Text Segmentation Using Exponential Models

Doug Beeferman, Adam Berger, and John Lafferty

School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

Abstract

This paper introduces a new statistical approach to automatically partitioning text into coherent segments. Our proposed model enlists both long-range and short-range language models to help it sniff out likely sites of topic changes in text. To aid its search, the model consults a set of simple lexical hints it has learned to associate with the presence of boundaries through inspection of a large corpus of annotated data. We also propose a new probabilistically motivated error metric for use by the natural language processing community, intended to supersede precision and recall for appraising segmentation algorithms. Qualitative assessment of our algorithm as well as evaluation using this new metric demonstrate the effectiveness of our approach in two very different domains, *Wall Street Journal* articles and broadcast news transcripts.

1 Introduction

The task we are concerned with in this paper seems on the face of it quite simple: identify where one region of text ends and another begins. This work was motivated by the observations that such a seemingly simple problem is actually quite difficult, and that a tool for partitioning a stream of undifferentiated text or multimedia into coherent regions would be of great benefit in a number of existing applications.

The task itself is ill-defined: what exactly is meant by the "end" of a region of text? We confront this issue by adopting an empirical definition of segment. At our disposal is a collection of online data (38 million words of Wall Street Journal archives and another 150 million words from selected news broadcasts) in which the boundaries between regions (articles or news segments, respectively) are explicit. With this data in hand, the task of constructing a segmenter may be cast as a problem in machine learning: "learn" to identify segment boundaries by discovering correlations in the data.

The existence of a general-purpose tool for partitioning expository text or multimedia data into coherent regions would have a number of immediate practical uses. In fact, this research was inspired by a problem in information retrieval: given a large unpartitioned collection of expository text and a user's query, return a collection of coherent segments which matches the user's query. Lacking a segmenting tool, an IR application may be able to locate positions in its database which are strong matches with the user's query, but be unable to determine how much of the surrounding data to provide to the user. This can manifest itself in quite unfortunate ways. For example, a video-on-demand application (such as the one described in [CKM+95]) responding to a query about a recent news event may provide the user with a news clip related to the event, followed by or preceded by (or both) an unrelated story or even a commercial.

Document summarization is another fertile area for an automatic segmenter. Summarization tools often work by breaking the input into "topics" and then summarizing each topic independently. A segmentation tool has obvious applications to the first of these tasks.

The output of a segmenter could also serve as input to various language-modeling tools. For instance, one could envision segmenting a corpus, classifying the segments by topic, and then constructing topic-dependent language models from the generated classes.

The paper will proceed as follows. In Section 2 we review some previous approaches to the text segmentation problem. In Section 3 we describe our model, including the type of linguistic clues it looks for in deciding when placing a partition is appropriate. In Section 4 we describe a feature induction algorithm that automatically constructs a set of the most informative clues. How to evaluate a tool for partitioning text, multimedia, or other types of data is a further matter for discussion, which we take up in Section 5. It is here that we introduce a new, probabilistically motivated way to evaluate a text segmenter such as ours. Finally, in Section 6 we demonstrate our model's effectiveness on two distinct domains.

2 Previous work

In this section we consider past approaches to the text segmentation problem, including data-driven algorithms like ours and an approach that requires a pre-existing knowledge base of word similarity information.

2.1 Text tiling

The TextTiling algorithm, introduced by Hearst [Hea94], segments expository texts into multiple paragraphs of coherent discourse units. Hearst's work emphasizes subtopic structure, although nothing excludes it from more objectively defined applications like article segmentation. A cosine measure is used to gauge the similarity between constant-size blocks of morphologically analyzed tokens. First-order rates of change of this measure are then calculated to decide the placement of boundaries between blocks, which are then adjusted to coincide with the paragraph segmentation, provided as input to the algorithm.

This approach leverages the observation that text segments are dense with repeated content words. Relying on this fact limits precision, because the repetition of concepts within a document is more subtle than can be recognized by only a "bag of words" tokenizer and morphological filter. Word pairs other than "self-triggers", for example, can be discovered automatically from training data using the techniques of mutual information employed by our language model. (As observed in [KF94] in regard to Youmans' [You91] Vocabulary Management Profile, words are often not simply repeated, but they are restated in different ways, particularly in vocabulary-rich domains.)

Furthermore, Hearst's approach segments at the paragraph level, which is too coarse for applications like information retrieval on transcribed or automatically recognized spoken documents, in which paragraph boundaries are not known.

2.2 Lexical cohesion

Kozima [Koz93] employs a "lexical cohesion profile" to keep track of the cohesiveness of words in a text within a fixed-length window. In contrast to Hearst's focus on strict repetition, Kozima uses a semantic network to provide knowledge about word pairs that are semantically related. Lexical cohesiveness between two words is a function implemented in the network by "activating" the node for one word and observing the "activity value" at the other word after some number of iterations of spreading activation between nodes. The network is trained automatically using a language-specific knowledge source (a dictionary of definitions). Kozima generalizes lexical cohesiveness to apply to a window of text, and plots the cohesiveness of successive text windows in a document, identifying the valleys in the measure as segment boundaries. This approach comes closer to involving nontrivial semantic relationships in boundary placement decisions. A weakness of the approach, however, is that it relies on knowledge that is not focused on any particular domain, and that may not exist in all languages.

A graphically motivated segmentation technique called *dotplotting* is offered by Reynar [Rey94]. This technique uses a simplified notion of lexical cohesion, depending exclusively on word repetition to find tight regions of topic similarity.

2.3 Other feature-based approaches

Litman and Passonneau [LP95] offer an algorithm that uses an existing machine learning program to combine multiple linguistic features extracted from corpora of spoken text, including prosodic and lexical cues. The program outputs a decision tree that can be used to decide whether boundaries exist between prosodic phrases in a test corpus.

The decision tree algorithm, like ours, chooses from a space of candidate features, some of which are similar to our vocabulary questions. Litman's approach, however, is lacking in features related to lexical cohesion and would seem to run the risk of overtraining. Litman uses a limited training set, and it is hard to predict how well the idea would apply to cases in which just the text is available.

3 An exponential model of segmentation

Our attack on the segmentation problem is based on a statistical framework that we call *feature induction* for random fields and exponential models [BDD96, DDL97]. The basic idea behind this approach is to assign a probability distribution to a particular position in the data stream by combining different "features" of the data and assigning each of them a weight in an exponential model. For simplicity, we assume that the features are binary questions. In the case of the segmentation problem for multimedia data where audio, text and video is available, for example, the features might include the answers to such questions as:

- Does the phrase COMING UP appear in the last utterance of the decoded speech?
- Is there a sharp change in the video stream in the last 20 frames?
- Does the language model degrade in performance in the next 2 utterances?
- Is there a "match" between the spectrum of the current image and an image near the last segment boundary?
- Are there blank video frames nearby?
- Is there a sharp change in the audio stream in the next utterance?

The rationale behind expecting such features to be useful stems from the characteristics of coupled image and audio data. For example, in the broadcast news domain, new stories are often introduced with a phrase such as COMING UP... or IN THE NEXT... Segment boundaries are also often, but not always, accompanied by a significant change in the visual scene, as well as in the use of new content words.

The idea of using features is a natural one, and indeed other recent work on segmentation supports this approach, as noted in the previous section. We take a unique approach to incorporating the information inherent in various features, using the statistical framework of exponential models to choose the best features and combine them in a principled manner.

3.1 A short-range model of language

Central to our approach to segmenting is a pair of tools: a short- and long-range model of language. Monitoring the relative behavior of these two models goes a long way towards helping our segmenter sniff out natural breaks in the text. In this section and the next, we describe the short- and long-range models and explain their utility in identifying segments.

The trigram models $p_{\text{tri}}(w \mid w_{-2}, w_{-1})$ we employ use the Katz backof scheme [Kat87] for smoothing. We relied on two different corpora for training these models:

- **WSJ**: A 38-million word collection of Wall Street Journal articles. Here we used a vocabulary of $\mid \mathcal{V} \mid \approx 20,000$ words drawn from words occurring in this corpus.

- **BN**: Approximately 150 million words (four and a half years) of transcripts of various news broadcasts. The broadcasts include CNN news, political roundtables, NPR broadcasts, and interviews. In this case $\mid \mathcal{V} \mid \approx 50,000$.

By restricting the conditioning information to the previous two words, the trigram model is making the simplifying assumption—clearly false—that the use of language one finds in television, radio, and newspaper can be modeled by a second-order Markov process. Although words prior to w_{-2} certainly bear on the identity of w, higher-order models are impractical: the number of parameters in an n-gram model is $O(\mid \mathcal{V} \mid^n)$, and finding the resources to compute and store all these parameters becomes a hopeless task for $n > 3$. Usually the lexical myopia of the trigram model is a hindrance; however, we will see how a segmenter can in fact make positive use of this shortsightedness.

