Enriching Speech Recognition with Automatic Detection of Sentence Boundaries and Disfluencies

Yang Liu, *Member, IEEE,* Elizabeth Shriberg, Andreas Stolcke, *Senior Member, IEEE,* Dustin Hillard, *Student Member, IEEE,* Mari Ostendorf, *Fellow, IEEE,* Mary Harper, *Senior Member, IEEE*

***Abstract*— Effective human and automatic processing of speech requires recovery of more than just the words. It also involves recovering phenomena such as sentence boundaries, filler words, and disfluencies, referred to as structural metadata. We describe a metadata detection system that combines information from different types of textual knowledge sources with information from a prosodic classifier. We investigate maximum entropy and conditional random field models, as well as the predominant HMM approach, and find that discriminative models generally outperform generative models. We report system performance on both broadcast news and conversational telephone speech tasks, illustrating significant performance differences across tasks and as a function of recognizer performance. The results represent the state of the art, as assessed in the NIST RT-04F evaluation.**

***Index Terms*— Rich transcription, metadata extraction, prosody, maximum entropy, conditional random field, sentence boundary, disfluency, punctuation, confusion network**

I. Introduction

Although speech recognition technology has improved significantly in recent decades, current recognition systems still output simply a stream of words. The unannotated word stream lacks useful information about punctuation and disfluencies that could assist the human readability of speech transcripts [1]. Such information is also crucial to subsequent natural language processing techniques, which typically work on fluent and punctuated input. Recovering structural information in speech has thus become the goal of many studies in computational speech processing [2], [3], [4], [5], [6], [7], [8]. We describe our approach that automatically adds information about the location of sentence boundaries and speech disfluencies in order to enrich speech recognition output.

To illustrate the importance of structural information, consider the following excerpt of a spontaneous conversational speech transcription:

> *but uh i i i i think that you know i mean we always uh i mean i've i've had a a lot of good experiences with uh with many many people especially where they've had uh extended family and i and an- i i kind of see that that you know perhaps you know we may need to like get close to the family environment*

Even this human-produced transcription, which contains no word errors, is difficult to read because of the absence of punctuation and the presence of speech disfluencies. Automatic detection of such structural events can enrich speech recognition output and make it more useful for downstream language processing modules. For example, if the disfluencies are removed from the transcription and sentences are presented with the appropriate punctuation, the cleaned-up transcription would be as follows:

> *But i've had a lot of good experiences with many people especially where they've had extended family. I kind of see that perhaps we may need to get close to the family environment.*

Clearly, this cleaned-up transcription is more readable, is easier to understand, and is more appropriate for subsequent language processing modules. Jones et al. [1] have conducted experiments showing that cleaned-up transcriptions improve human readability compared to the original transcriptions.

There has been growing interest recently in the study of the impact of structural events. Recent research has investigated whether automatically generated sentence information can be useful to parsing algorithms [9], [10], [11]. Significant error reductions in parsing performance have been achieved when using information about sentence boundaries and disfluencies [10], [11]. Structural information was also found promising for reducing speech recognition error rates [12].

In this paper, we describe the ICSI-SRI-UW structural metadata extraction (MDE) system that automatically detects structural events, including sentence boundaries, disfluencies, and filler words. We aim to answer some important questions: What knowledge sources are helpful for each of these tasks? What are the effective modeling approaches for combining different knowledge sources? How is the model performance affected by various factors such as corpora, transcriptions, and event types? Can structural MDE be improved by considering alternate word hypotheses?

The rest of the paper is organized as follows. We introduce the MDE tasks and performance metrics in Section II. Previous work is described in Section III. Section IV describes the corpora used in the experiments. The general approaches we use for the MDE tasks are introduced in Section V. Sections VI through IX detail the methods used and present results for different MDE tasks. Section X summarizes the results. Future work is presented in Section XI.

II. Structural Metadata Extraction

A. Metadata Annotation

Metadata events such as sentence boundaries, topic boundaries, and disfluencies may be useful for various applications. The structural events annotated in the data set used here are briefly described below (more information on the data will be provided later in Section IV). Details concerning the V6.2 LDC annotation guideline can be found in [13].

1) *Sentence-like Units (SUs):*
The notion of a sentence in conversational speech is different from that in written text. The sentence-like units (SUs) may correspond to a grammatical sentence, or they may be semantically complete but smaller than a sentence (e.g., a noun phrase in response to a question). Four types of SUs are annotated according to the annotation guideline [13]: statement, question, backchannel (e.g., *uhhuh*), and incomplete SUs.
2) *Edit disfluencies:*
Edit disfluencies involve syntactically relevant content that is either repeated, revised, or abandoned. Edit disfluencies follow a basic pattern:

(reparandum) * ⟨editing term⟩ correction

The reparandum is the portion of the utterance that is corrected or abandoned entirely (in the case of restarts). An interruption point (IP), marked with '$*$' in the pattern above, is the point at which the speaker breaks off the original utterance and then repeats, revises, or restarts his or her utterance. The editing term is optional and consists of one or more filler words, which are described in item (3) below. The correction is the portion of the utterance that corrects the original reparandum. Markup of the reparandum and editing terms enables the transcription to be cleaned up.
Based on their structure, edit disfluencies can be divided into the following four subtypes. LDC did not provide subtype information in their annotations, and the correction region in an edit disfluency was also not annotated. We describe the edit disfluency subtypes to motivate the edit detection features used here.

- Repetitions: A speaker repeats some part of the utterance. For example, **[(we may not) * we may not]** *go there.*
- Revisions (content replacements): A speaker modifies the original utterance using a similar syntactic structure. For example, *Show me flights* **[(from Boston on) * ⟨uh⟩ from Denver on]** *Monday.*
- Restarts (false starts): A speaker abandons an utterance or a constituent and then starts over entirely. For example, **[(It's also) *]** *I used to live in Georgia.*
- Complex disfluencies: A speaker produces a series of disfluencies in succession or with nested structure, such as **[(I think * I * I) * ⟨now⟩ I think]** *people generally volunteer*. The internal interruption points were annotated in these disfluencies, but the internal structure was not.

3) *Filler words:*
Fillers include filled pauses, discourse markers, and explicit editing terms. Filled pauses (FPs) are employed by a speaker to indicate hesitation or to maintain control of a conversation while he or she is thinking about what to say next, for example, *ah, eh, uh, um*. A discourse marker (DM) is a word or phrase that functions primarily as a structuring unit of spoken language, for example, *actually, now, anyway, see, basically, so, I mean, well, let's see, you know, like*. To the listener, it signals the speaker's intention to mark a boundary in discourse, including a change in the dominant speaker or the beginning of a new topic. An **explicit** editing term is an editing term in an edit disfluency that is not a filled pause or a discourse marker. For example, in *"Today is [(Monday), ⟨sorry⟩, Tuesday]"*, *"sorry"* is an explicit editing term.

Using the V6.2 annotation specification [13], the training, development, and test sets were annotated by LDC with the following structural information: SUs and their subtypes, the reparandum and IPs in edit disfluencies, and filler words and their types.

B. Metadata Extraction (MDE) Tasks

In the NIST RT-04F evaluation,[1] there are four structural MDE tasks corresponding to the annotated events in the corpus.

- SU detection aims to find the end point of an SU. Detection of subtype (statement, backchannel, question, and incomplete) for each SU boundary was also a task evaluated in RT-04F. However, in this paper we focus only on SU boundary detection, which is clearly an important first step for downstream language processing.
- Filler word detection aims to identify words used as filled pauses, discourse markers, and explicit editing terms.
- Edit word detection aims to find all the words within the reparandum region of an edit disfluency.
- Interruption point (IP) detection aims to find the inter-word location at which point fluent speech becomes disfluent.[2]

For MDE evaluation, two different types of transcriptions are used: human-generated reference transcriptions (REF) and speech-to-text recognition output (STT). Evaluation across transcription types enables the investigation of the structural event detection tasks both with and without the confounding effect of speech recognition errors. Using the reference transcriptions provides the best-case scenario for the evaluation of MDE algorithms. MDE evaluation will be conducted across two different corpora (conversational telephone speech and broadcast news speech). Investigations across corpora should enhance our understanding of how structural information is represented in different speech genres.

C. Performance Measures

Each MDE task is evaluated separately in the NIST RT-04F evaluation. Scoring tools, created for these tasks by NIST, first align the reference and hypothesis words to minimize the word error rate. After alignment, the hypothesized structural events are mapped to the reference events using the word alignment information, and then unmatched structural events

[1] See http://www.nist.gov/speech/tests/rt/rt2004/fall/ for information on the evaluation.

[2] In the RT-04F MDE evaluation, IPs include the starting point of a filler word string in addition to the IPs inside edit disfluencies. We will focus on the IPs in edit disfluencies in this paper since it is a more challenging task and since there is a separate filler word detection task.

(i.e., insertions and deletions) are counted. The error rate is the average number of misclassified boundaries or words per reference event boundary or word. For example, the following shows the SU error rate:

$$\text{SU error rate} = \frac{\text{number of incorrect boundaries}}{\text{total number of SU boundaries}}$$

A detailed description of the scoring tool is provided in [14]. Note that the NIST metric error rates can be greater than 100%. In the following example:

```
Reference:  w1      w2   w3  /   w4
System:     w1  /   w2   w3      w4
               ins          del
```

where w_i is a word and '/' indicates an SU boundary, one insertion and one deletion error (indicated by 'ins' and 'del') result in a NIST SU error rate of 200%.

