at every phone boundary that was located in the first stage. The time resolution at the phone level permits the algorithm to run very quickly while maintaining the same accuracy as hypothesizing a boundary at every frame. The speaker change decision takes the form of a likelihood ratio test where the null hypothesis is that the adjacent segments are produced from the same underlying distribution. Given two segments $\mathbf{x} = \{x_i, i = 1, \cdots, N\}$ and $\mathbf{y} = \{y_j, j = 1, \cdots, M\}$ with feature vectors x_i and y_j, respectively, we assume that $\mathbf{x}$ and $\mathbf{y}$ were produced by Gaussian processes. Since the means of the two segments are quite sensitive to background effects, we only use the covariances for the generalized likelihood ratio, which takes the form [22]

$$\lambda = \frac{|\Sigma_z|^{(N+M)/2}}{|\Sigma_x|^{N/2}|\Sigma_y|^{M/2}} \tag{1}$$

where $\mathbf{z}$ is the union of $\mathbf{x}$ and $\mathbf{y}$ and Σ is the maximum-likelihood estimate of the covariance matrix for each of the processes. It is usually the case that the more data we have for estimating the Gaussians, the higher λ is [22]. To alleviate this bias, a normalization factor is introduced, so the ratio test changes to

$$\lambda' = \frac{\lambda}{(N+M)^{\theta}} \tag{2}$$

where θ is determined empirically and is usually greater than one. This normalized likelihood ratio is similar to the Bayesian information criterion used in [23]. However, in our case, we can make use of the extra knowledge that a speaker change is more likely to happen during a nonspeech interval in order to enhance our decision making. The final test, therefore, takes the following form.

1) *During nonspeech regions*: if $\log \lambda' \leq t$, then the segments $\mathbf{x}$ and $\mathbf{y}$ are deemed to be from the same speaker, otherwise not, where t is a threshold that is adjusted such that the sum of false acceptance and false rejection errors is a minimum.
2) *During speech regions*: the test changes to $\log \lambda' \leq t + \alpha$, where α is a positive threshold that is adjusted in the same manner as in 1). α is introduced to bias the placement of the speech/nonspeech boundary toward the nonspeech region so that the boundary is less likely to break up words.

We implemented a sequential procedure that increments the speaker segments one phone at a time and hypothesizes speaker changes at each phone boundary using the algorithm given above. The procedure is nearly causal, with a look-ahead of only 2 s, enough to get sufficient data for the detection. The result of this procedure when applied to the DARPA Broadcast News test was to find 72% of the speaker changes within 100 ms of the correct boundaries (about the duration of one phoneme), with a false acceptance rate of 20%. Most of the missed boundaries were brief greetings or interjections such as "good morning" or "thanks," while most of the false acceptances were during nonspeech periods and, therefore, inconsequential.

B. Speaker Clustering

The goal of speaker clustering is to identify all segments from the same speaker in a single broadcast or episode and assign them a unique label; it is a form of unsupervised speaker identification. The problem is difficult in broadcast news because of the extreme variability of the signal and because the true number of speakers can vary so widely (on the order of 1–100). We have found an acceptable solution to this problem using a bottom-up (agglomerative) clustering approach [24], with the total number of clusters produced being controlled by a penalty that is a function of the number of clusters hypothesized.

The feature vectors in each speaker segment are modeled by a single Gaussian. The likelihood ratio test in (1) is used repeatedly to group cluster pairs that are deemed most similar until all segments are grouped into one cluster and a complete cluster tree is generated. At each turn in the procedure, and for each cluster, a new Gaussian model is estimated for that cluster [25]. The speaker clustering problem now reduces to finding that cut of the cluster tree that is optimal based on some criterion. The criterion we choose to minimize is the sum of two terms

$$S = \log \left| \sum_{j=1}^{k} N_j * \Sigma_j \right| + \theta \log k \tag{3}$$

where k is the number of clusters for any particular cut of the tree and N_j is the number of feature vectors in cluster j. The first term in (3) is the logarithm of the determinant of the within-cluster dispersion matrix [24], and the second term is a regularization or penalty term that compensates for the fact that the determinant of the dispersion matrix is a monotonically decreasing function of k. The final clustering is that cut of the cluster tree that minimizes (3). The value of θ is determined empirically to optimize performance; it is usually in the range $0 < \theta < 1$.

This algorithm has proved effective over a very wide range of news broadcasts. It performs well regardless of the true numbers of speakers in the episode, producing clusters of high purity. The cluster purity, which is defined as the percentage of frames that are correctly clustered, was measured to be 95.8%.

C. Speaker Identification

Every speaker cluster created in the speaker clustering stage is identified by gender. A Gaussian mixture model for each gender is estimated from a large sample of training data that has been partitioned by gender. The gender of a speaker segment is then determined by computing the log likelihood ratio between the male and female models. This approach has resulted in a 2.3% error in gender detection.

In addition to gender, the system can identify a specific target speaker if given approximately one minute of speech from the speaker. Again, a Gaussian mixture model is estimated from the training data and is used to identify segments of speech from the target speaker using the approach detailed in [26]. Any number of target models can be constructed and used simultaneously in the system to identify the speakers. To make their labeling decisions, the set of target models compete with a speaker-independent *cohort* model that is estimated from the speech of hundreds of speakers. Each of the target speaker models is adapted from the speaker-independent model. To ameliorate the effects of channel changes for the different speakers, cepstral mean subtraction is performed for each speaker segment whereby the mean of the feature vectors is removed before modeling.

In the DARPA Broadcast News corpus, 20% of the speaker segments are from 20 known speakers. Therefore, the speaker identification problem here is what is known as an *open set* problem in that the data contains both known and unknown speakers and the system has to determine the identity of the known-speaker segments and reject the unknown-speaker segments. Using the above approach, our system resulted in the following three types of errors: a false identification rate of 0.1%, where a known-speaker segment was mistaken to be from another known speaker; a false rejection rate of 3.0%, where a known-speaker segment was classified as unknown; and a false acceptance rate of 0.8%, where an unknown-speaker segment was classified as coming from one of the known speakers.