3.2 A long-range model of language

One of the fundamental characteristics of language, viewed as a stochastic process, is that it is highly *nonstationary*. Throughout a written document and during the course of spoken conversation, the topic evolves, affecting local statistics on word occurrences. A model which could adapt to its recent context would seem to offer much over a nonstationary model such as the trigram model. For example, an adaptive model might, for some period of time after seeing a word like `homerun`, boost the probabilities of the words {`pitcher, homerun, fielder, error, batter, triple, out`}. For an empirically-driven example, we provide an excerpt from the **BN** corpus. Emphasized words mark where a long-range language model might reasonably be expected to outperform (assign higher probabilities than) a short-range model.

> Some doctors are more **skilled** at doing the **procedure** than others so it's **recommended** that **patients** ask **doctors** about their track record. People at high **risk** of **stroke** include those over age 55 with a family **history** or high **blood pressure**, **diabetes** and **smokers**. We urge them to be evaluated by their family **physicians** and this can be done by a very simple **procedure** simply by having them test with a **stethoscope** for symptoms of blockage.

One means of injecting long-range awareness into a language model is by retaining a cache of the most recently seen n-grams which is smoothed together (typically by linear interpolation) with the static model; see for example [JMRS91, Kd90]. Another approach, using maximum entropy methods, introduces a parameter for *trigger pairs* of mutually informative words, so that the occurrence of certain words in recent context boosts the probability of the words that they trigger [LRR93].

The method we use here, described in [BBL97], employs a static trigram model as a "prior," or default distribution, and adds certain features to a family of conditional exponential models to capture some of the nonstationary features of text. The features are simple trigger pairs of words chosen on the basis of mutual information. Figure 1 provides a small sample of the (s,t) trigger pairs used in most of the experiments we will describe.

(s,t)	e^λ
residues, carcinogens	2.3
Charleston, shipyards	4.0
microscopic, cuticle	4.1
defense, defense	8.4
tax, tax	10.5
Kurds, Ankara	14.8
Vladimir, Gennady	19.6
Steve, Steve	20.7
education, education	22.2
music, music	22.4
insurance, insurance	23.0
Pulitzer, prizewinning	23.6
Yeltsin, Yeltsin	23.7
Russian, Russian	26.1
sauce, teaspoon	27.1
flower, petals	32.3
casinos, Harrah's	42.8
drug, drug	47.7
Claire, Claire	80.9
picket, scab	103.1

(s,t)	e^λ
recent, recent	2.3
won't, won't	2.6
national, national	3.3
University, University	3.5
notes, notes	5.2
Doo, Chun	6.3
Soviet, Soviet	6.9
fraud, fraud	8.0
detergent, Tide	9.2
Carolco, Hoffman	9.7
Freddie, conventional	10.0
aluminium, smelter	10.4
officers, officers	11.0
records, records	11.5
merger, merger	11.6
proportionate, chances	15.6
nutrasweet, sweetener	18.4
waste, waste	20.7
Belzberg, Belzbergs	31.4
Hutton's, Hutton	64.2

Table 1: On the left is a sample of the 84,694 word pairs from the **BN** domain; on the right is a sample of the 41,263 word pairs from the **WSJ** domain. Roughly speaking, after seeing an "s" word, the empirical probability of witnessing the corresponding "t" in the next N words is boosted by the factor in the third column. In the experiments described herein, $N = 500$.

To incorporate triggers into a long-range language model, we begin by constructing a standard, static backoff trigram model $p_{\text{tri}}(w \mid w_{-2}, w_{-1})$ as described in 3.1. We then build a family of conditional exponential models of the general form

$$p_{\exp}(w \mid H) = \frac{1}{Z(H)} \exp\left(\sum_i \lambda_i f_i(H,w)\right) p_{\text{tri}}(w \mid w_{-2}, w_{-1}) \tag{1}$$

where $H \equiv w_{-N}, w_{-W+1}, \ldots, w_{-1}$ is the word *history* (the N words preceding w in the text), and $Z(H)$ is the normalization constant

$$Z(H) = \sum_w \exp\left(\sum_i \lambda_i f_i(H,w)\right) p_{\text{tri}}(w \mid w_{-2}, w_{-1})\,. \tag{2}$$

The functions f_i, which depend both on the word history H and the word being predicted, are called *features*, and each feature f_i is assigned a weight λ_i. In the models that we built, feature f_i is an indicator function, testing for the occurrence of a trigger pair (s_i, t_i):

$$f_i(H,w) = \begin{cases} 1 & \text{if } s_i \in H \text{ and } w = t_i \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Equation (1) reveals that the probability of a word t involves a sum over all words s such that $s \in H$ (s appeared in the past 500 words) and (s,t) is a trigger pair. One propitious manner of viewing this model is to imagine that, when assigning probability to a word w following a history of words H, the model "consults" a cache of words which appeared in H and which are the left half of some s,t trigger pair. In general, the cache consists of content words s which promote the probability of their mate t, and correspondingly demote the probability of other words. As described in [BBL97], for each (s,t) trigger pair there corresponds a real-valued parameter λ; the probability of t is boosted by a factor of e^{λ} for W words following the occurrence of s_i.

The training algorithm we use for estimating λ values is the *Improved Iterative Scaling* algorithm of Della Pietra et al. [DDL97], which is a maximum-likelihood scheme. Assuming robust estimates for the λ parameters, the resulting model is essentially guaranteed to be superior to the trigram model.

For a concrete example, if $s_i =$ `Vladimir` and $t_i =$ `Gennady`, then $f_i = 1$ if and only if `Vladimir` appeared in the past N words and the current word w is `Gennady`. Consulting Table 1, we see that in the **BN** corpus, the presence of `Vladimir` will boost the probability of `Gennady` by a factor of 19.6 for the next $W = 500$ words.

3.3 Incorporating "relevance" features

A long-range language model such as that described in Section 3.2 uses selected words from the past ten, twenty or more sentences to inform its decision on the possible identity of the next word. This is likely to help if all of these sentences are in the same document as the current word, for in that case the model has presumably begun to adapt to the idiosyncracies of the current document. In the case of the trigger model described above, the cache will be filled with "relevant" words. In this setting, one would expect a long-range model to outperform a trigram (or other short-range) model, which doesn't avail itself of long-range information.

On the other hand, if the present document has just recently begun, the long-range model is wrongly conditioning its decision on information from a different—and presumably unrelated—document. A soap commercial, for instance, doesn't benefit a long-range model in assigning probabilities to the words in the news segment following the commercial. Often a long-range model will actually be misled by such irrelevant context; in this case, the myopia of the trigram model is actually helpful.

Roughly speaking, then, the idea is this: by monitoring the long- and short-range models, one might be more inclined towards a partition when the long-range model suddenly shows a dip in performance—a lower assigned probability to the words seen—compared to the short-range model. Conversely, when the long-range model is consistently assigning higher probabilities to the observed words, a partition is less likely.

This motivates a quantitative measure of instantaneous "relevance," which we define as the difference of the log-probability assigned to the next word (or sentence) by the the exponential model, and the log-probability assigned by the short-range trigram model:

$$R_I(w) \equiv \log \left(\frac{p_{\text{exp}}(w \mid H)}{p_{\text{tri}}(w \mid w_{-2}w_{-1})} \right) . \tag{4}$$

When the exponential model outperforms the trigram model, $R_I > 0$. If we observe the behavior of R_I as a function of the position of the word within a document, we find that on average R_I slowly increases from below zero to well above zero. Figure 1 illustrates this phenomenon. This behavior is consistent with our earlier intuition: the cache of the long-range model is destructive early in a document, when the new content words bear little in common with the content words from the previous article. Gradually, as the cache fills with words drawn from the current article, the long-range model gains steam and R_I improves.