It is also common to use error trade-off curves in detection tasks, where one may want to consider different operating points. We expect that different applications may prefer different tradeoffs between precision and recall and, in fact, recent experiments show that the best operating point for parsing is not that which minimizes SU error rate [11]. Hence, we also use detection-error tradeoff (DET) curves to compare the performance of systems that output posterior probabilities of the SU events at each word boundary. A DET curve displays the missed detection rate versus the false detection rate on a normal deviate scale.

For assessing the significance of performance differences, we build on prior work by NIST [15] in speech recognition. In particular, we use the matched pair test, comparing scores on segments of speech from a single speaker, bounded by a pause of at least 3 seconds.

III. Previous Work

A. Sentence Boundary Detection

To detect sentence boundaries or punctuation in speech, various approaches have been used. A general hidden Markov model (HMM) framework was designed to combine lexical and prosodic cues for tagging speech with various kinds of hidden structural information, including sentence boundaries [4], [16], punctuation marks [3], [17], disfluencies [18], topic boundaries [16], dialog acts [19], and emotion [20]. Wang and Narayanan [21] developed a method that used only the prosodic features (mostly pitch features) in a multipass way; however, they did not use any textual information, which has been shown to be very important for detecting sentence boundaries. Huang and Zweig [22] developed a maximum entropy model to add punctuation (period, comma, and question mark) in the Switchboard corpus by using a variety of textual cues. Studies on sentence boundary detection in speech have also been conducted for other languages, for example, Chinese [23] and Czech [24]. In this paper, we use two different corpora and two types of transcriptions for SU boundary detection. We investigate different modeling approaches, along with the impact of a greater variety of prosodic and textual features. In addition, experiments cover different MDE tasks (e.g., SU boundary detection and edit disfluency detection).

B. Edit Disfluency Processing

Much of the prior research on automatic disfluency detection has been based largely on human transcriptions of speech and so has focused on textual information. Heeman and Allen [2] proposed a statistical language model to identify part-of-speech (POS) tags, discourse markers, speech repairs, and intonational phrases. Charniak and Johnson [25] built a classifier to predict the reparandum of an edit disfluency based on features such as the POS tags of the preceding word and the following word, and whether or not the word token appeared in a "rough" copy. Recently, Johnson and Charniak [8] evaluated a noisy channel model for speech repair detection. They used a tree adjoining grammar (TAG) to represent the cross-serial dependencies between the reparandum and correction regions. This approach incorporated more syntactic information (via the source syntactic language model and the TAG channel model) and yielded improved performance for disfluency detection. Honal and Schultz have also used a source-channel model for disfluency detection [26], [27].

Several corpus studies have found that combinations of prosodic cues could be used to identify disfluencies with reasonable success, such as [28], [29], [30], [31]. Nakatani and Hirschberg [28] built a decision tree model to detect speech repairs using a variety of acoustic and prosodic signals plus textual features. An HMM approach similar to [16] is used to combine textual and prosodic information for interruption point detection [32]. Snover et al. [6] used transformation based learning (TBL) for the detection of disfluencies in conversational telephone speech, using many lexical features plus the pause information after a word. TBL was also used for disfluency detection by Kim et al. [33], in which prosodic information was combined via a decision tree prosody model.

For disfluency detection, some models in the prior work were trained using the annotated data with both the reparandum and the correction region; however, correction regions are not labeled in the data set we use here. Therefore, we develop models that do not rely on the annotation of the entire edit disfluency region. In addition, many prior studies on disfluency detection assume the availability of sentence boundary information. This is not the scenario used to evaluate our structural event detection tasks; there is no given sentence boundary information for the edit disfluency detection task. Finally, most of the previous work has been conducted on human transcriptions. Because it is important to also evaluate the impact of speech recognition errors on the techniques used for disfluency detection, this research is conducted using human transcriptions and speech recognition output.

IV. MDE Corpora

Conversational telephone speech (CTS) and broadcast news (BN) are used for the structural event detection tasks in this work. CTS and BN are very different genres. They differ, for example, in the average SU length and frequency of disfluencies. Speech in BN contains fewer disfluencies, sentences are longer and more grammatical, and the speakers are mostly professionals reading teleprompted text. Speech in CTS is more casual and conversational, containing many backchannels, filler words, and edit disfluencies.

In this paper we use the data from the NIST RT-04F evaluation. Training data contains about 40 hours of CTS data and 20 hours of BN speech. The training data set is RT-04F MDE training data, that is, LDC2005S16 for the speech and LDC2005T24 for the transcriptions and their annotation. Test data contains 3 hours (36 conversations) of CTS speech and 6 hours (12 shows) of BN speech. Two development sets of similar sizes were used for parameter tuning. The training, development, and evaluation sets were annotated with metadata events by LDC according to the annotation guideline V6.2 [13]. For BN, we combined the training data used in the RT-03F MDE evaluation with the data annotated for RT-04F.[3]

Table I shows the size of the training and testing sets, the word error rate (WER) on the test set,[4] and the distribution of different structural events in the two corpora (per reference words) for the test set. It is worth pointing out that there are slight differences between these distributions on the training and the test sets. The difference is greater for CTS, for which the training set is taken from the Switchboard data and the test set is taken from the Fisher data collection. The WER was obtained from the combination of multiple systems developed for the RT-04F speech recognition evaluation [34], [35].

TABLE I
INFORMATION ON THE CTS AND BN CORPORA, INCLUDING DATA SET SIZES, WER OF THE SPEECH RECOGNIZER ON THE TEST SETS, AND PERCENTAGE OF THE DIFFERENT TYPES OF STRUCTURAL EVENTS IN THE TRAINING SET.

	CTS	BN
Training size (number of words)	484K	353K
Test size (number of words)	35K	46K
Recognizer WER %	14.9	11.7
SU %	14.2	8.0
Edit word %	8.5	1.7
Edit IP %	5.4	1.3
Filler word %	9.2	2.1

V. KNOWLEDGE SOURCES AND GENERAL MODELING APPROACHES INVESTIGATED

For each interword boundary or each word, we use various knowledge sources to determine whether there is a structural event at that boundary or whether that word belongs to an extent of a structural event such as an edit disfluency. The task can be generalized as follows: given a word sequence W and the corresponding features (e.g., prosodic features) F, find the most likely metadata events E. A general description of the knowledge sources and the three approaches (i.e., hidden Markov model, maximum entropy model, and conditional random field) that we have used for the various tasks is provided in this section, whereas the details for each task will be discussed in the relevant sections.

A. The Knowledge Sources

We use both lexical information and prosodic cues to reduce the ambiguity inherent in any one knowledge source.

[3]We did not combine training data for CTS because the differences in annotation used for the two evaluations were greater for CTS than BN.

[4]Note that the recognition output is not released as part of the standard corpora, but is from other experiments [34], [35].

1) Lexical Features: The word identities (from human transcriptions or automatic recognition output) constitute a primary knowledge source for the structural event detection tasks. Some key words are good indicators for events, for example, filler words (e.g., *"uh"*, *"you know"*) and backchannel words (e.g., *"uhhuh"*). Lexical information can also be represented by the co-occurrence of a word with other words or with the neighboring events, a word's part-of-speech tag, or its semantic class. We also use information such as whether part of a word string is repeated for the detection of disfluencies.

2) Prosodic Features and the Prosody Model: Prosodic features reflect information about temporal effects, as well as intonational and and energy contours. Prosody can provide information complementary to the word sequence, and thus is a valuable source of additional information for structural event detection. Furthermore, in the face of high word error rates, prosody is more robust than word-level information. Past research results (such as [5], [16], [28], [36], [37], [38], [39], [40], [41]) suggest that speakers use prosody to impose structure on both spontaneous and read speech. The prosodic features we use are associated with each interword boundary and can be automatically extracted from the words and phonetic alignments of the transcription. We compute prosodic features based on duration, fundamental frequency (F0), energy, and pause information [16]. Duration features, such as word duration, pause duration, and phone-level duration, are normalized by overall phone duration statistics and speaker-specific statistics. To obtain F0 features, pitch tracks are extracted from the speech signal and then post-processed to obtain stylized pitch contours [42], from which F0 features are extracted. Examples of F0 features are the distance from the average pitch in the word to the speaker's pitch floor and the change in the average stylized pitch across a word boundary. Similar processing is performed to obtain energy features.

Some nonprosodic information is used by the prosody model, such as a speaker's gender and speaker change, which can also be automatically extracted from the audio data. Generating speaker change information is easier on CTS than on BN. In CTS, each channel corresponds to one speaker and thus it is easy to detect speaker change using time information. Since BN contains only one channel of audio, estimating speaker change is much less straightforward. For BN, during testing, speech is first segmented based on pause information. Then a speaker label is assigned to each pause-based segment. We have investigated two different methods to generate the speaker labels. One method is based on the ICSI speaker diarization system [43]. The pause-based segments are assigned a speaker label by choosing the speaker who was attributed with the majority of the speech in each segment (by the diarization system). Another method is to use automatic clustering as used for STT (e.g., for speaker adaptation and feature normalization) [44]. The effect of these two approaches to speaker turn change detection will be described in Section VI.