VI. Name Spotting

The objective of name spotting in Rough'n'Ready is to extract important terms from the speech and collect them in a database. Currently, the system locates names of persons, places, and organizations. Most of the previous work in this area has considered only text sources of written language and has concentrated on the design of rule-driven algorithms to locate the names. Extraction from automatic transcriptions of spoken language is more difficult than written text due to the absence of capitalization, punctuation, and sentence boundaries, as well as the presence of recognition errors. These have significant degrading effects on the performance of rule-driven systems. To overcome these problems, we have developed an HMM-based name extraction system called IdentiFinder [27]. The technique requires only that we provide training text with the type and location of the named entities marked. The system has the additional advantage that it is easily ported to other languages, requiring only a set of annotated training data from a new language.

The name spotting problem is illustrated in Fig. 6. The names of people (Michael Rose, Radovan Karadzic) are in bold; places (Bosnia, Pale, Sarajevo) are underlined; and organizations (U.N.) are in italics. We are required to find all three sets of names but classify all others as general language (GL).

Fig. 7 shows the hidden Markov language model used by IdentiFinder to model the text for each type of named entity. The model consists of one state for each of the three named entities plus one state (GL) for all other words in the text, with transitions from each state to every other state. Associated with each of the states is a bigram statistical model on all words in the vocabulary—a different bigram model is estimated for each of the states. By thinking of this as a generative model that generates all the words in the text, most of the time we are in the GL state emitting general-language words. We then transition to one of the named-entity states if we want to generate a name; we stay inside the state generating the words for that name. Then, we either transition to another named-entity state or, more likely, back to the GL state. The decision to emit each word or to transition to another state depends on the previous word and the previous state. In this way the model uses context to help detect and classify names. For example, the word "Mr." in the GL state is likely to be followed by a transition to the PERSON state. After the person's name is generated, a transition to the GL state is likely and general words like "said" or "departed" may follow. These context-dependent effects are included in our model.

The parameters of the model in Fig. 7 are estimated automatically from annotated training data, where the three sets of named entities are marked in the text. Then, given a test sample, the model is used to estimate the probability of each word's belonging to one of the three named entities or to none. We then use the Viterbi algorithm [28] to find the most likely sequence of states to account for the text. The result is the answer for the sequence of named entities.

The delegation, which included the commander of the *U.N.* troops in Bosnia, Lt. Gen. Sir **Michael Rose**, went to the Serb stronghold of Pale, near Sarajevo, for talks with Bosnian Serb leader **Radovan Karadzic**.

Fig. 6. A sentence demonstrating three types of named entities: people (Michael Rose, Radovan Karadzic), locations (Bosnia, Pale, Sarajevo), and organizations (U.N.).

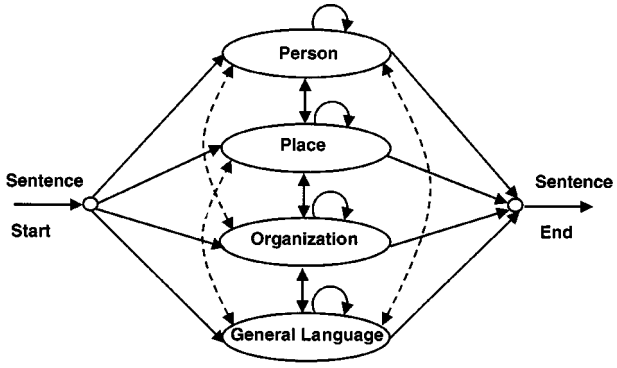

Fig. 7. The hidden Markov model used by IdentiFinder for name finding. Each of the states includes a statistical bigram language model of all the words in the vocabulary.

Since our system has been trained on only 1 million words of annotated data from broadcast news, many of the words in an independent test set will be unknown to the name-spotting system, even though they might be known to the speech recognizer. (Words that are not known to the speech recognizer will be recognized incorrectly as one of the existing words and will, of course, cause performance degradation, as we shall see below.) It is important to deal with the unknown word problem since some of those words will be among the desired named entities and we would like the system to spot them even though they were not seen before by the training component. During training, we divide the training data in half. In each half we replace every string that does not appear in the other half with the string "UNKNOWN." We then are able to estimate all the probabilities involving unknown words. The probabilities for known words are estimated from all of the data. During the testing phase, we replace any string that is unknown to the name spotting system by the label "UNKNOWN" and are then able to find the best matching sequence of states. We have found that by making proper use of context, many of the names that were not known to the name-spotting system are labeled correctly by the system.

One advantage of our approach to information extraction is the ease with which we can learn the statistics for different styles of text. For example, let us say we want the system to work on text without case information (i.e., the text is displayed as either all lower case or all upper case). It is a simple matter to remove the case information from our annotated text and then reestimate the models. If we want to use IdentiFinder on the output of a speech recognizer, we expect that the text will not only be caseless but will also have no punctuation. In addition, there will be no abbreviations, and numeric values will be spelled out (e.g., TWENTY FOUR rather than 24). Again, we can easily simulate this effect on our annotated text in order to learn a model of text output from a speech recognizer. Of course, given annotated data from a new language, it is a simple matter to train the same system to recognize named entities in that language.

We have performed several experiments to measure the performance of IdentiFinder in finding names. In addition, we have measured the degradation when case and punctuation information is lost, or when faced with errors from automatic speech recognition. In measuring the accuracy of the system, both the type of named entity and the span of the corresponding words in the text are taken into consideration. We measure the *slot error rate*—where the type and span of a name is each counted as a separate slot—by dividing the total number of errors in named entities (substitutions, deletions, and insertions) by the total number of true named entities in the reference answers [29].