Raw instantaneous relevance is a useful feature for segmentation, but it can be significantly improved. In particular, there are certain trigram contexts in which the trigger feature cache is likely to pay off, and certain contexts in which the trigram model alone provides all of the necessary predictive information. Often this distinction is a domain-specific matter of style. For instance, in the *Wall Street Journal* an occurrence of MR. or MRS. often indicates that a proper name triggered earlier in the document will follow. Similarly, SHAREHOLDERS OF... predicts a company name that has already been introduced, and LOS predicts the second half of a Spanish city name tied closely with the topic. Other contexts are tightly bound (with low entropy) to a small set of possible successor words in a way that is independent of the long-range history. Computing the

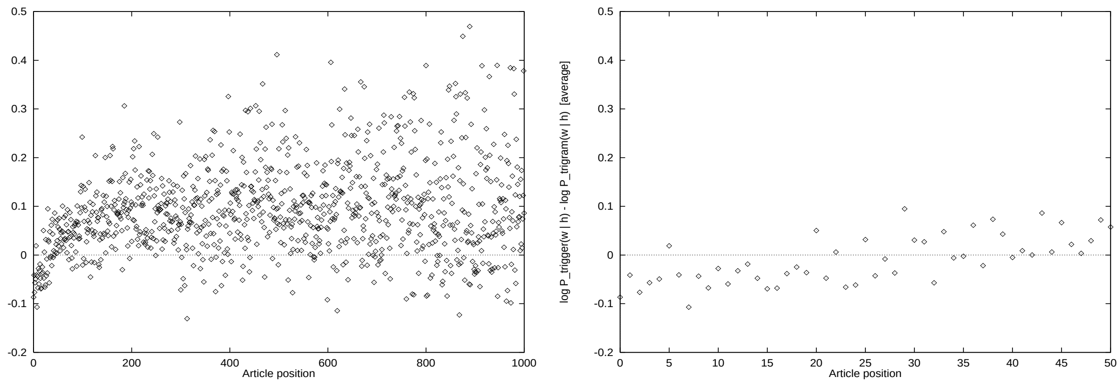


Figure 1: Near the beginning of a document, an adaptive, long-range language model is on average less accurate than a static trigram model. The left figure plots the log of the ratio of the two models as function of relative position in the document. The right figure shows this trend for the first 50 words. Each figure is constructed from statistics on 240 articles of test data.

MR.	1.402986
MS.	1.134188
MRS.	1.063191
DOCTOR	0.874050
WINTER	0.799196
EL	0.707947
SENATOR	0.696035
⋮	⋮
FIVE	-0.525790
FREEMAN	-0.556112
EIGHTEEN	-0.569623
ZERO	-0.600722
FAIRCHILD	-0.614652
ROW	-0.688912
BILLION	-0.722252

Table 2: The top and bottom few unigram contexts in terms of average instantaneous relevance in the training data. Note that the top contexts are *introductory* in nature.

"context relevance", the expected value of the instantaneous relevance for a given context, for all one- and two-word contexts in the training data gives us a ranking excerpted in Table 3.3. For bigram contexts (w_{-2}, w_{-1}), the context-relevance $R_C(w_{-2}, w_{-1})$ is as follows:

$$R_C(w_{-2}, w_{-1}) = \sum_{w \in \mathcal{V}} \widetilde{p}(w_{-2}, w_{-1}, w) \log \left(\frac{p_{\text{exp}}(w|w_{-2}, w_{-1})}{p_{\text{tri}}(w|w_{-2}, w_{-1})} \right) , \tag{5}$$

where $\widetilde{p}$ is the empirical probability distribution.

Normalizing instantaneous relevance by the context-dependent expectation above gives us a final relevance indicator R_i (where i is a position in the corpus) that we can incorporate into our segmentation model. As we assume that sentence boundaries are provided in this presentation, the questions we ask are actually about the relevance score assigned to entire sentences (normalized by sentence length), a geometric mean of relevance ratios. Within our feature induction framework, we chose to include the relevance questions described in Figure 3.3 as candidate features in the model.

3.4 Vocabulary features

In addition to the estimate of instantaneous topicality that relevance features provide, we included features pertaining to the identity of words before and after potential segment boundaries as candidates in our exponential model. As explained in Section 6, these features capture the domain-specific conventions and habits of introduction and conclusion. For each domain, and for each word in the vocabulary of that domain, we added candidate questions listed in Table 3.4.

In Figure 3 we present the overall dataflow used to train our exponential model of segmentation.

$-0.001 < R_{i+1} < 0$	$0 < R_{i+1} < 0.001$
$-0.003 < R_{i+1} < 0$	$0 < R_{i+1} < 0.003$
$-0.005 < R_{i+1} < 0$	$0 < R_{i+1} < 0.005$
$-0.010 < R_{i+1} < 0$	$0 < R_{i+1} < 0.010$
$-0.030 < R_{i+1} < 0$	$0 < R_{i+1} < 0.030$
$-0.050 < R_{i+1} < 0$	$0 < R_{i+1} < 0.050$
$-0.100 < R_{i+1} < 0$	$0 < R_{i+1} < 0.100$
$-0.300 < R_{i+1} < 0$	$0 < R_{i+1} < 0.300$
$-0.500 < R_{i+1} < 0$	$0 < R_{i+1} < 0.500$
$-1.000 < R_{i+1} < 0$	$0 < R_{i+1} < 1.000$
$R_{i+1} \leq -1.000$	$R_{i+1} \geq 1.000$
$(R_{i+1} + R_{i+2})/2 \leq 0$	
$(R_{i+1} + R_{i+2} + R_{i+3})/3 \leq 0$	
$(R_{i+1} + R_{i+2} + \ldots + R_{i+4})/4 \leq 0$	
$(R_{i+1} + R_{i+2} + \ldots + R_{i+5})/5 \leq 0$	
$(R_{i+1} + R_{i+2} + \ldots + R_{i+10})/10 \leq 0$	
$(R_{i+1} + R_{i+2} + \ldots + R_{i+5})/5 \leq -0.01$	
$(R_{i+1} + R_{i+2} + \ldots + R_{i+10})/10 \leq -0.01$	
$(R_{i+1} + R_{i+2} + \ldots + R_{i+5})/5 \leq -0.05$	
$(R_{i+1} + R_{i+2} + \ldots + R_{i+10})/10 \leq -0.05$	

Figure 2: Questions about relevance included as candidate features in the segmentation model. R_i is the relevance score of sentence number i in the corpus. Questions asked at sentence i are used to predict whether a document boundary occurs between sentences i and $i+1$.

Does the word appear up to 1 sentence in the future? 2 sentences? 3? 5?
Does the word appear up to 1 sentence in the past? 2 sentences ? 3? 5?
Does the word appear up to 5 sentences in the past but not 5 sentences in the future?
Does the word appear up to 5 sentences in the future but not 5 sentences in the past?
Does the word appear up to 1 word in the future? 5 words?
Does the word appear up to 1 word in the past? 5 words?
Does the word begin the preceding sentence?

Table 3: Vocabulary features

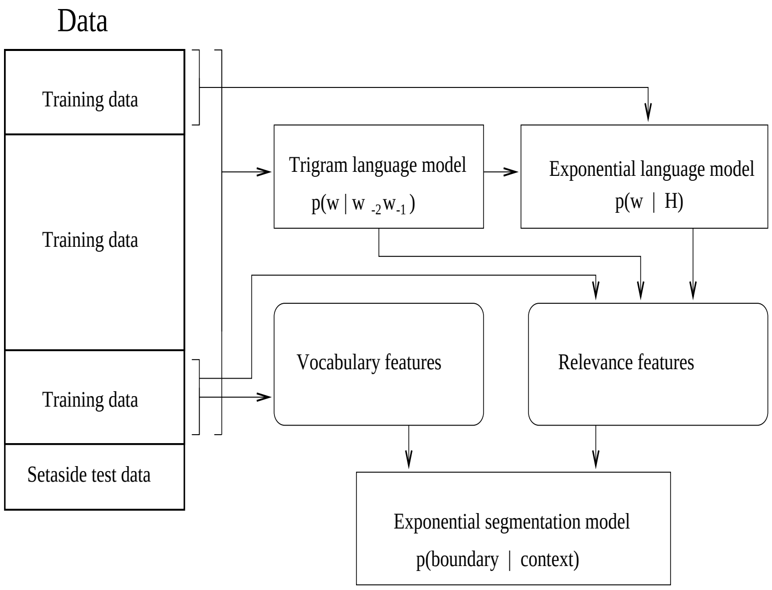


Figure 3: Data flow in training the exponential segmentation model

4 Feature Induction

4.1 The feature induction algorithm

To cast the problem of determining segment boundaries in statistical terms, we set as our goal the construction of a probability distribution $q(b\,|\,\omega)$, where $b \in \{\text{YES}, \text{NO}\}$ is the value of a random variable describing the presence of a segment boundary in context ω. We consider distributions in the *linear exponential family* $\mathcal{Q}(f, q_0)$ given by

$$\mathcal{Q}(f, q_0) = \left\{ q(\cdot\,|\,\omega) \;:\; q(b\,|\,\omega) = \frac{1}{Z_\lambda(\omega)} e^{\lambda \cdot f(\omega)}\, q_0(b\,|\,\omega) \right\} \tag{6}$$

where $q_0(b\,|\,\omega)$ is a prior or *default* distribution on the presence of a boundary, and $\lambda \cdot f(\omega)$ is a linear combination of binary features $f_i(\omega) \in \{0, 1\}$ with *feature parameters* $\lambda_i \in \mathbf{R}$:

$$\lambda \cdot f(\omega) = \lambda_1 f_1(\omega) + \lambda_2 f_2(\omega) + \cdots \lambda_n f_n(\omega)\,. \tag{7}$$

The normalization constants

$$Z_\lambda(\omega) = 1 + e^{\lambda \cdot f(\omega)} \tag{8}$$

insure that this is indeed a family of conditional probability distributions. (This family of models is closely related to the class of sigmoidal belief networks [Nea92].)