We chose to use a decision tree classifier for the implementation of the prosody model. During training, the decision tree algorithm selects a single feature that has the highest predictive value at each node. During testing, the decision tree generates probabilistic estimates based on the class distribution in a leaf

node. A decision tree classifier is used for several reasons. First, it predicts the posterior probabilities of the metadata events given the prosodic features. These probabilities can be easily combined with a language model. Second, it offers the advantage of interpretability. This is crucial for obtaining a better understanding of how prosodic features are used to signal various event types, and for selecting or designing useful features. Third, the decision tree classifier can handle missing feature values,[5] as well as both continuous and categorical features. We use the IND package [45] in our experiments. More details on decision tree classifiers can be found in [46].

B. Modeling Approaches

1) Hidden Markov Model (HMM): Our baseline model, and the one that forms the basis of much of the prior work on MDE [3], [4], [16], [17], is HMM-like in that the events are treated as hidden states. Given the word sequence W and the prosodic features F, the most likely event sequence E in the HMM framework is:

$$\begin{aligned}\hat{E} &= \text{argmax}_E\, P(E|W,F) = \text{argmax}_E\, P(W,E,F) \\ &= \text{argmax}_E\, P(W,E)P(F|W,E).\end{aligned} \quad (1)$$

We further assume conditional independence of the prosodic features and the word sequence given the events, that is,

$$P(F|E,W) \approx P(F|E) \approx \prod_{i=1}^{n} P(F_i|E_i).$$

This conditional independence assumption is an approximation, since F_i is based on the phone times and labels associated with the word W_i, but the approximation is fairly reasonable since the word identity is not used directly.

There are two sets of parameters in the HMM framework. The **state transition probabilities** are estimated using a hidden event N-gram language model (LM) [47], which models the joint distribution of the word and event sequence $W, E = W_1, E_1, W_2, \ldots E_{n-1}, W_n, E_n$:

$$P(E,W) = \prod_{i=1}^{n} P(W_i|W_{i-k}^{i-1}, E_{i-k}^{i-1})P(E_i|W_{i-k}^{i}, E_{i-k}^{i-1}),$$

letting negative indices for W_i and E_i be start symbols to simplify notation. k is determined by the order used in the N-gram LM. The hidden event LM is trained similarly to the normal N-gram LM except that the SU boundary is inserted explicitly in the word sequence and is a token in the vocabulary. Note that non-SU tokens are not included in the vocabulary. We used the SRILM toolkit [48] for the hidden event LM and Kneser-Ney smoothing, which has been shown to be a very effective LM smoothing method [49].

The second set of HMM parameters is the **observation likelihood** $P(F_i|E_i)$. Instead of training a likelihood model here, we make use of the decision tree prosody model (that estimates $P(E_i|F_i)$) to obtain the observation probabilities:

$$P(F_i|E_i) = \frac{P(E_i|F_i)}{P(E_i)}P(F_i).$$

[5]For example, F0 features in an unvoiced region are missing features.

The most likely sequence $\hat{E}$ given W and F can thus be obtained using the observation likelihoods and the transition probabilities, as follows:

$$\hat{E} = \underset{E}{\text{argmax}}\, P(E|W,F) = \underset{E}{\text{argmax}}\, P(W,E) \prod_i \frac{P(E_i|F_i)}{P(E_i)}. \quad (2)$$

$P(F_i)$ is not shown in the equation above since it is fixed and thus can be ignored when carrying out the maximization. In practice when combining the prosody model and the LM, we use a weighting factor, which is determined based on the development set (roughly 0.8-1.0 is a good range for the weight for the prosody model). In addition, instead of finding the best event sequence, we use the forward-backward algorithm to find the event with the highest posterior probability for each interword boundary:

$$\hat{E}_i = \underset{E_i}{\text{argmax}}\, P(E_i|W,F).$$

This generally minimizes the classification error rate, which is measured for each boundary or each word.

The HMM is a generative modeling approach; it describes a stochastic process with hidden variables (metadata events) that produces the observable data. The HMM approach has two main drawbacks. First, the standard training methods for HMMs maximize the joint probability of the observed and the hidden events, as opposed to the posterior probability of the correct hidden events assignment given the observations. The latter criterion is more closely related to classification performance for the metadata event detection task, which is measured for each word or word boundary. Second, when using correlated textual information, the only principled way of modeling these correlated features in HMMs is to formulate the model in terms of the cross-product of the feature values, thus significantly increasing the number of parameters (as the product of feature set cardinalities). Hence, it is hard to incorporate a wide variety of textual knowledge sources in the HMM.

2) Maximum Entropy (ME) Model: Maximum entropy was first introduced for language processing problems in [50]. We evaluate the ME model in an attempt to overcome the shortcomings of the HMM approach [51]. An ME model, which estimates the conditional probability of the events given the features, takes the exponential form:

$$P(E_i|W,F) = \frac{1}{Z_\lambda(W,F)} exp(\sum_k \lambda_k g_k(E_i,W,F)),$$

where $Z_\lambda(W,F)$ is the normalization term

$$Z_\lambda(W,F) = \sum_{E_i} exp(\sum_k \lambda_k g_k(E_i,W,F)). \quad (3)$$

The functions $g_k(E_i,W,F)$ are indicator functions corresponding to features defined over events, words, and prosodic features. The index k is used to indicate different features, each of which has an associated weight λ_k. For example, one such feature function for the SU detection task might be

$$g(E_i,W,F) = \begin{cases} 1 : \text{if } W_i = \textit{uhhuh} \text{ and } E_i = \textit{SU} \\ 0 : \text{otherwise.} \end{cases}$$

This feature is triggered for the word *uhhuh* followed by a backchannel SU boundary.

The ME model is estimated by finding the parameters λ_k with the constraint that the expected values of the various feature functions $E_P[g_k(E_i, W, F)]$ match the empirical averages in the training data. These parameters ensure the maximum entropy of the distribution and also maximize the conditional likelihood $\prod_i P(E_i|W, F)$ over the training data. In the ME model, decoding is conducted for each sample individually. In our experiments we used an existing ME toolkit,[6] which uses the L-BFGS parameter estimation method [52] and Gaussian-prior smoothing [53] to avoid overfitting. The Gaussian prior was determined based on the development sets and was set to 1 in our experiments.

The conditional likelihood $P(E_i|W, F)$ is a better match to the classification task (i.e., determining the most likely event E_i at each point), which is an advantage of the ME model over an HMM. In addition, the ME framework provides a principled way to combine a large number of overlapping features, as confirmed by the results of [51]; however, it makes the decision for each boundary individually, unlike the HMM, which uses more contextual event information.

3) Conditional Random Fields (CRF): A CRF [54] is an undirected graphical model that defines a global log-linear distribution of the state (or label) sequence E conditioned on an observation sequence, consisting of the word sequence W and prosodic features F. The conditional probability has the following formula:

$$P(E|W, F) = \frac{1}{Z_\lambda(W, F)} exp(\sum_k \lambda_k * G_k(E, W, F)),$$

where the functions G are potential functions over the events and the observations. Each of the potential functions can be either a state feature function of the event at position i and the observation sequence $s(E_i, W, F, i)$ or a transition feature function of the observation sequence and the labels at position i-1 and i, $t(E_{i-1}, E_i, W, F, i, i-1)$. Similar to the ME model, these features are indicator functions that capture the co-occurrence of events and some observations. The key difference between ME and CRF is that the feature functions G_k are over the sequence E in a CRF versus individual events E_i in the ME model. In our case, we use a first-order model that includes only two sequential events in the feature set. The index k represents different features, and λ is the weight for a feature. Z_λ is the normalization term:

$$Z_\lambda(W, F) = \sum_E exp(\sum_k \lambda_k * G_k(E, W, F)). \quad (4)$$

We use the Mallet package [55] to implement the CRF model. The CRF model is trained to maximize the conditional log-likelihood of a given training set $P(E|W, F)$. The most likely sequence E is found using the Viterbi algorithm during testing.[7] To avoid overfitting, similar to training the ME model, we employed a Gaussian prior [53] with a zero mean to the parameters, which was optimized based on the development set.

CRFs have been successfully used for a variety of text processing tasks, such as [54], [56], [57], but have not been widely applied to speech-related tasks that utilize both acoustic and textual knowledge sources.

4) Model Tradeoffs: A CRF differs from an HMM with respect to its training objective function and its handling of correlated textual features. The CRF directly optimizes conditional posterior probabilities of the event labels $P(E|W, F)$. Like ME, the CRF model supports simultaneous correlated features, and therefore gives greater freedom for incorporating a variety of knowledge sources; hence it is able to better utilize multiple representations of the word sequence (e.g., words, POS) without the impact of the independence assumptions used by the HMM. A CRF differs from the ME method with respect to its ability to model sequence information. The primary advantage of the CRF over the ME approach is that the model is optimized globally over the entire sequence; the ME model makes a decision for each point individually without considering the contextual event information. Thus, a conditional random field (CRF) model combines the benefits of the HMM and ME approaches [54].

Training a CRF model is more computationally expensive than the ME model because of its inherent characteristics (optimization at the sequence level versus individual events, cf Equation (3) and (4)). HMM training is the most time efficient among these approaches because it only needs to estimate the N-gram LM parameters in addition to the prosody model, which is needed for all three approaches. Decoding is not costly in any of the approaches.

VI. SU Boundary Detection

We first provide a more detailed description of the features and the setup used by the models for SU detection, and then provide the results.