In a test from the DARPA Broadcast News corpus,[1] where the number of types of named entities was seven (rather than the three used by Rough'n'Ready), IdentiFinder obtained a slot error rate of 11.4% for text with mixed case and punctuation. When all case and punctuation were removed, the slot error rate increased to only 16.5%.

In recent DARPA evaluations on name spotting with speech input, again with seven classes of names, the slot error rate for the output of the Byblos speech recognizer was 26.7% with a speech recognition word error rate of 14.7% [30]. When all recognition errors were corrected, without adding any case or punctuation information, the slot error rate decreased to 14.1%. In general, we have found that the named-entity slot error rate increases linearly with the word error rate in approximately a one-to-one fashion.

VII. Topic Classification

Much work has been done in topic classification, where the models for the different topics are estimated independently, even if multiple topics are assigned to each document. One notable exception is the work of Yang and Chute [31], who, as part of their model, take into consideration the fact that multiple simultaneous topics are usually associated with each document. Our approach to topic classification is similar in spirit to that of Yang and Chute, except that we use a Bayesian framework [32] instead of a distance-based approach. Our topic classification component, called OnTopic, is a probabilistic HMM whose parameters are estimated from training samples of documents with given topic labels, where the topic labels number in the thousands. The model allows each word in the document to contribute different amounts to each of the topics assigned to the document. The output from OnTopic is a rank-ordered list of all possible topics and corresponding scores for any given document.

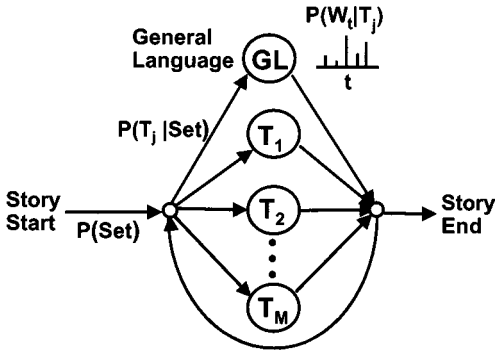

Fig. 8. The hidden Markov model used in OnTopic to model the set of topics in a story. The model is capable of assigning several topics to each story, where the topics can number in the thousands.

A. The Model

We choose the set of topics Set that corresponds to a given document D such that the posterior probability $P(Set|D)$ is maximized

$$P(Set|D) = P(Set)\frac{P(D|Set)}{P(D)}. \tag{4}$$

For the purpose of ranking the sets of topics, $P(D)$ can be ignored. The prior probability $P(Set)$ is really the joint probability of a document having all the labels in the set, which can be approximated using topic co-occurrence probabilities $P(T_k, T_m)$

$$P(Set) \approx \left(\prod_{k \in Set} \prod_{\substack{m \in Set \\ m>k}} P(T_k, T_m) \right)^{1/\binom{N}{2}} \tag{5}$$

where N is the number of topics in Set and the exponent serves to place on similar footing topic sets of different sizes. $P(T_k, T_m)$ is estimated by taking the product of the maximum-likelihood estimates of $P(T_k|T_m)$ and $P(T_m)$. The former is estimated as the fraction of those documents with T_m as a topic that also have T_k as a topic, and the latter is estimated as the fraction of documents with T_m as a topic.

What remains to be computed is $P(D|Set)$, the conditional probability of the words in the document, given that the document is labeled with all the topics in *Set*. We model this probability with an HMM consisting of a state for each of the topics in the set, plus one additional topic state, GL, as shown in Fig. 8. The model "generates" the words in the document one by one, first choosing a topic distribution from which to draw the next word, according to $P(T_j|Set)$, then choosing a word according to $P(W_t|T_j)$, then choosing another topic distribution to draw from, etc. The formula for $P(D|Set)$ is, therefore

$$P(D|Set) \approx \prod_t \sum_{j \in Set} P(T_j|Set)P(W_t|T_j) \tag{6}$$

where t varies over the set of words in the document. The elements of the above equation are estimated from training data as described below.

B. Estimating HMM Parameters

We use a biased form of the Expectation-Maximization (EM) algorithm [33] to find good estimates for the transition probabilities $P(T_j|Set)$ and the emission probabilities $P(W_t|T_j)$ in the HMM in Fig. 8. The transition probabilities are defined by

$$P_{k+1}(T_j|Set) = \frac{E(\#\text{ times any word is emitted in state } T_j|\text{model } k)}{E(\#\text{ times any word is emitted in any state}|\text{model } k)}$$

which can be estimated as

$$P_{k+1}(T_j|Set) = bias(T_j)\frac{\sum_D \sum_{W \in D} q_{k,j}(W, T_j, D)}{\sum_i \sum_D \sum_{W \in D} q_{k,i}(W, T_i, D)} \tag{7}$$

where

$$bias(T_j) = \frac{\sum_{D \text{ with } T_j} l(D)}{\sum_D l(D)} \tag{8}$$

is the bias term, $l(D)$ is the number of words in the document D, and

$$q_{k,j}(W, T_j, D) = c(W|D)I(D \text{ has } T_j)\frac{P_k(T_j|Set)P_k(W|T_j)}{\sum_{i \in Set} P_k(T_i|Set)P_k(W|T_i)}. \tag{9}$$

$q_{k,j}(W, T_j, D)$ is the fraction of the counts for W in D that are accounted for by T_j, given the current set of parameters in the generative model; $c(W|D)$ is the number of times that word W appears in the document; and $I(x)$ is an indicator function returning one if its predicate is true and zero otherwise. The bias term is needed to bias the observations toward the GL state; otherwise, the EM algorithm would result in a zero transition probability to the GL state [31]. The effect of the bias is that the transition and emission probabilities for topic T_j will be set such that this topic accounts for a fraction of the words in the corpus roughly equal to $bias(T_j)$. The emission probabilities are then estimated from

$$P_{k+1}(W|T_j) = \frac{\sum_D q_{k,j}(W, T_j, D)}{\sum_D \sum_{W \in D} q_{k,j}(W, T_j, D)}. \tag{10}$$