Our judgement of the merit of a model $q \in \mathcal{Q}(f, q_0)$ relative to a reference distribution $p \notin \mathcal{Q}(f, q_0)$ during training is made in terms of the Kullback-Leibler divergence

$$D(p\,\|\,q) = \sum_{\omega \in \Omega} p(\omega) \sum_{b \in \{\text{YES,NO}\}} p(b\,|\,\omega) \log \frac{p(b\,|\,\omega)}{q(b\,|\,\omega)}\,. \tag{9}$$

Thus, when p is chosen to be the empirical distribution of a sample of training events $\{(\omega, b)\}$, we are using the maximum likelihood criterion for model selection. Under certain mild regularity conditions, the maximum likelihood solution

$$q^{\star} = \underset{q \in \mathcal{Q}(f, q_0)}{\arg\min}\; D(p \,\|\, q) \tag{10}$$

exists and is unique. To find this solution, we use the iterative scaling algorithm presented in [DDL97].

This explains how a model is chosen once we know the features $f_1, \ldots, f_n$, but how are these features to be found? The procedure that we follow is a greedy algorithm akin to growing a decision tree. Given an initial distribution q and a set of candidate features $\mathcal{C}$, we consider the one-parameter family of distributions $\{q_{\alpha,g}\}_{\alpha \in \mathbf{R}} = \mathcal{Q}(g, q)$ for each $g \in \mathcal{C}$. The *gain* of the candidate feature g is defined to be

$$G_q(g) = \operatorname{argmax}_{\alpha} \left(D(\tilde{p} \,\|\, q) - D(\tilde{p} \,\|\, q_{\alpha,f}) \right) . \tag{11}$$

This is the improvement to the model that would result from adding the feature g and adjusting its weight to the best value. After calculating the gain of each candidate feature, the one with the largest gain is chosen to be added to the model, and all of the models parameters are then adjusted using iterative scaling. In this manner, an exponential model is incrementally built up using the most informative features.

4.2 Example: flipping coins

A simple example of feature induction may help to explain some of its subtleties which arise in the segmentation applications we present in the following sections.

Suppose that we flip a biased coin, whose probability of heads is $p(\mathtt{H}) = \frac{2}{3}$ and whose entropy is thus $H(\frac{2}{3}) \approx 0.918$ bits. Depending on the outcome of this random variable, we answer 50 questions in the following manner. For the i-th question, with probability $\left(\frac{9}{10}\right)^i$ the answer is YES if the coin came up heads and NO if the coin came up tails. With probability $1 - \left(\frac{9}{10}\right)^i$ the answer to the question is chosen uniformly at random. We wrote a little program to generate such events; a sample of 10 of them is shown below.

```
H 11111110101010100001100110110110101101000000001111
T 00100100010000110001111111110001000111100110111100
H 11111011011000100110101100100101110010101111100101
H 11111111100000011110001010111100101001100000110010
T 00001000000000001000100110001000101111110110101000 0
H 11111111110110111100111111000111111111110000010011 1
H 11100111111111111100101010101010101101000111100100
T 00001000110011101010001000010001011101111010001111
H 11011010110111101000101000011111111101100010101101
T 00100010011010100010000101101001001000111000000101
```

To learn the posterior distribution $p(b \,|\, \omega = \omega_1 \cdots \omega_{50})$ of the coin given the bitstrings, we carried out feature induction on 100 synthetic events of this form. The results are tabulated in the chart below, which shows the first five induced features. For each feature, the gain that resulted from adding the feature is shown, together with the cross entropy of the model after the feature was added, and the initial and final values of the parameter $\beta = e^{\lambda}$.

feature	gain	entropy	initial β	final β
bit 1	0.563	0.435	41.5	54.7
bit 32	0.114	0.315	0.20	0.16
bit 35	0.024	0.280	0.45	0.29
bit 2	0.026	0.246	2.93	2.65
bit 16	0.035	0.206	0.34	0.25
bit 5	0.023	0.177	2.15	2.15

Table 4: Statistics on the first five features induced for the toy coin-flipping example. The gain is the reduction in entropy after the feature is added. The final value of the parameter $\beta = e^{\lambda}$ after all five features are added and iterative scaling is carried out.

The first feature chosen is, naturally, the first bit. Out of the 100 events that were generated, it happened that the first bit disagreed with the coin only four times, and among these events only one of the coin tosses was a tail. Thus, the first feature constrains the conditional probability of heads to be $p(\mathtt{H} \,|\, \omega, \omega_1 = 1) = 0.99$. After five iterations of iterative scaling training, the model probability is $p(\mathtt{H} \,|\, \omega, \omega_1 = 1) = \frac{41.5}{1+41.5} \approx 0.98$. After the first feature then, the cross entropy of the model with respect to the true distribution is $\frac{2}{3}H(0.98) + \frac{1}{3}H(0.5) \approx 0.427$ bits, and the cross entropy with respect to the actual 100 events turns out to be 0.435 bits.

The second feature chosen queries the 32nd bit. The weight λ initially used for this feature is $\lambda = \log(0.20) \approx -1.61$. As a consequence, the effect of this feature is to lower the probability of heads when the 32nd bit is set. This a good idea since after the first feature was added, roughly one third of the time the distribution is still uniform on heads and tails. Since $\left(\frac{9}{10}\right)^{32} \approx 0.034$, this bit is not well correlated with the coin, and reducing the probability of heads when it is set improves the model. The third feature, which queries the 35th bit, similarly lowers the probability of heads. At this point the probability of heads has been pushed too far down, due to the overlap of events where the first, 32nd, and 35th bits are set. To compensate for this, the model chooses to query the second bit, and thereby reestablish the proper distribution on heads.

Similar effects are seen to be at play in the feature induction results that we present for the segmentation problem in the following sections. For the details on feature induction and other examples of it in action, we refer to the papers [BDD96, DDL97]. The discussion in [DDL97] also explains how the feature induction algorithm generalizes decision trees. While decision trees recursively partition the training data, the features in an exponential model can be overlapping, so that the scheme is much less prone to overfitting. This is an important distinction when drawing inferences from text, where the sparse data problem is typically so severe. Our experiments with segmentation bear this out since good results are obtained after only a handful of features are found, with no fussing over stopping, pruning, or smoothing. While these issues are certainly relevant, they are not the primary considerations in obtaining useful models.

5 A probabilistically motivated error metric

Precision and recall statistics are commonly used in information retrieval to assess the quality of algorithms. For the segmentation task they are used to gauge how frequently boundaries actually occur when they are hypothesized and vice versa. Though they have snuck into the literature in this disguise, we believe they are unwelcome guests.

A useful error metric should somehow correlate with the utility of the instrumented procedure in a real application. In almost any conceivable application, a segmenting tool that consistently comes close—off by a sentence, say—is preferable to one which places boundaries willy-nilly. Yet an algorithm that places a boundary a sentence away from the actual boundary every time actually receives *worse* precision and recall scores than an algorithm which hypothesizes a boundary at every position. We believe that in a segmenter, close should count for something.

A useful metric should also be robust with respect to the scale (words, sentences, paragraphs, for instance) at which boundaries are determined. However, precision and recall are scale-dependent quantities. [Rey94] uses an error window that redefines "correct" to mean hypothesized within some constant window of units away from a reference boundary, but this approach still suffers from overdiscretizing error, drawing all-or-nothing lines insensitive to gradations of correctness.

Finally, a useful metric should be a single number. A commonly-cited flaw of the precision/recall figures is their complementary nature: hypothesizing more boundaries raises precision at the expense of recall, allowing an algorithm designer to tweak parameters to trade precision for recall. One proposed work-around is to employ dynamic time warping to come up with an explicit alignment between the segments proposed by the algorithm and the reference segments, and then to combine insertion, deletion, and substitution errors into an overall penalty. This error metric, in common use in speech recognition, can be achieved by a similar Viterbi search. Though the result is indeed a single figure of merit, this number doesn't seem to correspond to anything meaningful in the real world of applications.