A. SU Boundary Detection Setup

1) Knowledge Source Refinement: For the textual knowledge sources, in addition to the word identities, we make use of additional corpora and various automatic taggers that map the word sequence to other representations. The tagged versions of the word stream are obtained to support generalizations based on syntactic or semantic structure and to smooth out possibly undertrained word-based probability estimates (the data set is small for training a word-based LM as shown in Table I). The additional textual sources used for SU boundary detection are the following:[8]

- POS tags: The TnT tagger [58] (which does HMM-based POS tagging) is used to obtain POS sequence P for a word sequence W. The POS tagger was trained from two LDC Treebanks, the Switchboard Treebank for CTS data, and the Wall Street Journal Treebank for the BN data.

[6]The ME toolkit is available from http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html.

[7]The forward-backward algorithm would most likely be better here, but it is not currently implemented in the software used [55].

[8]We describe these lexical features here rather than in Section V-A because these are used only for SU detection.

- Automatically induced classes: Word class labels (we used 100 classes) are automatically induced from bigram word distributions [59].
- Chunks: The chunks capture phrase-level information (e.g., noun phrase, verb phrase). The chunk tags for the BN task were generated using a TBL chunker [60] trained on the Wall Street Journal Treebank.
- Out-of-domain corpora: In addition to the MDE-annotated training data released by LDC, we used auxiliary text corpora that are not annotated precisely according to LDC's annotation guideline, but are generally quite large. For example, we used the 130M word corpus of standard broadcast news transcripts with punctuation for the BN task, and the Switchboard Treebank data for CTS. Punctuation was used to approximate SUs for model training.

We used the prosodic features and the decision tree classifiers for the SU prosody model as described in Section V-A.2. We have previously conducted studies to evaluate different sampling approaches for training better decision tree prosody models [61]. Based on our prior work, we employ bagging [62] (50 bags in our experiments) for prosody model training, which has significantly improved SU boundary classification performance. We also examine the impact of different speaker turn identification approaches for SU detection.

2) HMM Setup for SU Boundary Detection: Figure 1 shows the method used during testing to combine the prosody model and various LMs. The region surrounded by the dotted line box is the HMM, for which the state transition probability is obtained by combining three LMs: Word-LM, a word-based LM trained from the LDC data; Word-LM-OOD, a word-based LM trained using the out-of-domain corpus; and AIC-LM, the automatically induced class-based LM trained from the LDC data (after mapping the word sequence W to class sequence C). All these LMs are hidden event 4-gram LMs. The LM probability of W_i given its history H_i from these three LMs in the HMM box is

$$P(W_i|H_i) = \alpha P_{\text{word}}(W_i|H_i) + \beta P_{\text{AIC}}(W_i|H_i) + (1-\alpha-\beta)P_{\text{word-ood}}(W_i|H_i), \quad (5)$$

where the subscript for P is used to indicate the LM type, and weights α and β are tuned using the development set.[9] The probability from the interpolated LMs (as shown in Equation (5)) is then combined with the prosody model as shown in Equation (2). For the two word-based LMs (from the LDC training set and the additional corpus), interpolating them is usually more effective than pooling the training data because it allows control over the contributions of the different sources.

The POS-LM is trained from the LDC training data (after tagging the word sequence W with POS tags P). Since POS tags cannot be obtained on the fly, during testing, we adopt a loosely coupled approach. For each word boundary, the POS-based LM is applied via the HMM approach (without using the prosody model) to generate the posterior probabilities of the events $P_{\text{POS}}(E_i|P)$, where P is the POS sequence corresponding to the word sequence W. This is then combined with $P(E_i|W,F)$, the output from the HMM module, as indicated by the box "posterior probability interpolation" in Figure 1:

$$P_{\text{final}}(E_i|W,F) = \lambda P(E_i|W,F) + (1-\lambda)P_{\text{POS}}(E_i|P).$$

The interpolation weight λ is tuned using the development set.[10] More details on HMM can be found in [63].

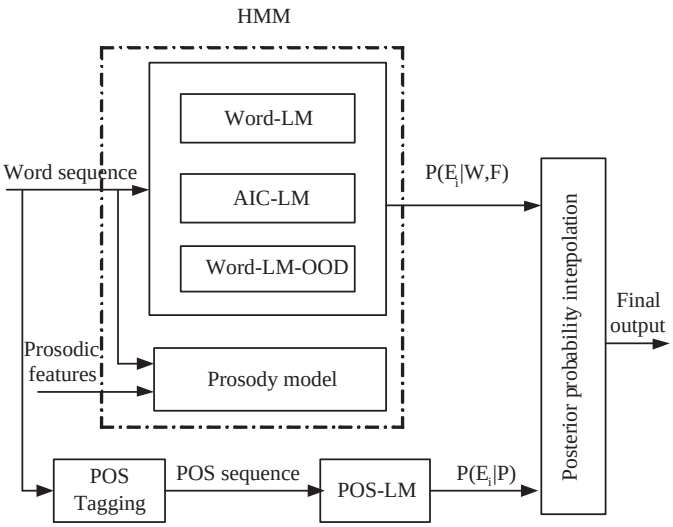

Fig. 1. Integration methods for the various LMs and the prosody model in the HMM for the SU detection task. Word-LM is the word-based LM trained using the LDC training set. AIC-LM is the LM based on the automatically induced classes. Word-LM-OOD is the word-based LM that is trained from the out-of-domain data (i.e., data that is not annotated according to the same annotation guideline). POS-LM is the POS-based LM.

3) Features for the ME and CRF Models on the SU Boundary Detection Task: Knowledge sources similar to those used by the HMM are employed in the ME and CRF models, but with a slightly different representation:

- Word: The word features include different lengths of N-grams (up to 4) and different positional information for a location i, e.g., $<w_i>$, $<w_{i+1}>$, $<w_i, w_{i+1}>$, $<w_{i-1}, w_i>$, $<w_{i-2}, w_{i-1}, w_i>$, and $<w_i, w_{i+1}, w_{i+2}>$.
- POS: Features capturing POS information are similar to those used for words, replacing words W_i with tags P_i.
- Chunk: We use the same combination of contexts for chunk tags as used for words and POS tags. This type of feature is used only on the BN task because in our preliminary studies it did not improve performance on the development set for CTS.
- Class: Similarly to the word and POS N-grams, we also use features from the automatically induced classes.
- Turn: A binary feature is used to indicate whether there is a speaker change.
- Prosody: The prosody model is the same as in the HMM. We encode the posterior probabilities from the prosody

[9] In our experiments, for CTS: α=0.45 and β=0.45; for BN: α=0.2 and β=0.2.

[10] λ is set to 0.9 in our experiments.

model into several binary features through thresholding in a cumulative fashion: $p > 0.1$, $p > 0.3$, $p > 0.5$, $p > 0.7$, $p > 0.9$, with heuristically chosen thresholds. This representation is more robust to the mismatch between the posterior probability in training and test sets, since small changes in the posterior value affect at most one feature. Even though ME does not prevent using continuous features, a preliminary study (see [63]) has shown that using the posterior probabilities as continuous features is less effective in practice than cumulatively binning them. Note that a jack-knife or round-robin paradigm is needed to generate these features for the training set.

- Additional LMs: We include posterior event probabilities from the additional LMs (obtained using the HMM framework) as features, in a way similar to that for the prosody model (i.e., cumulatively binning them using the same thresholds).

Our experiments (see [63]) showed that preserving all the features benefits performance compared to pruning features using information-theoretic metrics. There are about 4 million features for BN and 5 million for CTS in both the ME and CRF models. To date, we have not fully investigated compound features that might be able to improve the performance by modeling the interaction among jointly informative features explicitly. We do include one such feature in our experiments, that is, the combination of the decision tree's hypothesis and POS contexts.

B. SU Boundary Detection Results

For all the MDE tasks we trained the models using the reference transcriptions (plus speech waveforms and metadata annotation), and applied the models to both the reference and the recognition output conditions in the same way. All the results presented in the paper were obtained using the NIST scoring tool md-eval-v20.pl[11].

1) HMM Results: We first look at the HMM SU detection results, since it is the most commonly used approach in previous work. Table II shows the SU boundary detection results using the setup described in Section VI-A.2. For comparison, we also report results from two baseline systems: one used only the hidden event LM, without the prosody model; the other one used the HMM approach, in which a word-based hidden event LM is trained using only the LDC MDE training data, and the decision tree prosody model is trained from a downsampled training set (DS). From Table II, we observe significant improvements over the baseline system by applying bagging in the prosody model and incorporating additional textual knowledge sources in the HMM. Table II also shows the importance of prosodic modeling (e.g., comparing results from the two baseline systems, with and without a prosody model). All the improvements shown in this table are statistically significant with confidence level $p < 0.05$ using the matched pair test described in Section II-C.

Results reported in Table II used speaker change information obtained from speaker diarization results [43]. We have observed significant improvements when the speaker information is obtained from the speaker diarization results compared to using automatic clustering. For example, for the BN reference condition, SU detection error rate is 58.87% using automatic clustering [44] compared to 52.23% using diarization as shown in Table II. Speaker diarization aims to segment speech by speaker and provides more appropriate speaker turn change information for the SU boundary detection task than automatic speaker clustering as used in speech recognition, in which two different speakers can be merged into one cluster for adaptation purposes.

[11]The scoring tool is available from http://www.nist.gov/speech/tests/rt/rt2004/fall/tools/.

TABLE II
SU BOUNDARY DETECTION RESULTS (NIST SU ERROR RATE IN %) ON THE RT-04F EVALUATION DATA SET.