C. Classification

To perform classification for a given document, we need to find the set of topics that maximizes (4). But the total number of all possible sets is $\sum_{k=1}^{M} k!$, which is a very large number if the number of possible topics M is in the thousands. Since scoring such a large number of possibilities is prohibitive computationally, we employ a two-pass approach. In the first pass, we select a small set of topics that are likely to be in the

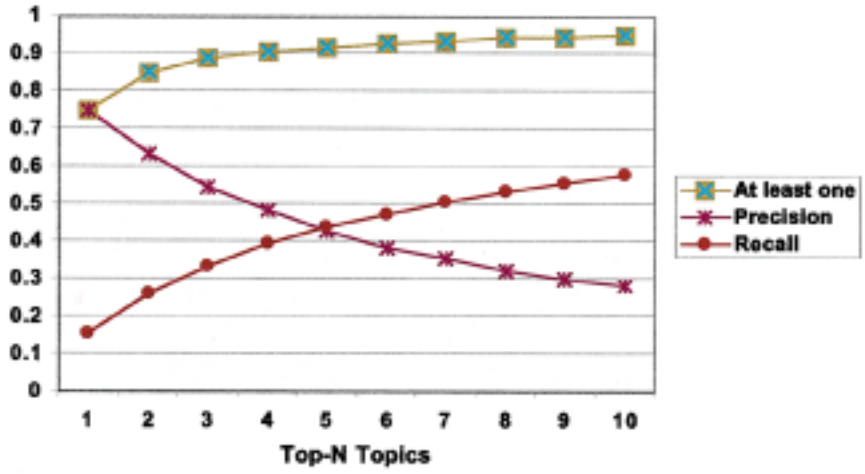

Fig. 9. Performance of OnTopic's classification algorithm on broadcast news when the top-N scoring topics are matched against what the human annotators recorded for each story. The top curve shows the performance when at least one of the N topics matches one of the annotator's topics. The *precision* and *recall* curves score all N topics against all of the annotator's topics.

best set. In the second pass, we score all sets of these candidates using (4). We select candidate topics in the first pass by scoring each topic independently, as if it were a complete set on its own, using a slight modification of (4)

$$\log P(T_j|D) \approx \alpha \log P(T_j) + \sum_t \phi \log\left(P(T_j|Set)^{\beta} \frac{P(W_t|T_j)}{P(W_t)} \right) \quad (11)$$

where $\phi(x)$ is zero if $x < 0$ and x otherwise, and serves to filter out the effect of words in documents that constitute negative evidence for a topic. The parameter α has been introduced to balance the prior against the generative model and is optimized from training data. The parameter β is there to flatten (if less than one) or sharpen (if greater than one) the transition probability distribution, in order to compensate for the independence assumption over words in the document.

D. Experiments

We applied the two-pass procedure of the OnTopic classifier described above to a corpus of broadcast news stories, transcribed and annotated by Primary Source Media. For each story, the annotators gave a number of topic labels that they thought represented the topics in the story. The number of topics for each story was anywhere between one and 13, with an average of 4.5 topics per story. The corpus was divided into one year, or 42 502 stories, for training, and one month, or 989 stories, for test. The training set contained a total of 4627 unique topic labels.

Measuring the performance of our system against what the human annotators wrote down as the topic labels is not straightforward, because our system gives an ordered list of all topics, each with a score, while the annotators have a small, unordered list of topics for each story. Fig. 9 shows different reasonable ways of measuring performance. The abscissa of the figure shows the number N of top-ranking topics provided by the system. For each value of N, we compare the top-N topics produced by the system against the set of topics generated by the annotators. The comparison is done in two ways. The at-least-one-correct curve shows the fraction of stories for which at least one of the top N topic labels for each story was included in the annotations for that story. Clearly, that fraction increases with increasing N. We see, for example, that the top scoring topic was deemed correct 76% of the time. In the second method of comparison, we compare all N top topics generated by the system against the set of annotated topics and count how many are the same, then we measure *precision* and *recall*. *Precision* is the fraction of N topics that the system got correct (i.e., matched the human annotators) and *recall* is the fraction of the topics generated by the annotators that the system got correct. As usual, precision decreases as recall increases.

We have indications that the criteria we have adopted for measuring the performance of our system may be less forgiving than necessary. Topic annotation is not an easy task when the number of topics is large; people tend to undergenerate labels for documents because it is difficult to remember so many topics. Upon informal examination of stories for which the top scoring topic was not included in the list given by the annotators, we often found that the topic given by the computer was quite reasonable for the story.

While it is possible to apply OnTopic to any segment of broadcast news (e.g., for every speaker segment), for the purpose of indexing, it would be even more useful to use topic classification as a means to finding story boundaries. This is the subject of the next section.

VIII. Story Segmentation

Story segmentation turns the continuous stream of spoken words into document-like units with a coherent set of topic labels assigned to each story. In Rough'n'Ready, we apply OnTopic to overlapping data windows of 200-words span, with a step size of four words between successive windows. For each data window, and for each topic of the 5500 topics known to the system, we compute the log probability of the topic given the words in the window. The list of 5500 such topic scores for each data window is pruned automatically to