Our proposed metric satisfies the listed desiderata. It formalizes in a probabilistic manner the effect of document co-occurrence on goodness, in which it is deemed desirable for related units of information to appear in the same document and unrelated units to appear in separate documents.

5.1 A proposed new error metric

Segmentation, whether at the word or sentence level, is about identifying boundaries between successive units of information in a text corpus. Two such units are either related or unrelated by the intent of the document author. A natural way to reason about developing a segmentation algorithm is therefore to optimize the likelihood that two such units are correctly labeled as being related or being unrelated. Our error metric E_D is simply the *probability that two sentences drawn randomly from the corpus are correctly identified as belonging to the same document or not belonging to the same document.* More formally, for a corpus n sentences long,

$$E_D = \sum_{i=1}^{n} \sum_{j=i}^{n} D(j-i) \cdot (\delta_{\text{ref}}(i,j) \overline{\oplus} \delta_{\text{hyp}}(i,j)) \tag{12}$$

Here δ_{ref} is an indicator function which is 1 if the two corpus indices specified by its parameters belong in the same document, and 0 otherwise; similarly, δ_{hyp} is 1 if the two indices are hypothesized to belong in the same document, and 0 otherwise. The $\overline{\oplus}$ operator is the `XNOR` function ("both or neither") on its two operands. The parameter D is a *distance probability distribution* over the set of possible distances between sentences chosen randomly from the corpus. If D is uniform over the length of the text, then the metric represents the probability that any two sentences drawn from the corpus are correctly identified as being in the same document or not.

Consider the implications of this for information retrieval. Suppose there is precisely one sentence in a target corpus that satisfies our information demands. For some applications it may be sufficient for the system to return only that sentence, but in general we desire that it return as many sentences directly related to the target sentence as possible, without returning too many unrelated sentences. If we assume "related" to mean "contained in the same document", then our error metric judges algorithms based on how often this happens.

In practice letting D be the uniform distribution is unreasonable, since for large corpora most randomly drawn pairs of sentences are in different documents and are correctly identified as such by even the most naive algorithms. We instead adopt a distribution that focuses on small distances. In particular we choose D to be an exponential distribution with mean μ, a parameter that we fix at the approximate mean document length for the domain:

$$D(k) = \frac{e^{-k\mu}}{Z} \tag{13}$$

In the above, Z is a normalization chosen so that D is a probability distribution over the range of distances it can accept.

There are several sanity checks that validate the use of our metric. Firstly, since the measure is a probability and therefore a real number between 0 and 1, we expect 1 to represent perfection; indeed, an algorithm scores 1 with respect to some data if and only if it predicts its segmentation exactly. It captures the notion of nearness in a principled way, gently penalizing algorithms that hypothesize boundaries that aren't quite right, and scaling continuously down with the algorithm's degradation. Furthermore, it is not possible to "cheat" with this metric: spurious behavior such as never hypothesizing boundaries and hypothesizing nothing but boundaries are severely penalized.

6 Experimental Results

In this section we present the results we have obtained using exponential models for segmentation.

For the **WSJ** experiments we ran, a total of 299,771 candidate features were available to the induction program. The training was carried out on only 325,000 words of data, although the trigram prior was trained on 38 million words and the trigger parameters were trained on approximately one million words of text.

We first present a sample of some of the features the algorithm uncovers. Our objective is to give the reader some intuition for what the induction process decides to be important for segmenting text. We emphasize that the process is completely automatic once the set of candidate features has been selected.

In Figure 4, the word or relevance score for each feature is indicated, together with the value of β for the feature after iterative scaling is complete for the final model. The $\vdash\!\!- \quad -\!\!\dashv$ figures indicate features that are active over a range of sentences. Thus, the symbol $\vdash\!\!\longrightarrow \overset{\text{MR.}}{0.07} \overset{+1}{\longrightarrow\!\!\dashv}$ represents the feature "Does the word MR. appear in the next sentence?" which, if true, contributes a factor of $\beta = 0.07$ to the exponential model. Similarly, the $\bullet\!\!- \quad -\!\!\bullet$ figures represent features that are active over a range of words. For example, the figure $\bullet\!\!\longrightarrow \overset{\text{HE}}{0.08} \overset{+5}{\longrightarrow\!\!\bullet}$ represents the question "Does the word HE appear in the next five words?" which is assigned a weight of 0.08. The symbol $\overset{-5}{\vdash\!\!\longrightarrow} \text{SAID} \longrightarrow\!\!\dashv\!\vdash\!\!\longrightarrow \overset{\neg\,\text{SAID}}{2.7} \overset{+5}{\longrightarrow\!\!\dashv}$ stands for a feature which asks "Does the word SAID appear in the previous five sentences but *not* in the next five sentences?" and contributes a factor of 2.7 if the answer is yes.

Several interesting things can be noticed in this set of features. Clearly the model quickly picks up on some of the idiosyncracies of the data. Many of the articles concern some piece of news about a particular corporation. The formal name of the company is often introduced at the

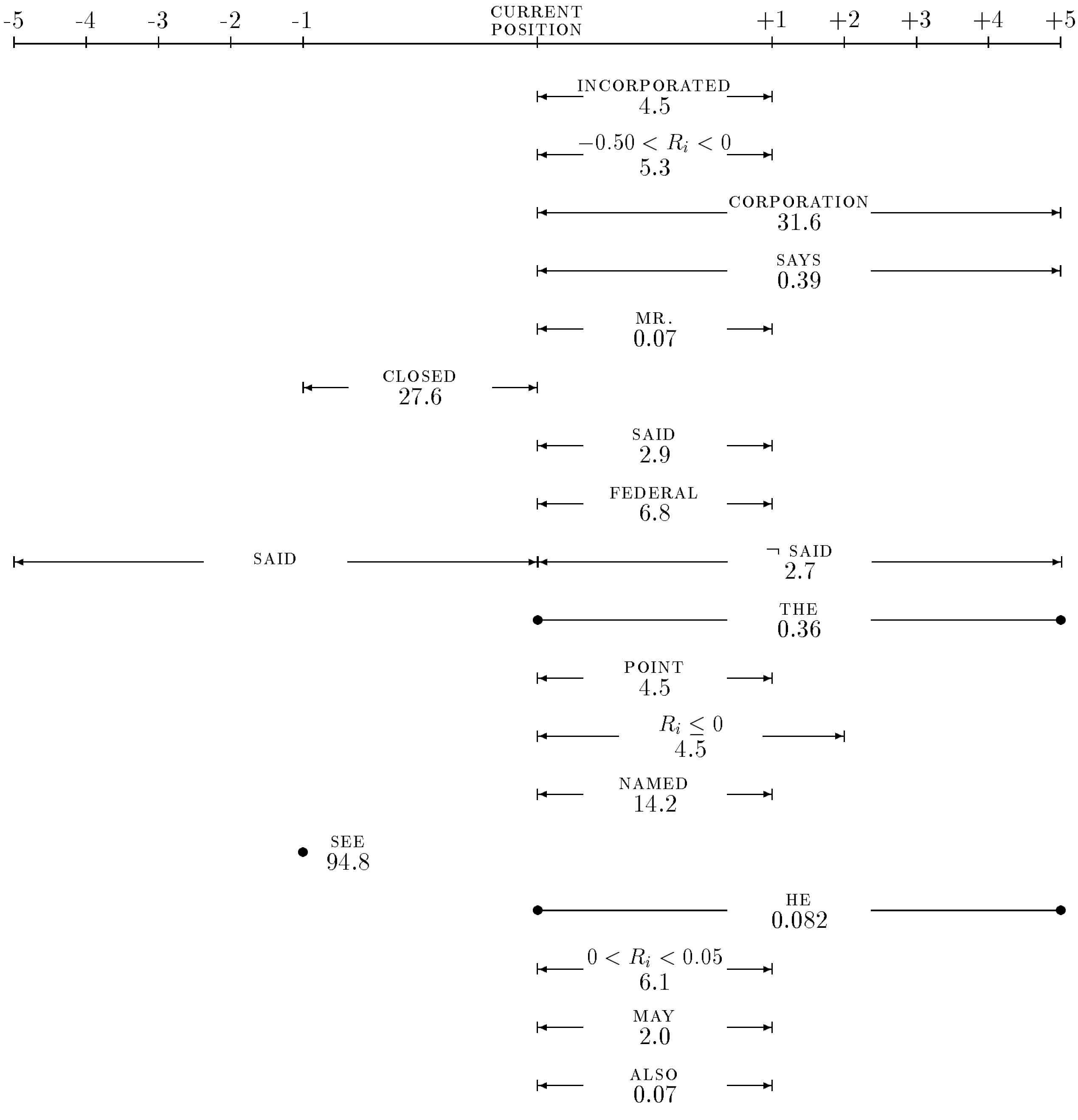


Figure 4: First several features induced for **WSJ** corpus, presented in order of selection, with β factors underneath. The size of the bars indicate active range of the feature, in words or sentences, relative to the current position in the data. The details are given in the text.