Models	SU Boundary Error Rate (%)			
	CTS		BN	
	REF	STT	REF	STT
Baseline LM:	39.70	46.76	73.74	79.39
Baseline HMM: word LM + prosody (DS)	33.20	41.53	61.41	68.90
Improved HMM: all LMs + prosody (DS)	30.73	39.48	55.94	64.14
Improved HMM: all LMs + prosody (bagging on DS)	28.84	37.47	52.23	60.64

2) Effect of Modeling Approaches: Table III shows the results for each of the different modeling approaches alone and in combination on CTS and BN. One combination result is based on interpolating the HMM and ME posterior probabilities (with weights of 0.5 for each). Another combination result is from the majority vote of the three approaches. The toolkit we use for the implementation of the CRF does not have the functionality of generating a posterior probability for each boundary. Therefore, we do not yet combine the system output via posterior probability interpolation, which we would expect to yield better performance. We will investigate this in future work.

TABLE III
SU BOUNDARY DETECTION RESULTS (NIST SU ERROR RATE %) USING VARIOUS MODELING APPROACHES ALONE AND IN COMBINATION.

	CTS		**BN**	
	REF	STT	REF	STT
HMM	28.84	37.47	52.23	60.64
ME	27.81	37.39	49.60	58.60
CRF	26.55	37.25	49.88	58.21
HMM+ME combination	26.96	36.68	47.44	57.36
Majority vote combination	26.43	36.26	48.21	57.23

As can be seen from the table, on CTS using the reference words, the ME model performs better than the HMM, and the CRF achieves the best performance among the three approaches when used alone. On the STT condition, the ME and CRF models are more negatively impacted by word errors because of their heavier reliance on the textual information, and so their gain over the HMM on the STT condition is negligible. For BN, the conditional models (ME and CRF) perform better than the HMM on both reference and STT conditions. However, the difference between the ME and CRF is marginal, possibly because SUs are generally long in BN, so there is a smaller gain from modeling the event sequence.

On CTS, the majority vote combination is not significantly better than using the CRF alone on the reference condition; however, on the STT condition, the difference between the combined results and the CRF model is statistically significant. On BN, the combination of the three models always yields the best result (significantly better than the best single model). The combined results from the HMM and ME are significantly better than either model alone, and are comparable to the majority vote results on the STT condition. The combination of HMM and ME is better than majority vote on the reference condition, but the improvement is not statistically significant.

There is a large performance degradation for both CTS and BN on the STT condition compared to the reference condition. Generally, the LM is impacted more by word errors than the prosody model [63]. The degradation is greater on CTS than on BN because of the higher WER on CTS. Also notice that the SU error rate is generally higher on BN than on CTS. This is partly because the performance is measured per reference SU event, and on BN the percentage of SUs is smaller than on CTS as shown in Table I. On the other hand, this also suggests that detecting SUs on BN is even harder than on CTS (relative to the chance performance in each domain). In conversational speech, there are many backchannels and first person pronouns, which are very good cues for SU boundaries; for BN, sentence initial and final words are quite variable and thus the data tends to be sparse. In addition, speaker change information is more reliable in CTS than in BN.

3) Performance Tradeoffs across Corpora: Figure 2 illustrates the missed versus false SU detection error for a range of operating points on a DET curve for the reference and STT conditions for CTS and BN. Results are a combination of the confidences from the HMM and ME systems, where the confidences were available. The curves for the STT output level off at a high missed detection rate because of deletions in recognition output; boundary events after deleted words are always "missed" in this scoring framework. The CTS detection performance is always at least as good as that for BN over a range of operating points, with dramatic differences at low false alarm rates. At higher false alarm rates, the missed SU detections are mainly due to STT deletions and differences across tasks become insignificant.

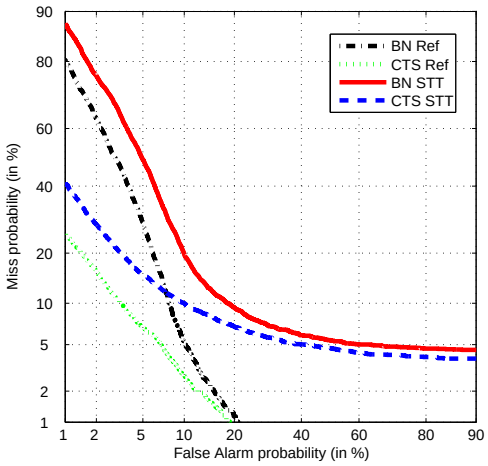

Fig. 2. DET curves for SU boundary detection on both REF and STT conditions for CTS and BN.

VII. Edit Word and IP Detection

A. Edit Detection Approaches

For edit disfluency detection, we need to identify where the edit region starts and ends, or in other words, all the words in an edit disfluency. Since speakers are still fluent at the starting point of an edit disfluency, it is likely that there are no acoustic or lexical cues at that location, but there should be cues at the point when the speaker interrupts his or her speech. Therefore, we use these features in our models to help identify interruption points, which indicate the end of simple edit disfluencies.

1) HMM: Our HMM approach first uses prosodic and lexical features to detect the IP and then uses knowledge-based rules to locate the corresponding reparandum onset for the hypothesized IPs. Figure 3 shows a system diagram for this method.

The top left box in Figure 3, shown with a dashed line, represents a two-way classification model for IP detection using the HMM. The LM is a 4-gram hidden event LM trained from the LDC training set. We trained a decision tree prosody model to detect IP and non-IP boundaries using a downsampled training set[12] to better capture the characteristics of the infrequent IPs. The prosodic features used for IP detection are exactly the same as those used for SU detection. On the reference transcription condition, all the word fragments are IP hypotheses.

A word-based LM can learn only certain frequently occurring disfluencies from the training data and tends not to generalize to other related disfluencies involving different words. To address this issue, we have modified the word-based LM to account for repetitions [64]. In addition, in the repetition detection module, we predefine some cue words that tend to be SUs and so should not be considered to be edit disfluencies (such as *'uhhuh uhhuh'*).

We use a simple rule to resolve the conflict of an IP hypothesis and an SU boundary hypothesis for a word boundary. If the posterior probability of the SU hypothesis is greater than a threshold (0.85 in our experiments), we keep the SU hypothesis and remove the IP hypothesis. Otherwise, we keep the IP hypothesis.

The "knowledge rule" box in Figure 3 uses the IP hypotheses as input, and applies heuristic knowledge to determine the extent of the reparandum in a disfluency (i.e., output where the edit disfluency starts). Linguistic studies (e.g., [65]) suggest that people tend to start from the beginning of a constituent in repetitions or revisions (e.g., repeating the function words). For example, a revision disfluency may be *"a red a blue car"*. If an IP is correctly hypothesized at the interword boundary between *"red "* and *"a"*, that is, *"a red ⟨IP⟩ a blue car"*, then we can search backward to find whether the same word as the word after the IP (*"a"* in this example) has occurred before the IP and thus determine the onset of

[12]We did not use bagging for IP detection because we did not observe significant improvement in a prior study [63].

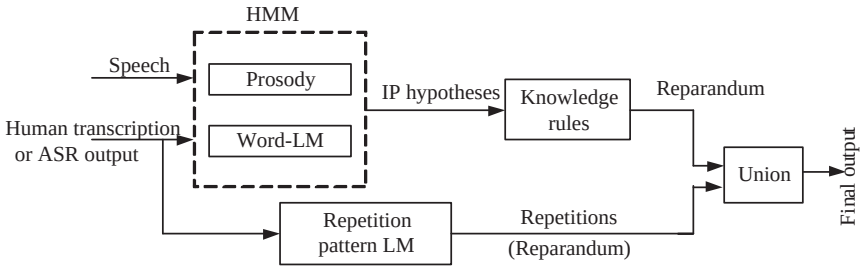

Fig. 3. System diagram for edit word and IP detection.

the edit disfluency. We stop searching backward when an SU hypothesis is met or the number of edit words is greater than a predefined threshold (6 in our experiments). The "union" box combines the output from repetition detection and the HMM-based IP detection followed by the rule-based reparandum onset detection algorithm.

2) ME: In the ME approach, first an ME classifier is used for a 3-way classification: SU, IP, or NULL. Then similar to the HMM, heuristic rules are used to determine the onset of the reparandum. By using three target classes, we can potentially gain from jointly modeling the SU and IP events. Take *"that is great. that is great"* as an example. These are two SUs, rather than an edit disfluency, even though the word sequence is repeated. In the HMM, we predefined some words that should not be considered as edits even though they are repeated, whereas the probabilistic ME model is able to learn these kinds of cue words, together with other features, from the training set and thus model them more elegantly. Also note that in the heuristic rules, the system's SU hypotheses from the ME model itself are used when determining the onset of a reparandum based on the IP hypotheses; therefore, a separate SU detection module is not needed. For the HMM approach we have also investigated using a 3-way classification approach; however, it turned out to be less effective, possibly because IPs are rarer than SUs and the HMM is less able to utilize textual features (as used in the ME model) for IP detection.

The features used in the ME model for the SU/IP/NULL detection task are as follows:

- A All the features used for SU detection as described in Section VI-A.3.
- B Repetition information. At each word boundary, this feature represents whether there is a repeated word sequence (as many as three words) that ends at that point, with optional filler words allowed starting from that point.
- C Fragment information. This feature represents whether a word is a fragment. Only in the reference transcription condition can this feature be triggered. In the speech recognition output condition, no word fragment information is available.
- D Filler words. This feature represents whether there is a predefined filler phrase after a word boundary.[13]
- E Prosody model posterior probabilities. This is obtained from the same prosody model as used in the HMM IP detection module. The posterior probabilities are discretized by cumulative binning, as described in Section VI-A.3 using the same binning thresholds.