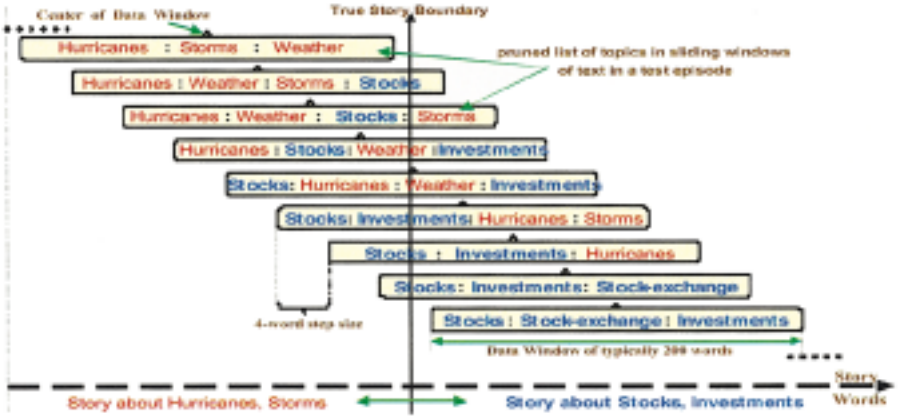

Fig. 10. The story segmentation component first chooses a few top scoring topics for each 200-word data window on a sliding basis every four words. Shown above are the chosen topics as the window passes across two stories, one about *hurricanes* and the other about *stocks*.

preserve only the top scoring (i.e., the most relevant) topics for that data window, as follows. We assume that the scores of the top scoring 100 topics are drawn from a Gaussian process, and we choose as our pruned list those topics that lie above twice the standard deviation from the mean score. The result of this process is depicted in Fig. 10, which shows the results of the topic pruning process during a transition from one story about *hurricanes* to another about *stocks*.

The challenge now is to locate the boundary between stories. We define a *topic window* as the aggregate of 50 consecutive pruned topic lists, and we compute *topic persistence* as the number of occurrences of each topic label found in a topic window. We then measure the *maximum-persistence* score as the largest persistence found for any topic in a given topic window. Fig. 11 shows the maximum-persistence score as a function of topic window across an episode. The maximum value of 50 is typically reached during regions that are within the same story. The vertical dashed lines in Fig. 11 show the true boundaries between different stories. By setting a threshold of 90% of the maximum, as shown by the horizontal line in Fig. 11, we can narrow the search for the story boundaries to the regions below the threshold.

The story boundaries are then located more precisely by taking note of the locations of *topic support words* within the text. Topic support words (or keywords) are those words in a topic window that contribute to the score of one of the surviving topics for the putative story. We observe that only about 6%–8% of the words in a story provide support for any of the topic labels assigned to a story. We also observe that the support words are most often easily separable into two groups whenever they span a true story boundary. One group supports the topics identified in the preceding story and the other supports topics in the succeeding story. We exploit this effect to automatically locate the story boundaries occurring between stable topic regions. We also constrain the boundary decision to prefer a nearby speaker boundary and to avoid splitting names. Further details are provided in [34].

The performance of the story segmentation procedure was tested on a test corpus consisting of 105 episodes with a total of 966 stories. Given a 50-word tolerance, the story segmentation procedure correctly detected 77% of the true boundaries and had a false acceptance of 90%, i.e., for every true boundary, approximately two boundaries were found on the average by the segmentation procedure. Longer stories, in which the topics drift, tended to be subdivided by our procedure and this is why the false acceptance was high. Note that, for indexing and retrieval purposes, such a high false acceptance rate is of little consequence. Fig. 4 shows an example of the results of story segmentation on one episode of broadcast news.

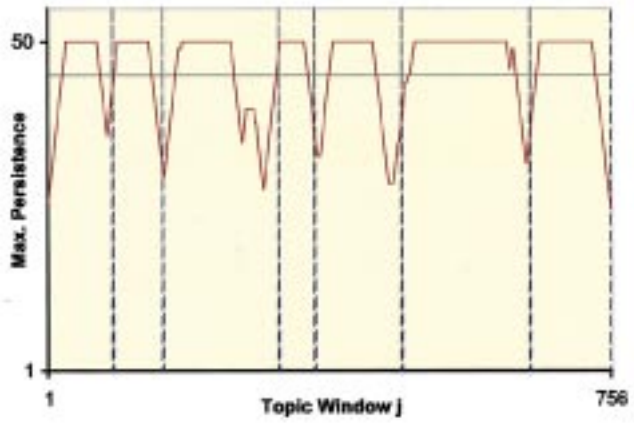

Fig. 11. A plot of persistence as a function of topic window number for a broadcast news episode. The high persistence regions are ones where the set of topics chosen are uniform; the persistence dips across story boundaries. The vertical dashed lines show the true story boundaries.

IX. Information Retrieval

The Rough'n'Ready browser is capable of retrieving stories of interest based on speakers, topics, and/or names of people, places, and organizations. Another capability of the browser is to retrieve stories that are similar to a given story of interest. To perform this task, we employ a novel information retrieval (IR) system, called Golden Retriever [35]. Information indexing and retrieval take place on the Rough'n'Ready server. Whenever a new episode is processed

by the indexer, a new retrieval index is generated over the entire archive of indexed stories. The browser gives the user a powerful query-by-example capability whereby an entire news story is submitted to the Golden Retriever search engine as a query to find all similar stories in a large audio archive. This provides an effective means for a user to find related passages once a single example of interest has been found. This capability makes valuable use of the topic classification and story segmentation capabilities described above.

Golden Retriever is a novel probabilistic HMM-based IR system that computes the probability that a document is relevant, given a query, and ranks all documents in the collection based on this measure. Our approach to IR mirrors our topic classification work; we allow a corpus of examples to drive our selection of models and our estimation procedures. The corpus consists of a set of documents, a set of natural language queries (tens of words), and a number of relevance judgments that state whether each document is relevant to the query or not. Human annotators make the relevance judgements on some significant sampling of the corpus of documents for each query. We build a statistical model capable of ranking training documents effectively by their labeled relevance to given training queries.