beginning of the article, and thus the word INCORPORATED within the first sentence is the first feature the induction process chooses. The second feature uses the relevance statistic. If the trigger model is bad relative to the trigram model in the following sentence, this feature contributes a factor of 5.3 to the probability of a segment boundary. The fifth feature asks about the presence of the word MR. In hindsight, we can explain this feature by noting that in **WSJ** data the style is to introduce a person in the beginning of an article by writing, for example, WILE E. COYOTE, PRESIDENT OF ACME INCORPORATED... and then later in the article using a shortened form of the name: MR. COYOTE CITED LACK OF EXPLOSIVES... Thus, the presence of MR. contributes an unambivalent discount of 0.07 for an article boundary. Upon inspecting some of the data one finds that the appearance of the word CLOSED in the previous sentence is a strong indication of an article boundary since articles often end with a statement of a company's performance on the stock market during the day of the story of interest. Similarly, the end of an article is often made with an invitation to visit a related story, hence a sentence beginning with SEE boosts the probability of a segment boundary by a large factor of 94.8. Since a pronoun typically requires an antecedent, the presence of HE among the first words is a sign that the current position is *not* near an article boundary, and this feature therefore has a discounting factor of 0.082.

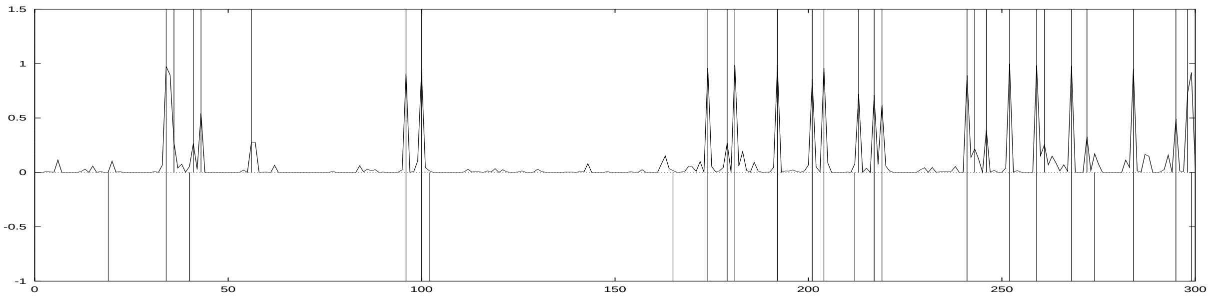


Figure 5: Performance of the algorithm on the first 300 test sentences. The lower dashed lines indicate reference segmentations ("truth"). The upper dashed lines are boundaries placed by the algorithm. The fluctuating curve is the probability of segment boundary according to the exponential model after 70 features were induced.

After the first 70 features were induced, a simple decision procedure was used for actually placing boundaries: a segment boundary was placed at each position for which the model probability was above a fixed threshold. The threshold was determined on heldout data in order to maximize the probability E_D, and turned out to be 0.20 for the **WSJ** corpus, and 0.18 for the **BN** corpus.

We now present examples of the segmentation algorithm at work on previously unseen test data. In Figure 5 we show the performance of the algorithm on the first 300 sentences in the 325,000 word test corpus. One of the first things to notice is that many of the segments are quite short, adding special difficulties for the segmentation problem. For example, here is one article, in its entirety, from the data:

> R. J. REYNOLDS TOBACCO U. S. A. THE DOMESTIC TOBACCO UNIT OF R. J. R. NABISCO INCORPORATED SAID IT SOLD ITS TOP TOBACCO BRANDS TO REPUBLIC TOBACCO INCORPORATED CHICAGO. TERMS WEREN'T DISCLOSED. REPUBLIC TOBACCO DISTRIBUTES SMOKING TOBACCO AND CIGARETTE PAPERS. THE SALE COMPLETES R. J. R.'S PREVIOUSLY REPORTED PLAN TO SELL OFF ITS NON CIGARETTE SMOKING TOBACCO BRANDS. R. J. R. A FOOD AND TOBACCO CONCERN EARLIER THIS YEAR SOLD ITS PRINCE ALBERT CARTER HALL APPLE ROYAL COMFORT AND WINCHESTER LITTLE CIGAR BRANDS.

If the segmenter were to cut off the article even one sentence too early, it would be "close but no cigar." (It turns out that the segmenter we present results for here successfully found the end of this article, but not the beginning.)

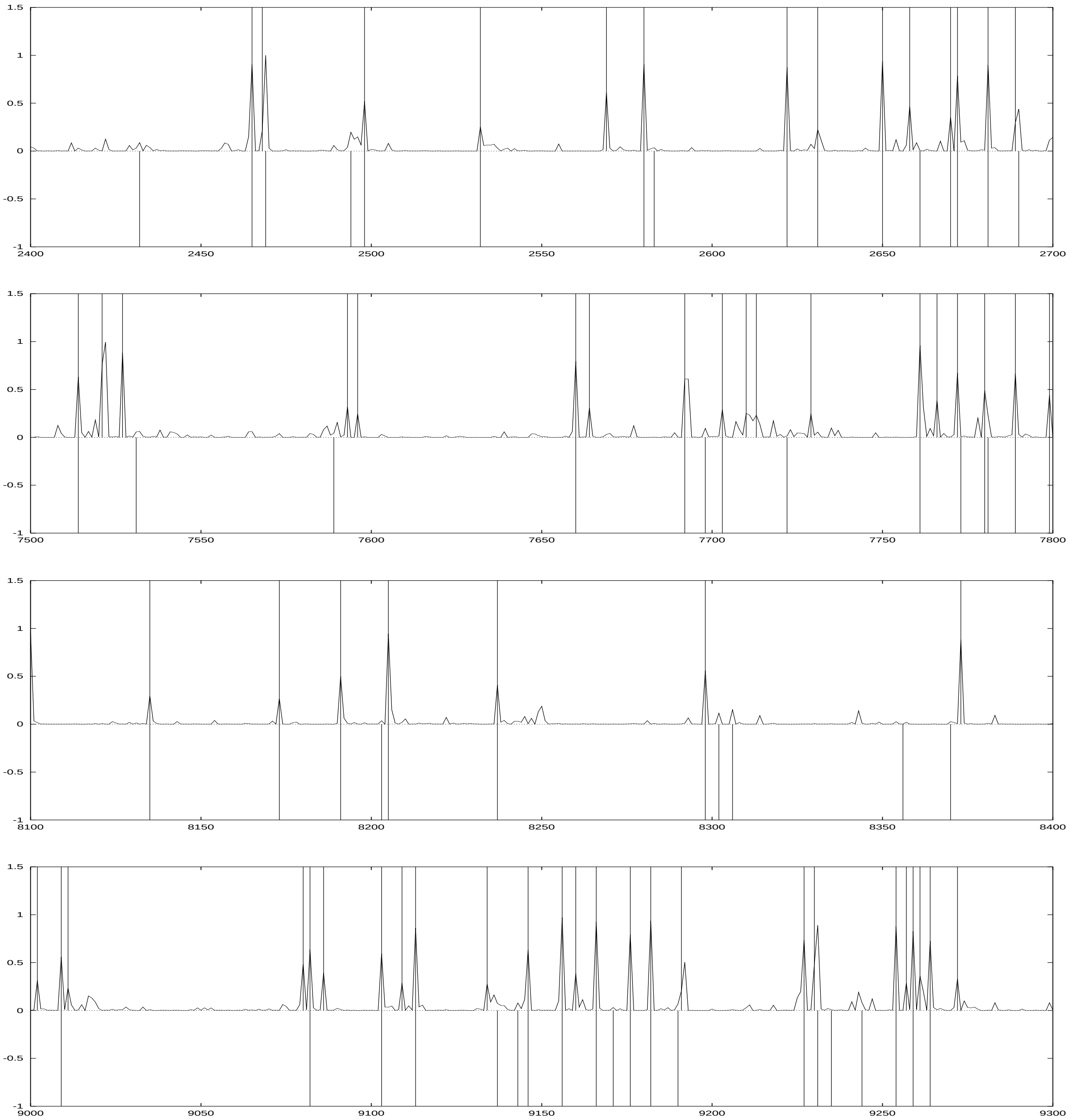
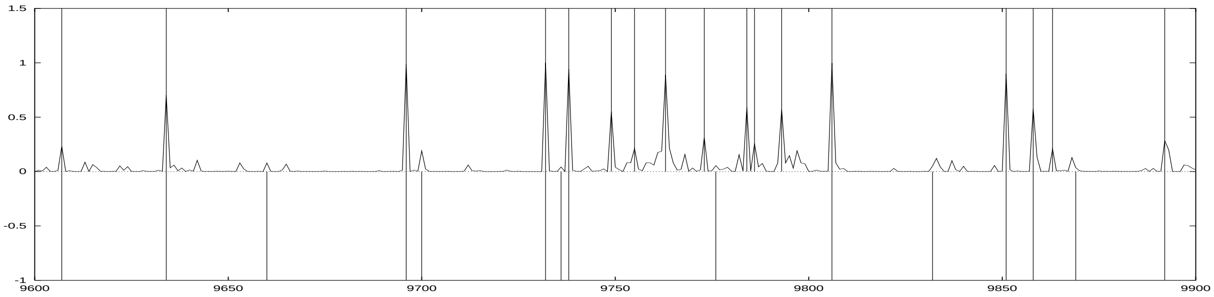