[13]This is not from the filler word detection results; rather, a list of cue words is used.

3) CRF: The CRF approach used for edit word detection was designed to find the entire region of the reparandum, similar to the named entity recognition task [57]. Each word has an associated tag, representing whether or not it is an edit word. The classes used in the CRF edit word detection approach are the beginning of an edit (B-E), inside of an edit (I-E), each of which has a possible IP associated with it (B-E+IP or I-E+IP), and outside of an edit (O), resulting in five states, as shown in Table IV. The following is an example of a transcription excerpt together with the class tags used in the CRF edit word detection model:

```
I      *  I   work  * uh  i'm an analyst
B-E+IP   I-E  I-E+IP  O   O   O   O

and it   got   *  it got real rough
O   B-E  I-E+IP   O   O   O    O
```

Note that IPs are included in the target class when using the CRF for edit detection in order to identify the internal IPs inside complex edit disfluencies. For example, "*I I work*" in the above example is the reparandum in a complex edit disfluency, with an internal IP after the first " *I* ".

TABLE IV
STATES AND TRANSITIONS USED BY THE CRF FOR EDIT WORD AND IP DETECTION. THE CLASS TAGS ARE THE BEGINNING OF AN EDIT (B-E), INSIDE OF AN EDIT (I-E), EACH OF WHICH HAS A POSSIBLE IP ASSOCIATED WITH IT (B-E+IP OR I-E+IP), AND OUTSIDE OF AN EDIT (O).

State	Notation	Meaning	Possible state destinations
0	O	outside edit	O, B-E+IP, B-E
1	B-E+IP	begin edit with an IP	O, I-E+IP, I-E
2	B-E	begin edit	I-E+IP, I-E
3	I-E+IP	inside edit with an IP	O, B-E+IP, I-E+IP, I-E
4	I-E	inside edit	I-E+IP, I-E

The CRF model is able to learn valid state transitions from the training data, for example, state 0 cannot transit to state 3 or 4. All of the possible states that a state can go to are shown in the last column of Table IV. An advantage of the CRF method is that it is a probabilistic model capable of representing the information captured by the heuristic rules used by the HMM and ME approaches.

Features used in the CRF method are the N-grams of words and POS tags, speaker turn change (as used in SU detection), and all the features used by the ME IP detection model that are not used for SU detection, that is, features (B) through (E) as described in Section VII-A.2.

B. Edit Detection Results

The different models for edit word and edit IP detection are compared in Table V on the CTS data using the NIST error rate. For the reference condition, the CRF is better at finding edit words, but poorer at IP detection compared to the HMM or ME methods. This is most likely due to how the models are trained: the HMM and ME systems are trained to detect IPs, but the heuristic rules used may not be as accurate at finding the correct onset for the reparandum. The CRF is trained to jointly detect the edit words and IPs and thus may not be as well trained for IP detection. On the STT condition, we observe that the CRF outperforms both the ME and HMM for both the edit word and edit IP tasks, suggesting that the CRF degrades less on this task for the STT condition.

TABLE V
RESULTS (NIST ERROR RATE IN %) FOR EDIT WORD AND IP DETECTION, USING THE HMM, ME, AND CRF APPROACHES ON CTS.

CTS				
Approaches	Edit word		Edit IP	
	REF	STT	REF	STT
HMM	54.19	84.61	34.64	75.89
ME	55.75	87.08	35.01	76.68
CRF	51.49	79.80	37.23	74.35

Table VI shows the results for the BN corpus, using the HMM and the ME approaches. A CRF was not used for the BN data because disfluencies are rare in BN, so there is a data sparsity problem. The ME approach yields better results for both edit word and IP detection than the HMM, in contrast to the CTS corpus. Performance degrades severely in the STT condition compared with reference transcriptions, with a larger error rate increase than that observed for the CTS edit word and IP detection tasks.

TABLE VI
RESULTS (NIST ERROR RATE IN %) FOR EDIT WORD AND IP DETECTION USING THE HMM AND ME APPROACHES ON BN.

BN				
Approaches	Edit word		Edit IP	
	REF	STT	REF	STT
HMM	44.80	91.66	33.62	91.81
ME	42.62	89.60	30.72	88.91

We observe from the table that the edit detection error rate on BN is not worse than CTS for the reference condition, even though the percentage of edit words is much smaller on BN than CTS, which significantly affects the denominator used in the performance measure. This suggests that, for the reference condition, edit word detection is an easier task on BN than on CTS. This makes sense since many of the disfluencies in BN are simple repetitions, whereas disfluencies are more complex in CTS.

Also note that an important feature for edit word and IP detection is the occurrence of word fragments, which is provided in the reference condition but is unavailable in the STT condition. The severe performance degradation in the STT condition is partly due to the unavailability of such word fragment information, as well as word errors in the edit disfluency region. A preliminary study [66] has shown that there are some acoustic-prosodic cues that can be effectively used for automatic word fragment detection, so in future work, we will attempt to incorporate those features.

VIII. FILLER WORD DETECTION

Presently, our system detects only filled pauses (FPs) and discourse markers (DMs), not explicit editing terms because they are extremely rare events (fewer than 0.1%). For filler word detection, we evaluated only the HMM boundary detection approach. The HMM first detects the filler word boundary, and then looks backward to find the onset of the filler word string based on a predefined list of possible filler words. Since filled pauses are only single words, we need to identify only the start of the filler words for discourse markers. We hypothesize that FPs and DMs are quite different phenomena; therefore, we use a separate model for each. For each of the FP and DM detection tasks, a prosody model is trained from a downsampled training set, using the same prosodic features as used for the SU and edit IP detection tasks. 4-gram hidden event word-based LMs are used, in which the events are the end boundary of an FP or DM, respectively.

Table VII shows the results for filler word detection. Again, performance degrades significantly in the STT condition. For tasks such as filler word detection, which strongly depend on word identity, inaccurate STT output severely affects the LM's performance. In related experiments [63], we found that the effective prosodic features for filler detection are quite different from those for SU detection, for example, duration features (e.g., word lengthening) are used more frequently in the decision trees for filler word detection than for SU or edit disfluency detection.

TABLE VII
FILLER WORD DETECTION RESULTS (NIST ERROR RATE %) FOR BN AND CTS, ON REF AND STT CONDITIONS.

Conditions		Error Rate (%)
CTS	REF	26.98
	STT	41.66
BN	REF	18.11
	STT	56.00

IX. EFFECT OF WER

A. Effect of WER across SU and Edit Word Detection Tasks

As we have already observed, there is a decreased accuracy for the MDE system when testing on STT output compared to human transcriptions, largely due to recognition errors. To understand just how much WER affects performance, we consider STT output from different recognition systems. Table VIII shows SU and edit word detection results using different STT systems on CTS and BN corpora. We chose to

focus on the SU and Edit detection tasks here because they are more challenging than filler word detection. The WER for each STT system is indicated in the table. For comparison, we also show results when using the reference transcription, which has essentially a 0% WER. The SU detection system is the majority vote of the HMM, ME, and CRF models. The edit word detection system is the CRF model for CTS, and the ME model for BN.

TABLE VIII
SU AND EDIT WORD DETECTION RESULTS (NIST ERROR RATE IN %) FOR CTS AND BN, ON REF AND VARIOUS STT CONDITIONS. STT-1 AND STT-2 ARE TWO DIFFERENT STT OUTPUTS, AND THE WER (%) FOR THEM IS SHOWN IN THE TABLE.

Conditions		WER	SU boundary	Edit word
CTS	REF	0	26.43	51.49
	STT-1	14.9	36.26	79.80
	STT-2	18.6	40.27	80.55
BN	REF	0	48.21	42.62
	STT-1	11.7	57.23	89.60
	STT-2	15.0	60.40	91.40

Comparing the results using the reference transcripts and STT outputs, we find that word errors have a more negative impact on edit word detection than on SU detection and that generally system performance degrades to a relatively greater degree when using a less accurate recognition output; however, the relationship between WER and structural event detection performance appears to be nonlinear, especially for edit word detection. For edit word detection, better STT accuracies improve performance only slightly, and there is a large gap between using the best STT output and the reference condition. This suggests that for STT output, more errors may occur in the region of edit disfluencies, and thus the word errors have a greater impact on the edit word detection task. Also recall that an important difference between the reference transcription and STT output for edit word detection is whether or not word fragment information is available. The lack of word fragment knowledge in the STT case greatly impacts edit word detection performance. Word errors in different positions may have different levels of impact on SU detection as well. Intuitively, sentence initial and final words have a greater effect on the system performance for SU detection. In addition, deletion errors in STT output are more likely to occur in short SUs, such as backchannels, and so would have a more severe impact on SU detection than other deletion word errors that occur in the middle of an utterance.

B. Joint Word and SU Decoding

Speech recognition errors limit the effectiveness of SU boundary prediction, because they introduce incorrect words to the word stream. As our experiments have shown, SU boundary detection error rates increased by roughly 20% and 40% relative, for BN and CTS respectively, when moving from the reference transcript to recognition output. Including additional recognizer hypotheses may allow for alternative word choices to improve SU boundary prediction.