A. A Bayesian Model for IR

Given a query, it seems sensible to rank the documents in a corpus by their probability of being relevant [36]. In other words, we want to use as our document ranking function the posterior probability $P(DisR|Q)$, the probability that the document D is relevant, given query Q. We again use Bayes' rule to decompose the posterior probability

$$P(DisR|Q) = \frac{P(DisR)P(Q|DisR)}{P(Q)}. \qquad (12)$$

$P(DisR)$ is the prior probability of a document's being relevant to any query. Currently, we assume that this prior is uniform, although, in principle, we can make the prior a function of document features. $P(Q)$ is simply the prior probability of the query's being posed in the first place. As this quantity does not alter the document ranking, we can safely ignore it. What is left is the conditional probability of the query being posed, under the hypothesis that the document is relevant, $P(Q|DisR)$. We model this remaining quantity with a discrete HMM that is dependent on the document. This will be a generative model where we think of the document HMM as generating the query. The parameters of the HMM should be estimated in such a way as to make it more likely that a document will generate a query to which it is relevant than a query to which it is not relevant.

A simple formulation of the requisite HMM has just two states, as shown in Fig. 12. The state labeled "D" represents the option of generating query words by drawing words directly from the document. The state labeled "GL" represents choosing words from general language, i.e., without regard to the document. Most queries contain words that are present in relevant documents, but all queries contain many general words that are not really part of the specification of relevant documents.

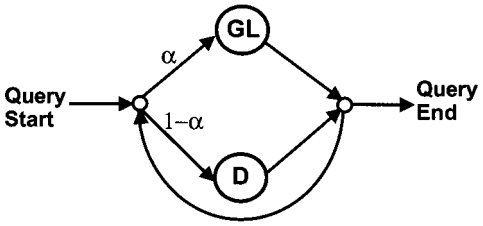

Fig. 12. A hidden Markov model for a single document for the Golden Retriever information retrieval component. This generative model assumes that the query is generated by a model comprising two states, one for the document and another for general language.

B. Training the IR HMM

The parameters of the HMM are the transition probability α and the emission probabilities for each of the words in each state. In principle, we would like to estimate these parameters from examples using the EM algorithm. In practice, however, we find that we do not have enough training examples to find good estimates for the emission probabilities. So we set the emission probabilities for the D and GL states to be the unigram distributions of the words in the document and the whole corpus, respectively. Further, we set the transition probabilities to be the same α for all documents, and we estimate α using the EM algorithm. We found that $\alpha = 0.7$ was the value to which EM converged.

C. Performance

We have tested this simple two-state HMM on the Text Retrieval Conference (TREC-7) corpus, which consists of 528 155 documents [35]. We preprocess the corpus lightly in order to split documents up into words and to allow morphologically similar words to match. The stream of characters in each document gets tokenized into words. Then we conflate terms by applying Porter's stemming algorithm [37]. Next, we discard anything found in a list of 400 "stop" words. Finally, numeric and nonword items are reduced to single tokens ("NUMBER," "DOLLAR," etc.).

The test comprised 50 queries with an average of 57.6 words per query. Each query was preprocessed in the same manner described above for the documents. Then, for each query, we compute (12) for each of the documents. The result is that the top scoring document for each query was found to be relevant 78% of the time.

The simple model described above has been extended by adding more states with different query term-generating mechanisms (e.g., synonyms, bigrams, topics, unsupervised relevance feedback), and by the inclusion of document priors, resulting in higher performance [38].

X. Future Directions

This paper focused on the speech and language technologies that are needed for the indexing and browsing of audio data. State-of-the-art technologies have been integrated in a system, Rough'n'Ready, that automatically takes the audio stream and extracts a structural summarization that is stored in a database and can be retrieved using a browser. The system

dealt specifically with spoken language because speech is always a rich source of information, and frequently—in telephone conversations, for example—it is the only source of information. It is clear, however, that a great deal of information is conveyed by means other than speech. In a medium such as broadcast news, numerous and valuable content cues can be extracted directly from the video or on-screen text. In such a medium, it is important to integrate information from all available modes into a complete index of the content in the medium. We have no doubt that an accurate index of speech content, such as that produced by Rough'n'Ready, can be effectively integrated as a component in a comprehensive multimedia indexing system.

The technologies described in this paper are at various stages of maturity, and much work remains to be done in each of them to improve accuracy in the broadcast news domain. Even though the acoustic environments in broadcast news can be quite varied—all the way from studio quality to live interviews over the telephone or in the field—much of the speech is broadcast from a single microphone usually close to the speaker's mouth. There are other applications, such as meetings, where the acoustic environment is much more challenging. Unless meetings use highly specialized microphone arrays, the captured speech signal is likely to be highly reverberant. Furthermore, meetings are characterized by more spontaneous, conversational speech, as well as overlapped speech from more than one speaker. All these effects will have a deleterious effect on the performance of the various speech technologies. Much research will be required to deal effectively with such applications.

We believe that the technology described in this paper has reached the point where commercial utility may be possible in the very near future, at least for applications similar to broadcast news. No doubt, it will take a number of actual attempts at commercialization before the technology takes root and is used widely. While improving the accuracy of the component technologies will render such systems more usable, it will be other engineering, interface, and human-factors issues that will determine the utility of these systems in the short term. How will the system interface to the sources of data? Where will the data be stored and at what cost? How will the users interact with the system? To what degree will the system have to be tuned to the specific application and at what cost? Will each new application require different capabilities that will have to be included in the system? For example, if the application requires that commercials be identified specifically, then such a capability may have to be developed separately and included. In short, while we believe that the speech processing technologies have reached the point where commercialization has become possible, the actual process of commercialization, as always with any new technology, is fraught with many obstacles and issues that have to be resolved before the technology sees its full potential.

ACKNOWLEDGMENT

Each of the core speech and language technologies in Rough'n'Ready has been supported by DARPA and other agencies of the U.S. government. The technologies have benefited greatly by formal competitive technology evaluations sponsored by DARPA and carried out by NIST over many years. Much of the data used in the development and evaluation of the technologies are distributed by the Linguistic Data Consortium at the University of Pennsylvania. The results of the various evaluations, along with papers from the different sites that participated in the evaluations, can be found in the proceedings of annual DARPA workshops published by Morgan Kaufmann.