Figure 6: Other typical sequences of 300 sentences.

Other typical examples are presented in Figure 6. In Figure 7 we look more closely at a region in the test data where the automatic segmenter seems to have become confused. In the neighborhood of sentence 9750 the algorithm proposes many segment boundaries with high probability, none of which appear in the reference transcripts. This is understandable once we look at the text, which reveals an "article" containing a hodge-podge of brief news summaries of unrelated stories.

We hasten to add that these results were obtained with no smoothing or pruning of any kind, and with only 70 features induced from the candidate set of 299,771 word and relevance features. We expect that better results can be obtained by simply training on much more data, and also by



REAGAN AIDES SAID MEESE WON'T BE CHARGED FOR HIS ROLE IN A PIPELINE PLAN. ACCORDING TO ADMINISTRATION AND LAW ENFORCEMENT OFFICIALS THE ATTORNEY GENERAL'S INVOLVEMENT IN A PRIVATE MIDEAST PIPELINE PROPOSAL MAY PRESENT MORE POLITICAL THAN LEGAL PROBLEMS. THOSE FAMILIAR WITH THE INVESTIGATION SAID INDEPENDENT COUNSEL MCKAY DOESN'T HAVE ENOUGH EVIDENCE TO SEEK CHARGES AGAINST MEESE AND <UNK> OWN STATEMENT FRIDAY THAT HE HOPES TO WRAP UP THE INVESTIGATION WITHIN THE NEXT FEW WEEKS SUGGESTS THAT MEESE PROBABLY WON'T FACE SUCH ACTION. A MEMO SENT TO MEESE FROM A FRIEND IN NINETEEN EIGHTY FIVE ALLEGEDLY PROPOSED PAYOFFS TO ISRAELI POLITICIANS. FOREIGN MINISTER PERES DENIED BEING OFFERED A BRIBE. ISRAEL <UNK> A CURFEW ON <UNK> AS CLASHES CONTINUED IN OCCUPIED AREAS. THE ISRAELI ARMY PLACED A CURFEW ON THE CITY THE LARGEST IN THE WEST BANK AFTER LENGTHY STREET BATTLES WITH ARAB YOUTHS. AUTHORITIES SAID AT LEAST FOUR PALESTINIANS WERE WOUNDED BY GUNFIRE. THE ARMY ALSO SAID THAT A <UNK> TOSSED AT A CAR IN THE CITY OF <UNK> CRITICALLY BURNED A JEWISH <UNK> [...] A CHURCH AND AS MANY AS FIVE HUNDRED HOMES WERE BURNED TO THE GROUND AT A <UNK> CAMP NEAR CAPE TOWN'S AIRPORT AND RESIDENTS ACCUSED THE POLICE OF AIDING <UNK>. THE WASHINGTON <UNK> WON PROFESSIONAL <UNK> SUPER BOWL DEFEATING THE DENVER <UNK> FORTY TWO TO TEN IN SAN DIEGO. DIED JAMES R. <UNK> JUNIOR EIGHTY THREE FIRST SCIENCE ADVISER AT THE WHITE HOUSE AND EX CHAIRMAN OF THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY FRIDAY IN CAMBRIDGE MASSACHUSETTS.

Figure 7: Excerpt of text for confusion region near sentence 9750. A quick perusal of this segment reveals that it is a collection of very brief summaries of a diverse range of news stories.

corpus	test corpus size (sentences)	hypothesized segments	reference segments	E_D
WSJ	13541	792	757	83.1%
BN	18943	227	251	80.2%

Table 5: Test results: article counts and E_D values for **WSJ** and **BN**. The mean document length μ is approximately 18 sentences for **WSJ** and 75 sentences for **BN**.

using cross-validation or other standard statistical techniques to determine the best stopping point, but we have not considered such refinements necessary for our initial results. Unlike many other machine learning methods, feature induction for exponential models is quite robust to overfitting since the features act in concert to assign probability to events rather than splitting the event space and assigning probability using relative counts.

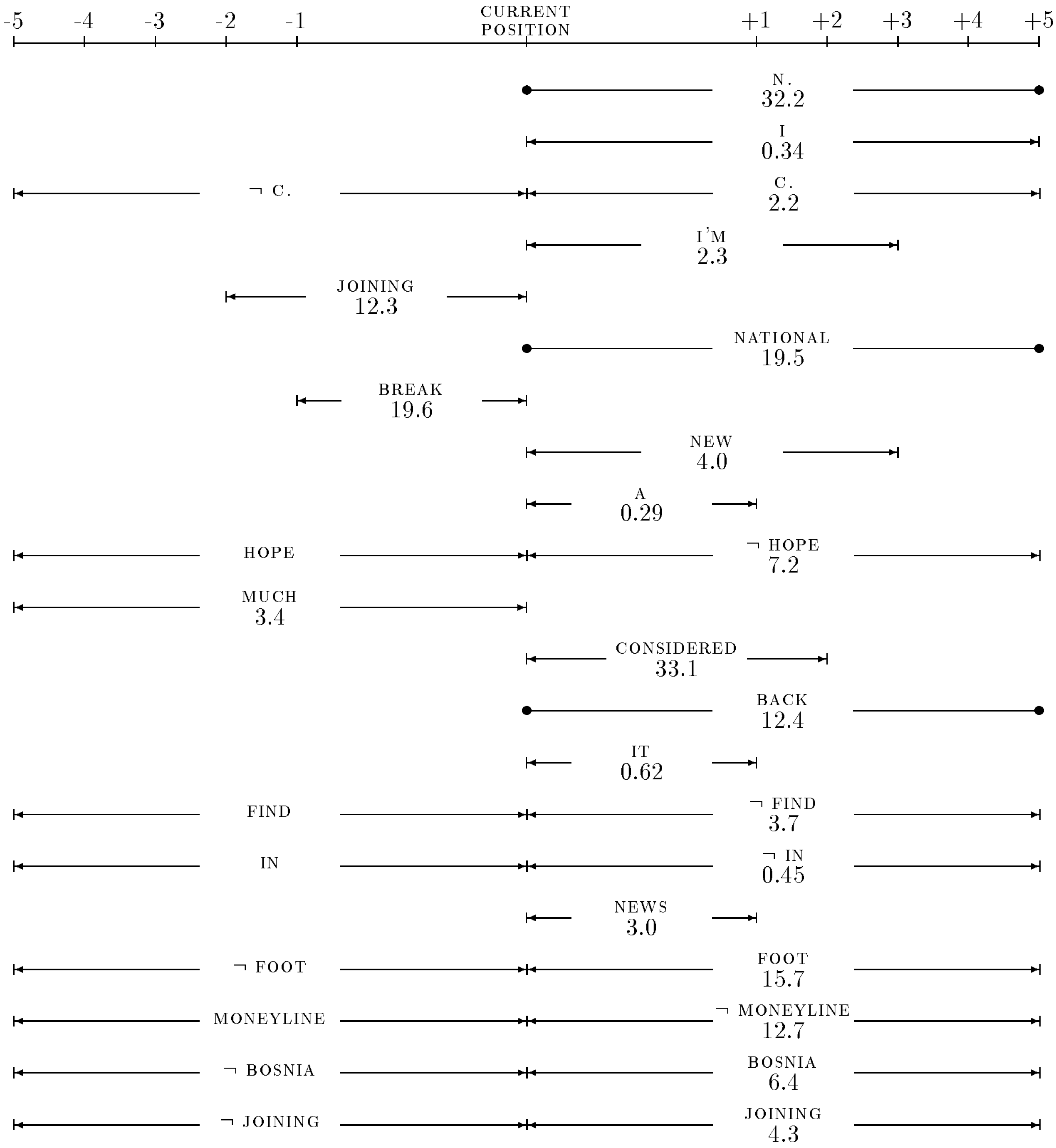


Figure 8: First several features induced for **BN** corpus, presented in order of selection, with β factors underneath.