In [67], we explored the use of N-best hypotheses in SU detection. The HMM (or HMM+ME) system was run on each hypothesis, and SU/non-SU events were inserted into the word stream at each word boundary with their respective posterior probabilities. The resulting hypotheses were then combined into a single word confusion network [68],[14] using only the recognition scores so that the single best word stream thus obtained was not changed by adding the SU events. The SU events in this network had probabilities based on a combination of predictions from individual sentence hypotheses, each weighted by the recognizer posterior for that hypothesis. Choosing SU events based on these probabilities gave a small reduction in SU error rate, but gains were not statistically significant, and there was no opportunity to improve ASR performance.

To consider a larger number of word hypotheses and with the hope of positively impacting recognition performance, we investigate here lattice-based joint decoding of words and SUs. To simplify the implementation, only the HMM SU framework is used. First, prosodic features are extracted over all words in the lattice, and then prosodic posteriors from the decision trees are attached to each possible boundary position (after each word). The probabilities for each of the SU LMs in the HMM model are also associated with each arc in the lattice. Running the forward-backward algorithm on the lattice, with the same score combination weights as in independent recognition and SU decoding, we obtain SU and non-SU posteriors for each word boundary in the lattice. The resulting log posteriors are used in joint word and SU decoding, with a tunable weight for the SU scores. The use of SU posteriors (that sum to one) rather than the separate SU-related scores is important for ensuring that word posteriors (as computed by the forward-backward algorithm) are unchanged.

The plot in Figure 4 illustrates the tradeoff in SU error and WER as the weight on the SU posterior is varied in STT decoding, for the CTS task. As the weight of the SU score is increased relative to the other STT scores (e.g., acoustic, LM, pronunciation), the SU error decreases, but it eventually causes a degradation in WER. Unfortunately, only small gains are obtained before causing a degradation in WER. To better understand this result, we analyzed confidence estimates for SUs and words to determine whether the SU detection scores are causing problems. As it turned out, the SU posteriors are relatively good estimates of confidence of an SU, but the word confidences are optimistically biased (i.e., higher than the frequency of correct words). We conjecture that this bias could be one reason for the lack of benefit to SU performance from considering different recognition hypotheses. The fact that SUs provide no benefit to recognition performance may be due to their relatively high error compared to recognition error rate, but it may also be the case that a more sophisticated integration of SUs and language modeling is needed.

X. SUMMARY

We have described our modeling efforts for rich transcription of speech and shown the MDE performance of a state-of-the-art system. We have investigated various knowledge

[14]A confusion network is a compacted representation of a word lattice or N-best list that is essentially a series of slots with a set of word hypotheses (and null arcs) and associated posterior probabilities.

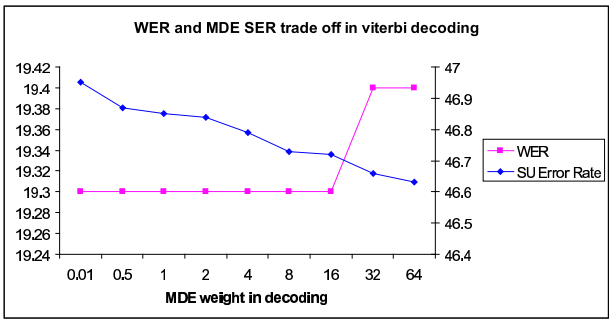

Fig. 4. The tradeoff in SU error rate and WER in lattice decoding on CTS.

sources and modeling approaches for different tasks in both conversational and broadcast news speech, using both reference transcriptions and speech recognition output. A significant difference between corpora and speaking styles was observed. Prosodic and textual information showed different strengths across models and tasks. For SU detection, we have shown that prosodic model performance was greatly improved by using bagging to build more robust classifiers for probability estimation, and that different textual information yields an additional gain beyond a purely word-based language model. We have explored new modeling techniques (i.e., the ME and CRF modeling approaches) in an attempt to address problems in the previous HMM approach. These methods are powerful at incorporating various features and outperform the HMM under most conditions. The combination of models generally achieved the best performance. We have examined the impact of WER across models and tasks. In addition, we looked at joint decoding of words and SU boundary events, but found that only small improvements in SU detection were obtained before WER degraded.

Similar knowledge sources were used in the HMM, ME, and CRF approaches, but with different representations. We have tried to optimize parameters for each of the approaches, in an attempt to compare the models in as parallel a fashion as possible. However, it should be noted that our main goal was to evaluate the inherent capabilities of these three modeling approaches to combine prosodic and textual knowledge sources; hence, we used features differently across the models in many cases.

XI. Future Work

We have focused only on SU boundary detection in this paper and left out SU subtype detection, for which our system uses a simple ME classifier after the SU boundaries are detected [63]. Subtypes of SUs (e.g., statements versus questions) could be useful for a spoken language understanding system. Progress on this task may be made by consulting the literature in related areas, such as dialog act modeling.

In the ME and CRF models, we have adopted a modular approach for incorporating the prosody model. Our future research will include the investigation of methods to more directly incorporate prosodic features into the ME or CRF models, as well as the design of compound features to model the interaction among the various knowledge sources. In addition, we plan to generate posterior probabilities for an event in the CRF model for more effective system combination. Given the results of Johnson et al. [69], we also plan to investigate methods to incorporate more syntactic information into our models for edit detection. For filler word detection, we plan to build word-dependent prosody models for discourse marker words such as *"so"* and *"like"*.

To further improve MDE performance, we will continue to search for better features and modeling approaches that are more discriminative and robust to speech recognition errors. We will also investigate how to make use of unlabeled or partially labeled data. Another future direction is to develop a unified framework that can detect various metadata events jointly, instead of building a separate model for each event.

Automatic detection of structural events can enrich speech recognition output, improve human readability, and aid downstream language processing modules. Investigating the impact of structural event detection on downstream applications, such as parsing, machine translation, summarization, and information extraction, is also an important future direction.

Acknowledgments

The authors are grateful for Barbara Peskin's advice while she was a principal investigator for the EARS MDE tasks. This research has been supported by DARPA under contract MDA972-02-C-0038, NSF under NSF-IRI 9619921, and ARDA under contract MDA904-03-C-1788. Distribution is unlimited. Any opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies. Part of this work was carried out while the last author was on leave from Purdue University and at NSF.

References

[1] D. Jones, F. Wolf, E. Gibson, E. Williams, E. Fedorenko, D. Reynolds, and M. Zissman, "Measuring the readability of automatic speech-to-text transcripts," in *Proc. of Eurospeech*, 2003, pp. 1585–1588.

[2] P. Heeman and J. Allen, "Speech repairs, intonational phrases and discourse markers: Modeling speakers' utterances in spoken dialogue," *Computational Linguistics*, vol. 25, pp. 527–571, 1999.

[3] J. Kim and P. C. Woodland, "The use of prosody in a combined system for punctuation generation and speech recognition," in *Proc. of Eurospeech*, 2001, pp. 2757–2760.

[4] Y. Gotoh and S. Renals, "Sentence boundary detection in broadcast speech transcripts," in *Proc. of ISCA Workshop: Automatic Speech Recognition: Challenges for the new Millennium ASR-2000*, 2000, pp. 228–235.

[5] R. Kompe, *Prosody in Speech Understanding System*. Springer-Verlag, 1996.

[6] M. Snover, B. Dorr, and R. Schwartz, "A lexically-driven algorithm for disfluency detection," in *Proc. of HLT/NAACL*, 2004.

[7] J. Kim, "Automatic detection of sentence boundaries, disfluencies, and conversational fillers in spontaneous speech," Master's thesis, University of Washington, 2004.

[8] M. Johnson and E. Charniak, "A TAG-based noisy channel model of speech repairs," in *Proc. of ACL*, 2004.

[9] M. Gregory, M. Johnson, and E. Charniak, "Sentence-internal prosody does not help parsing the way punctuation does," in *Proc. of HLT/NAACL*, 2004.

[10] J. G. Kahn, M. Ostendorf, and C. Chelba, "Parsing conversational speech using enhanced segmentation," in *Proc. of HLT/NAACL*, 2004.

[11] M. Harper, B. Dorr, B. Roark, J. Hale, Z. Shafran, Y. Liu, M. Lease, M. Snover, L. Young, R. Stewart, and A. Krasnyanskaya, "Final report: parsing speech and structural event detection," http://www.clsp.jhu.edu/ws2005/groups/eventdetect/documents/ finalreport.pdf, 2005.

[12] S. Coquoz, "Broadcast news segmentation using MDE and STT information to improve speech recognition," International Computer Science Institute, Tech. Rep., 2004.

[13] S. Strassel, *Simple Metadata Annotation Specification V6.2*, Linguistic Data Consortium, 2004. [Online]. Available: http://www.ldc.upenn.edu/Projects/MDE/Guidelines/SimpleMDE_V6.2.pdf

[14] NIST, http://www.nist.gov/speech/tests/rt/rt2004/fall/, 2004.

[15] ——, "Significance Tests for ASR," http://www.nist.gov/speech/tests/sigtests/sigtests.htm, 2000.

[16] E. Shriberg, A. Stolcke, D. Hakkani-Tur, and G. Tur, "Prosody-based automatic segmentation of speech into sentences and topics," *Speech Communication*, pp. 127–154, 2000.

[17] H. Christensen, Y. Gotoh, and S. Renal, "Punctuation annotation using statistical prosody models," in *ISCA Workshop on Prosody in Speech Recognition and Understanding*, 2001.

[18] A. Stolcke, E. Shriberg, R. Bates, M. Ostendorf, D. Hakkani, M. Plauche, G. Tur, and Y. Lu, "Automatic detection of sentence boundaries and disfluencies based on recognized words," in *Proc. of ICSLP*, 1998, pp. 2247–2250.