The authors would like to thank the reviewers for their excellent and helpful comments.

REFERENCES

[1] M. Lesk. (1997, Oct.) "How much information is there in the world?". [Online] Available: http://www.lesk.com/mlesk/ksg97/ksg.html

[2] M. Christel, T. Kanade, M. Mauldin, R. Reddy, M. Sirbu, S. Stevens, and H. Wactlar, "Informedia digital video library," *Commun. ACM*, vol. 38, no. 4, pp. 57–58, 1995.

[3] A. Hauptmann and M. Witbrock, "Informedia: News-on-demand multimedia information acquisition and retrieval," in *Intelligent Multimedia Information Retrieval*, M. T. Maybury, Ed. Cambridge, MA: AAAI/MIT Press, 1997, pp. 213–239.

[4] H. Wactlar, M. Christel, Y. Gong, and A. Hauptmann, "Lessons learned from the creation and deployment of a terabyte digital video library," *IEEE Comput.*, vol. 32, pp. 66–73, Feb. 1999.

[5] M. Maybury, "Intelligent multimedia for the new millennium," in *Proc. Eurospeech'99*, vol. 1, Budapest, Hungary, Sept. 1999, p. KN1-15.

[6] A. Merlino and M. Maybury, "An empirical study of the optimal presentation of multimedia summaries of broadcast news," in *Automated Text Summarization*, I. Mani and M. Maybury, Eds. Cambridge, MA: MIT Press, 1999, pp. 391–401.

[7] A. Waibel, M. Bett, and M. Finke, "Meeting browser: Tracking and summarising meetings," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*. San Mateo, CA: Morgan Kaufmann, Feb. 1998, pp. 281–286.

[8] F. Kubala, S. Colbath, D. Liu, A. Srivastava, and J. Makhoul, "Integrated technologies for indexing spoken language," *Commun. ACM*, vol. 43, pp. 48–56, Feb. 2000.

[9] F. Kubala, J. Davenport, H. Jin, D. Liu, T. Leek, S. Matsoukas, D. Miller, L. Nguyen, F. Richardson, R. Schwartz, and J. Makhoul, "The 1997 BBN Byblos system applied to broadcast news transcription," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*. San Mateo, CA: Morgan Kaufmann, Feb. 1998, pp. 35–40.

[10] S. Matsoukas, L. Nguyen, J. Davenport, J. Billa, F. Richardson, M. Siu, D. Liu, R. Schwartz, and J. Makhoul, "The 1998 BBN Byblos primary system applied to English and Spanish broadcast news transcription," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, Herndon, VA, Mar. 1999, pp. 255–260.

[11] L. Nguyen, S. Matsoukas, J. Davenport, D. Liu, J. Billa, F. Kubala, and J. Makhoul, "Further advances in transcription of broadcast news," in *Proc. Eurospeech'99*, vol. 2, Budapest, Hungary, Sept. 1999, pp. 667–670.

[12] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, pp. 257–286, Feb. 1989.

[13] D. Liu, L. Nguyen, S. Matsoukas, J. Davenport, F. Kubala, and R. Schwartz, "Improvements in spontaneous speech recognition," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*. San Mateo, CA: Morgan Kaufmann, Feb. 1998, pp. 123–126.

[14] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-5, pp. 179–190, Mar. 1983.

[15] R. Schwartz, L. Nguyen, and J. Makhoul, "Multiple-pass search strategies," in *Automatic Speech and Speaker Recognition: Advanced Topics*, C.-H. Lee, F. K. Soong, and K. K. Paliwal, Eds. Norwell, MA: Kluwer, 1996, pp. 429–456.