The quantitative results are similar for the **BN** data, but through different means. In Figure 8

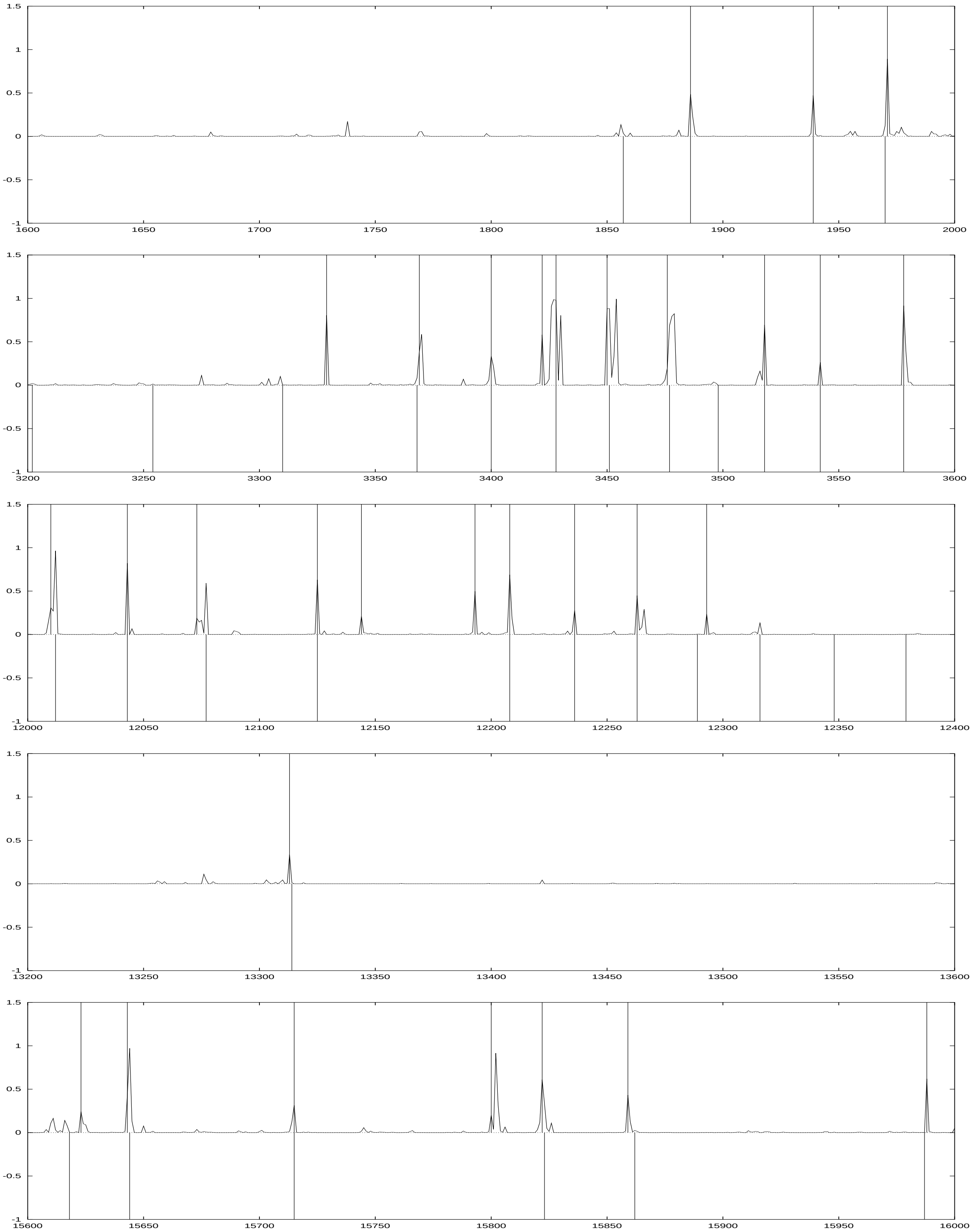


Figure 9: Example segmentations from the **BN** test data.

we show the first several features that the induction algorithm chooses when trained on 325,000 words of broadcast news data. There is a conspicuous lack of relevance features in this list, especially when we expect them to be more important for this domain. We believe that the reason for their non-appearance is the weakness of the trigger language model that we used. Due to time constraints, this model was trained on only a small amount of data. Since the vocabulary size employed was roughly 50,000 words, many of the trigger pairs simply did not appear in the training data. Nevertheless, simple word features are able to do a commendable job at segmenting.

The letters N. and C. naturally make their appearance early on because of the fact that the data is tokenized for speech processing (whence C. N. N. rather than CNN), and the network identification information is often given at the end and beginning of news segments (C. N. N.'S RICHARD BLYSTONE IS HERE TO TELL US...). Similar explanations of the word features are easy to guess from a perusal of Figure 8. In Figure 9 we show several typical sequences of reference and hypothesized segment boundaries on test data. Because the segments are generally much longer in this domain, we display them in blocks of 400 sentences.

7 Conclusions

We have presented and evaluated a new statistical model for segmenting unpartitioned text into coherent fragments. We leverage long- and short-range language models, as well as automatic feature induction techniques, in the design of this model. In this work we rely exclusively on simple lexical features, including a topicality measure called *relevance* and a number of vocabulary features that are induced from a large space of candidate features.

We have proposed a new probabilistically motivated error metric for the assessment of segmentation algorithms. Qualitative assessment as well as the evaluation of our algorithm with this new metric demonstrates its effectiveness in two very different domains, *Wall Street Journal* articles and broadcast news transcripts.

Our immediate application of this model will be to the video-on-demand application called Informedia [CKM+95]. We intend to mix simple audio and video features such as statistics from pauses, black frames, and color histograms with our lexical features in order to segment news broadcasts into component stories. Other applications that we have not explored in this paper include automatic inference of subtopic structure for information retrieval, document summarization, and improved language modeling.

References

[BBL97] D. Beeferman, A. Berger, and J. Lafferty. Accounting for distance in a long-range language model. In *Proceedings of the ACL*, Madrid, Spain, 1997. Submitted.

[BDD96] A. Berger, S. Della Pietra, and V. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[CKM+95] M. Christel, T. Kanade, M. Mauldin, R. Reddy, M. Sirbu, S. Stevens, and H. Wactlar. Informedia digital video library. *Communications of the ACM*, 38(4):57–58, 1995.

[DDL97] S. Della Pietra, V. Della Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(3), 1997. In press.

[Hea94] M.A. Hearst. Multi-paragraph segmentation of expository text. In *Proc. ACL*, 1994.

[JMRS91] F. Jelinek, B. Merialdo, S. Roukos, and M. Strauss. A dynamic language model for speech recognition. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 293–295, February 1991.

[Kat87] S. Katz. Estimation of probabilities from sparse data for the langauge model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-35(3):400–401, March 1987.

[Kd90] R. Kuhn and R. de Mori. A cache-based natural language model for speech recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12:570–583, 1990.

[KF94] H. Kozima and T. Furugori. Segmenting narrative text into coherent scenes. *Literary and Linguistic Computing*, 9:13–19, 1994.

[Koz93] H. Kozima. Text segmentation based on similarity between words. In *Proc. ACL*, 1993.

[LP95] D. J. Litman and R. J. Passonneau. Combining multiple knowledge sources for discourse segmentation. In *Proc. ACL*, 1995.

[LRR93] R. Lau, R. Rosenfeld, and S. Roukos. Adaptive language modeling using the maximum entropy principle. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 108–113. Morgan Kaufman Publishers, 1993.

[Nea92] R. Neal. Connectionist learning of belief networks. *Artificial Intelligence*, 56:71–113, 1992.

[Rey94] J. C. Reynar. An automatic method of finding topic boundaries. In *Proc. ACL*, 1994.

[You91] G. Youmans. A new tool for discourse analysis: The vocabulary-management profile. *Language*, 67:763–789, 1991.