[19] A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. V. Ess-Dykema, and M. Meteer, "Dialogue act modeling for automatic tagging and recognition of conversational speech," *Computational Linguistics*, vol. 26, pp. 339–373, 2000.

[20] J. Ang, R. Dhilon, A. Krupski, E. Shriberg, and A. Stolcke, "Prosody-based automatic detection of annoyance and frustration in human-computer dialog," in *Proc. of ICSLP*, 2002, pp. 2037–2040.

[21] D. Wang and S. S. Narayanan, "A multi-pass linear fold algorithm for sentence boundary detection using prosodic cues," in *Proc. of ICASSP*, 2004.

[22] J. Huang and G. Zweig, "Maximum entropy model for punctuation annotation from speech," in *Proc. of ICSLP*, 2002, pp. 917–920.

[23] C. Zong and F. Ren, "Chinese utterance segmentation in spoken language translation," in *The 4th International Conference on Computational Linguistics and Intelligent Text Processing*, 2003, pp. 516–525.

[24] J. Kolar, J. Svec, and J. Psutka, "Automatic punctuation annotation in czech broadcast news speech," in *Proc. of the 9th conference Speech and Computer*, 2004.

[25] E. Charniak and M. Johnson, "Edit detection and parsing for transcribed speech," in *Proc. of NAACL*, 2001, pp. 118–126.

[26] M. Honal and T. Schultz, "Automatic disfluency removal on recognized spontaneous speech - rapid adaptation to speaker dependent disfluencies," in *Proc. of ICASSP*, 2005.

[27] ——, "Corrections of disfluencies in spontaneous speech using a noisy-channel approach," in *Proc. of Eurospeech*, 2003.

[28] C. Nakatani and J. Hirschberg, "A corpus-based study of repair cues in spontaneous speech," *Journal of the Acoustical Society of America*, pp. 1603–1616, 1994.

[29] E. Shriberg, "Phonetic consequences of speech disfluency," in *Proc. of the International conference of Phonetics Sciences*, 1999, pp. 619–622.

[30] R. Lickley, "Juncture cues to disfluency," in *Proc. of ICSLP*, 1996.

[31] G. Savova and J. Bachenko, "Prosodic features of four types of disfluencies," in *Proc. of DiSS*, 2003, pp. 91–94.

[32] E. Shriberg and A. Stolcke, "A prosody-only decision-tree model for disfluency detection," in *Proc. of Eurospeech*, 1997, pp. 2383–2386.

[33] J. Kim, S. Schwarm, and M. Ostendorf, "Detecting structural metadata with decision trees and transformation-based learning," in *Proc. of HLT/NAACL*, 2004.

[34] B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau, G. Zweig, A. Stolcke, R. Gadde, W. Wang, J. Zheng, B. Chen, and Q. Zhu, "The bicoastal IBM/SRI CTS STT system," in *Rich Transcription (RT-04F) Workshop*, November 2004.

[35] P. Woodland, H. Chan, G.Evermann, M. Gales, D. Kim, X. Liu, D. Mrva, K. Sim, L. Wang, K. Yu, J. Makhoul, R. Schwartz, L. Nguyen, S. Masoukas, B. Xiang, M. Afify, S. Abdou, J. Gauvain, L. Lamel, H. Schwenk, G. Adda, F. Lefevre, D. Vergyri, W. Wang, J. Zheng, A. Venkataraman, R. R. Gadde, and A. Stolcke, "SuperEARS: Multi-Site Broadcast News System," in *Rich Transcription (RT-04F) Workshop*, November 2004.

[36] W. N. Campbell, "Durational cues to prominence and grouping," in *Proc. of ECSA Workshop on Prosody*, Lund, Sweden, 1993, pp. 38–41.

[37] J. R. De Pijper and A. A. Sanderman, "On the perceptual strength of prosodic boundaries and its relation to suprasegmental cues," *Journal of the Acoustical Society of America*, vol. 96, no. 4, pp. 2037–2047, October 1994.

[38] D. Hirst, "Peak, boundary and cohesion characteristics of prosodic grouping," in *Proc. of ECSA Workshop on Prosody*, Lund, Sweden, 1993, pp. 32–37.

[39] P. J. Price, M. Ostendorf, S. Shattuck-Hufnagel, and C. Fong, "The use of prosody in syntactic disambiguation," *Journal of the Acoustical Society of America*, vol. 90, no. 6, pp. 2956–2970, 1991.

[40] D. R. Scott, "Duration as a cue to the perception of a phrase boundary," *Journal of the Acoustical Society of America*, vol. 71, no. 4, pp. 996–1007, 1982.

[41] M. Swerts, "Prosodic features at discourse boundaries of different strength," *Journal of the Acoustical Society of America*, vol. 101, no. 1, pp. 514–521, January 1997.

[42] K. Sonmez, E. Shriberg, L. Heck, and M. Weintraub, "Modeling dynamic prosodic variation for speaker verification," in *Proc. of ICSLP*, 1998, pp. 3189–3192.

[43] C. Wooters, J. Fung, B. Peskin, and X. Anguera, "Towards robust speaker segmentation: The ICSI-SRI fall 2004 diarization system," in *Rich Transcription (RT-04F) Workshop*, 2004.

[44] A. Sankar, L. Heck, and A. Stolcke, "Acoustic modeling for the SRI Hub4 partitioned evaluation continuous speech recognition system," in *Proceedings DARPA Speech Recognition Workshop*. Chantilly, VA: Morgan Kaufmann, Feb. 1997, pp. 127–132. [Online]. Available: http://www.nist.gov/speech/proc/darpa97/html/sankar1/sankar1.htm

[45] W. Buntime and R. Caruana, *Introduction to IND version 2.1 and Recursive Partitioning*. NASA Ames Research Center, Moffett Field, CA, 1992.

[46] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[47] A. Stolcke and E. Shriberg, "Automatic linguistic segmentation of conversational speech," in *Proc. of ICSLP*, 1996, pp. 1005–1008.

[48] A. Stolcke, "SRILM – An extensible language modeling toolkit," in *Proc. of ICSLP*, 2002, pp. 901–904.

[49] S. F. Chen and J. T. Goodman, "An empirical study of smoothing techniques for language modeling," Harvard University, Computer Science Group, Tech. Rep., 1998.

[50] A. L. Berger, S. A. D. Pietra, and V. J. D. Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, pp. 39–72, 1996.

[51] Y. Liu, A. Stolcke, E. Shriberg, and M. Harper, "Comparing and combining generative and posterior probability models: Some advances in sentence boundary detection in speech," in *Proc. of EMNLP*, 2004.

[52] R. H. Ryrd, P. Lu, and J. Nocedal, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific and Statistical Computing*, vol. 16,5, pp. 1190–1208, 1995.

[53] S. Chen and R. Rosenfeld, "A Gaussian prior for smoothing maximum entropy models," Carnegie Mellon University, Tech. Rep., 1999.

[54] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random field: Probabilistic models for segmenting and labeling sequence data," in *Proc. of ICML 2001*, 2001, pp. 282–289.

[55] A. McCallum, "Mallet: A machine learning for language toolkit," 2002, http://mallet.cs.umass.edu.

[56] F. Sha and F. Pereira, "Shallow parsing with conditional random fields," in *Proc. of HLT/NAACL'03*, 2003.

[57] A. McCallum and W. Li, "Early results for named entity recognition with conditional random fields," in *Proc. of CoNLL*, 2003.

[58] T. Brants, "TnT a statistical part-of-speech tagger," in *Proc. of the 6th Applied NLP Conference*, 2000, pp. 224–231.

[59] P. F. Brown, V. J. D. Pietra, P. V. DeSouza, J. C. Lai, and R. L. Mercer, "Class-based n-gram models of natural language," *Computational Linguistics*, pp. 467–479, 1992.

[60] G. Ngai and R. Florian, "Transformation-based learning in the fast lane," in *Proc. of NAACL 2001*, June 2001, pp. 40–47.

[61] Y. Liu, N. Chawla, M. Harper, E. Shriberg, and A. Stolcke, "A study in machine learning from imbalanced data for sentence boundary detection in speech," *Computer Speech and Language*, To Appear.

[62] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24(2), pp. 123–140, 1996.

[63] Y. Liu, "Structural event detection for rich transcription of speech," Ph.D. dissertation, Purdue University, 2004.

[64] Y. Liu, E. Shriberg, and A. Stolcke, "Automatic disfluency identification in conversational speech using multiple knowledge sources," in *Proc. of Eurospeech*, 2003, pp. 957–960.

[65] H. H. Clark and T. Wasow, "Repeating words in spontaneous speech," *Cognitive Psychology*, pp. 201–242, 1998.

[66] Y. Liu, "Word fragment identification using acoustic-prosodic features in conversational speech," in *HLT Student Workshop*, 2003.

[67] D. Hillard, M. Ostendorf, A. Stolcke, Y. Liu, and E. Shriberg, "Improving automatic sentence boundary detection with confusion networks," in *Proc. of HLT/NAACL*, 2004, pp. 69–72.

[68] L. Mangu, E. Brill, and A. Stolcke, “Finding consensus in speech recognition: word error minimization and other applications of confusion networks,” *Computer Speech and Language*, pp. 373–400, 2000.
[69] M. Johnson, E. Charniak, and M. Lease, “Research on edit detection,” in *Rich Transcription (RT-04F) Workshop*, 2004.