[16] L. Nguyen and R. Schwartz, "Efficient 2-Pass N-best decoder," in *Proc. Eurospeech'97*, vol. 1, Rhodes, Greece, Sept. 1997, pp. 167–170.
[17] ——, "The BBN single-phonetic-tree fast-match algorithm," in *Proc. Int. Conf. Spoken Language Processing*, vol. 5, Sydney, Australia, Dec. 1998, pp. 1827–1830.
[18] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density HMMs," *Comput. Speech Lang.*, vol. 9, pp. 171–186, 1995.
[19] J. Davenport, L. Nguyen, S. Matsoukas, R. Schwartz, and J. Makhoul, "Toward realtime transcription of broadcast news," in *Proc. Eurospeech'99*, vol. 2, Budapest, Hungary, Sept. 1999, pp. 651–654.
[20] M. Padmanabhan, E. Jan, L. Bahl, and M. Picheney, "Decision-tree based feature-space quantization for fast Gaussian computation," in *Proc. IEEE Workshop Automatic Speech Recognition and Understanding*, Santa Barbara, CA, Dec. 1997, pp. 325–330.
[21] D. Liu and F. Kubala, "Fast speaker change detection for broadcast news transcription and indexing," in *Proc. Eurospeech'99*, Budapest, Hungary, Sept. 1999, pp. 1031–1034.
[22] H. Gish, M. H. Siu, and R. Rohlicek, "Segregation of speakers for speech recognition and speaker identification," *Proc. Int. Conf. Acoustics, Speech, Signal Processing*, vol. 2, pp. 873–876, May 1991.
[23] S. Chen and P. Gopalakrishnan, "Speaker, environment, and channel change detection and clustering via the Bayesian information criterion," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop,*. San Mateo, CA: Morgan Kaufmann, Feb. 1998, pp. 127–132.
[24] H. Jin, F. Kubala, and R. Schwartz, "Automatic speaker clustering," in *Proc. DARPA Speech Recognition Workshop*. San Mateo, CA: Morgan Kaufmann, Feb. 1997, pp. 108–111.
[25] L. Wilcox, F. Chen, D. Kimber, and V. Balasubramanian, "Segmentation of speech using speaker identification," in *Proc. Int. Conf. Acoustics, Speech, Signal Proc.*, Adelaide, Australia, Apr. 1994, pp. I-161–164.
[26] H. Gish and M. Schmidt, "Text-independent speaker identification," *IEEE Signal Processing Mag.*, pp. 18–32, Oct. 1994.
[27] D. M. Bikel, R. Schwartz, and R. M. Weischedel, "An algorithm that learns what's in a name," in *Mach. Learn.*, 1999, vol. 34, pp. 211–231.
[28] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, pp. 268–278, 1973.
[29] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel, "Performance measures for information extraction," in *Proc. DARPA Workshop on Broadcast News Understanding*. San Mateo, CA: Morgan Kaufmann, Mar. 1999, pp. 249–252.
[30] D. Miller, R. Schwartz, R. Weischedel, and R. Stone, "Named entity extraction from broadcast news," in *Proc. DARPA Workshop on Broadcast News Understanding*. San Mateo, CA: Morgan Kaufmann, Mar. 1999, pp. 37–40.
[31] Y. Yang and C. G. Chute, "An example-based mapping method for text categorization and retrieval," *ACM Trans. Inform. Syst.*, vol. 12, no. 3, pp. 252–277, July 1994.
[32] R. Schwartz, T. Imai, F. Kubala, L. Nguyen, and J. Makhoul, "A maximum likelihood model for topic classification of broadcast news," in *Proc. Eurospeech'97*, Rhodes, Greece, Sept. 1997, pp. 1455–1458.
[33] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc. B*, vol. 39, no. 1, pp. 1–22, 1977.
[34] A. Srivastava, "Story segmentation," Master's thesis, Northeastern Univ., Boston, MA, Aug. 1999.
[35] D. Miller, T. Leek, and R. Schwartz, "BBN at TREC7: Using hidden Markov models for information retrieval," in *Proc. 7th Text Retrieval Conference (TREC-7)*, July 1999, NIST Special Pub. 500-242, pp. 133–142.
[36] M. E. Maron and K. L. Kuhns, "On relevance, probabilistic indexing, and information retrieval," *J. Assoc. Comput. Mach.*, vol. 7, pp. 216–244, 1960.
[37] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.
[38] D. R. H. Miller, T. Leek, and R.M. Schwartz, "A hidden Markov model information retrieval system," in *Proc. 22nd Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Berkeley, CA, Aug. 1999, pp. 214–221.

John Makhoul (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the Massachusetts Institute of Technology (MIT), Cambridge, in 1970.

He is a Chief Scientist at BBN Technologies, GTE Corporation, Cambridge, MA, where he directs various projects in speech recognition and understanding, speech coding, speech synthesis, speech enhancement, signal processing, artificial neural networks, and character recognition. He is also an Adjunct Professor at Northeastern University, Boston University, and the University of Massachusetts, all in Boston.

Dr. Makhoul was a recipient of the 1978 Senior Award, the 1982 Technical Achievement Award, and the 1988 Society Award of the IEEE Signal Processing Society. He is also a Recipient of the IEEE Third Millennium Medal. He is a Fellow of the Acoustical Society of America.

Francis Kubala received the S.B. degree in electrical engineering from the Massachusetts Institute of Technology (MIT), Cambridge, in 1984.

He is currently a Division Scientist at BBN Technologies, GTE Corporation, Cambridge, MA. He works mainly in large-vocabulary continuous speech recognition, with a recent emphasis on developing methods for the automatic indexing and retrieval of audio information through the integration of various speech and language processing technologies, including speech and speaker recognition, story segmentation, named-entity extraction, topic classification, and information retrieval.

Timothy Leek received the B.Sc. degree in physics and the B.A. degree in English from the University of Connecticut, Storrs, in 1991, and the M.S. degree in computer science from the University of California, San Diego, in 1997.

He is currently at BBN Technologies, GTE Corporation, Cambridge, MA, working on hidden Markov model formulations of information extraction, information retrieval, topic spotting, and translation.

Daben Liu (Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from South China University of Technology, Guangzhou, in 1991 and 1994, respectively. He received another M.Sc. degree in electrical engineering from Tufts University, Medford, MA, in 1997.

He is currently with the Speech and Language Department at BBN Technologies, GTE Corporation, Cambridge, MA, where he performs research in speech and speaker technologies, including speech recognition, speaker segmentation, clustering and identification, and audio indexing systems.

Mr. Liu is a member of Eta Kappa Nu.

Long Nguyen (Member, IEEE) was born in Khanh Hoa, Vietnam, on March 20, 1960. He received the B.S. degree in mathematics and computer science from the University of Massachusetts, Boston, in 1990, where he continues to work toward a Ph.D. degree.

Since 1990, he has been with the Speech and Language Department at BBN Technologies, GTE Corporation, Cambridge, MA, where he is now a Senior Scientist performing research in continuous speech recognition, especially fast recognition search algorithms.

Richard Schwartz (Member, IEEE) received the B.S. degree in electrical engineering from the Massachusetts Institute of Technology (MIT), Cambridge, in 1972.

He is a Principal Scientist at BBN Technologies, GTE Corporation, Cambridge, MA. He has worked on phonetic recognition and synthesis, speech coding, speech enhancement in noise, speaker identification and verification, speech recognition and understanding, fast search algorithms, artificial neural networks, online handwriting recognition, optical character recognition, and statistical text processing.

Amit Srivastava (Member IEEE) received the B.Tech. degree (Honors) in electronics and electrical communication engineering from the Indian Institute of Technology (IIT), Kharagpur, India, in 1997, and the M.S.E.E. degree from Northeastern University, Boston, MA, in 1999.

In 1997, he joined the Rough'n'Ready team at BBN Technologies, GTE Corporation, Cambridge, MA, as a Research Assistant in a collaborative effort between Northeastern University and BBN. Since 1999, he has been working as a Staff Scientist at BBN in the fields of audio indexing, topic classification, and story segmentation.